Dissecting call-by-need by customizing multi type systems

Maico Leberle

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2021IPPAX023

Thèse de doctorat

Dissecting call-by-need by customizing multi type systems

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à l'École Polytechnique

École doctorale n°626 École doctorale de l'Institut Polytechnique de Paris (EDIPP)
Spécialité de doctorat : Informatique

Thèse présentée et soutenue à Palaiseau, le 7 mai 2021, par

LEBERLE, MAICO CARLOS

Composition du Jury :

HERBELIN, Hugo Directeur de recherche, Université de Paris (IRIF)	Président
RONCHI DELLA ROCCA, Simona Professeure émérite, Università di Torino (Dipartimento di Informatica)	Rapporteure
MAZZA, Damiano Directeur de recherche, CNRS	Rapporteur
BONELLI, Eduardo Augusto Professor, Stevens Institute of Technology (Schaefer School of Engineering and Science)	Examinateur
KESNER, Delia Professeure, Université de Paris (IRIF)	Examinateur
MILLER, Dale Directeur de recherche, École Polytechnique (LIX)	Directeur de thèse
ACCATTOLI, Beniamino Chargé de recherche, École Polytechnique (LIX)	Co-directeur de thèse

Index

Chapter 1

Introduction

1.1 Background and objectives

The λ-calculus is a model of computation underlying a variety of technological tools. While it is mostly known for being the backbone of functional programming languages, it is also used to model higher-order logic programming as well as the computational device in numerous proof assistants. Generally speaking, the λ-calculus is a perfect fit for formalizing higher-order computations, those where functions are considered values themselves; this even includes the imperative programming languages which implement higher-order functions.

As far as expressiveness is concerned, it is well-known that the λ-calculus is as expressive as Turing machines; we say that the λ-calculus is *Turing complete*. This result was obtained during the 1930s, around the time the λ-calculus was introduced to the community.

On the contrary, stating a similar property with respect to the *efficiency* of the λ-calculus was considerably more difficult to prove: while there exists an evident way to measure the computational effort required to run a program from beginning to end in a Turing machine, doing so with respect to a program in the λ-calculus is not at all self-evident. The main reason is that the syntax and computational rules of the λ-calculus are very simple and high-level. This implies that running a program in the λ-calculus requires first providing an implementation schema for a concrete target computer architecture—that is, providing a specification of the computational process sufficiently fine-grained and tuned to be suited for running on said computer architecture. Once the implementation schema has been established, one then needs to find the right way to measure its computational effort, which is also extremely delicate and has historically amounted to considerable work.

Indeed, it was only in 2014 that Accattoli and Dal Lago proved in their [AL16] that the λ-calculus can be fully simulated in Turing machines with a polynomially bounded overhead in time. This is called the *invariance* result for the λ-calculus, and implies that all the (super-)polynomial complexity classes of the λ-calculus match exactly those of Turing machines—where the latter are taken to be the standard for defining complexity classes.

As a matter of fact, there existed prior results concerning the invariance of the λ-calculus, like Accattoli and Dal Lago's "On the Invariance of the Unitary Cost Model for Head Reduction"[AD12]. However, it is only in [AL16] that the most general case of the λ-calculus is targeted. This is achieved in [AL16] by targeting in particular the standard evaluation strategy of the λ-calculus, called *leftmost-outermost*.

Proving this invariance result required adapting the reduction relation to implement an efficiency

technique that the authors call *usefulness*. Roughly speaking, a useful reduction relation only substitutes those variable occurrences whose substitution contributes to the creation of a β-redex, and are thus necessary for the execution of the program to reach the corresponding normal form.

The main objective of this thesis is to provide results in the theory of useful reductions. In this sense, our setting of choice for studying usefulness is *call-by-need*. Recently, Balabonski, Barenbaum, Bonelli and Kesner presented in their [Bal+17] a reduction relation which they call "*Strong Call by Need*", which performs strong reduction while being conservative with respect to the original call-by-need reduction[1]. Thus, we take a significant portion of the reduction relation in [Bal+17] and adapt it to satisfy the usefulness criterion.

1.2 Tools

Multi types

We shall give a type-theoretical presentation of the normalization process of our useful call-by-need reduction relation. Given the intrinsically complex nature of the subject, we shall first cover several related cases before attaining the final one. Thus, in the process of deriving the type system for the useful call-by-need relation, we shall study several other reduction relations and derive a separate type system for each one of them, in a well-principled and incremental manner.

All of these type systems fall into the category of *multi type systems*. Multi types were introduced as *non-idempotent intersection types* in Philippa Gardner's "Discovering Needed Reductions Using Type Theory"[Gar94]. The reasons for which multi type systems have recently become trending are numerous; our interest in them lies particularly in that each of them are designed specifically to characterize a certain notion of normalization. Moreover, type derivations shall provide upper bounds on the consumption of resources in the corresponding normalization process. These upper bounds may be obtained in such a precise way that some of them are even *exact* bounds, or simply *measures* of the normalization process. For this reason, multi types are said to be *resource aware*, as they allow us to reason about resources from a type-theoretical standpoint.

Multi types were first used to give precise measurements on operational quantities in de Carvalho's "Execution Time of lambda-Terms via Denotational Semantics and Intersection Types" [Car09]. The starting point therein is a multiset-based relational model of Linear Logic[2], called *System R*, which induces a relational semantics of the type-free λ-calculus. De Carvalho then proves that the size of type derivations and the size of types in System R are closely related to the execution time of λ-terms in Krivine's machine.

Since then, [Car09] has inspired numerous lines of research, and its results have been refined and extended to encompass different normalization processes as well as different operational measurements—this thesis intends to be one of such recent works.

Unlike de Carvalho, however, abstract machines are not our operational semantics of reference. Instead, when defining our reduction relations we shall resort to the *Linear Substitution Calculus*—or LSC for short[3]—a calculus which may be seen as intermediate in between the λ-calculus and abstract machines.

[1] As introduced by Wadsworth[Wad71]

[2] The relational semantics of Linear Logic is arguably its simplest *denotational* semantics.

[3] The LSC is a calculus implementing sharing mechanisms and bearing significant connections to environment-based abstract machines, like Krivine's—an in depth study of said connections is given in [ABM14].

Operational semantics

Among the different kinds of argument-passing styles that reduction relations may adopt, we shall be concerned with the three that we dim central to the theory of the λ-calculus: call-by-name, call-by-value and call-by-need:

- Since call-by-name is by far the most studied one—from a theoretical and foundational point of view—we shall not present novel results about it, and just use the ones from the literature to build upon. Call-by-name is the argument-passing technique implemented by the standard evaluation strategy of the λ-calculus, called *leftmost-outermost* reduction.
- Call-by-value has been relatively neglected when compared to the theoretical foundations given to call-by-name. Despite that, it is probably the most implemented argument-passing style among programming languages.
- Call-by-need, is probably the least studied one, theoretically-wise. In practice, call-by-need has *nevertheless* received considerable attention since its inception; however, this has always been restricted to technological implementations—like the *Haskell* programming language, or the *Coq* proof assistant—or theoretical foundations that only covered the most basic case—of weak reduction on closed λ-terms. This work intends to revert this, by providing results that might expand its theoretical understanding and foundational solidity beyond the usual weak and closed settings.
  We shall begin our study of call-by-need by realizing that it may be understood (in the weak and closed setting) as a combination of call-by-name and call-by-value. Hence, some of the results given for call-by-name or call-by-value shall contribute to our understanding of call-by-need.

Our operational results are based on three recent and important advances in the operational theory of the λ-calculus, namely:

- Firstly, the publishing in 2017 of Balabonski, Barenbaum, Bonelli and Kesner's "Foundations of Strong Call by Need" [Bal+17], where the authors propose a first-ever, call-by-need strategy computing strong normal forms and acting on potentially open terms. This is a call-by-need strategy in the sense that it guarantees that arguments are *only reduced once* and *only if—and when—needed*[4].
- Secondly, the publishing of Accattoli and Dal Lago's "(Leftmost-outermost) Beta Reduction is Invariant, Indeed" [AL16] in 2016, where the authors provide a unitary time cost model of the λ-calculus and show it is polynomially related to that of Turing machines or random access machines.
  As explained above, it required implementing useful reduction on leftmost-outermost reduction. In addition, it requires defining the useful variant of leftmost-outermost in a formalism implementing some notion of sharing, like the LSC.
- Finally, Accattoli and Guerrieri isolated the so called *Open Call-by-Value* calculus in [AG16], a setting halfway between the most basic and restricted variant of the call-by-value λ-calculus and the most general one: respectively, they are the *Weak* Call-by-Value—which only acts on *closed* λ-terms—and the *Strong* Call-by-Value—which may act on *open* λ-terms. The value of this result is that Strong Call-by-Value may be seen as the Open Call-by-Value iterated under λ-abstractions.

[4]Remarkably, they use techniques involving multi types in [Bal+17] to prove that their strategy is complete with respect to β-reduction to strong normal forms.

1.3 Development and outcomes

The starting point of this thesis are the multi type systems given in the literature for the weak and closed variants of CbN—that is, System R given by de Carvalho—and of CbV—see Ehrhard's call-by-value variant of System R in [Ehr12]. Since multi types model CbN and CbV well, it was only natural to try to apply this technique to CbNeed, more so considering the fact that CbNeed may be seen as a combination of CbN and CbV—in the particular way we shall explain in Chapter 4 (CbN, CbV and CbNeed).

We believe that the understanding of Useful Open CbNeed, both operationally and type-theoretically, is likely the most interesting result in our work. It is presented in Chapter 8 (Useful Open CbNeed) and intends to connect two of the recent advances in the operational theory of the λ-calculus—that we listed in Subsect. 1.2 (Operational semantics): we begin by taking a reduction sub-relation from [Bal+17]—which we call *Open CbNeed*—then we take the notion of usefulness as presented in [AL16], and we finally use it to adapt Open CbNeed to perform useful reduction, thus obtaining Useful Open CbNeed.

As we shall see in Chapter 6 (Open CbNeed), Open CbNeed is itself interesting, in particular because it happens to be the natural way to factor the difficulties faced in the development of a theory of strong reduction in a call-by-need setting. Moreover, it constitutes the natural setting where the restriction to useful reductions may be implemented and studied, as it is a significantly simpler scenario compared to variants of call-by-need implementing *strong reduction*.

Another natural intermediate step to achieving the results for Useful Open CbNeed is understanding Open Call-by-Value, for which a multi type system with a class of type derivations providing exact bounds is given in [AG18]. Here, we shall extend the reduction relation of Open Call-by-Value to define the more general *Strong Call-by-Value* relation—which encompasses Open Call-by-Value—and then provide a multi type system whose notion of typability matches that of normalization in Strong Call-by-Value. As a matter of fact, the multi type system for Strong Call-by-Value served as inspiration for the one for Useful Open CbNeed during the preparation of this work. Nevertheless, we shall not merely present it as an intermediate step towards the latter—unlike all other multi type systems—but rather as a separate case study, interesting in itself.

We would also like to remark the importance of studying strong reduction in a call-by-value setting, both operationally and type-theoretically. This is a previous and necessary step to attaining the final goal of producing a multi type system characterizing and providing measurements for a strong and useful call-by-need reduction relation. Unfortunately, this final goal remains unachieved by the time we write these lines.

With intention to keep matters as simple as possible, all the case studies in this work shall morally follow the same structure, methodologically speaking:

- We begin by defining the reduction relation and proving that it enjoys some kind of determinism.
- Next, we provide a set of predicates that characterize the normal forms of said reduction relation. This may be extremely intricate, in particular in the cases of Open CbNeed and Useful Open CbNeed.
- We then move on to define the multi type system meant to characterize normalization in the reduction relation. To attain this, we must prove that the type system is sound and

complete with respect to normalizing terms. Needless to say, proving this requires an extremely detailed understanding of the operational aspects of the reduction relation from a type-theoretical perspective. An overview of the way in which we prove this—which is consistently applied to every case study—is presented in upcoming Chapter 3 (A bird's eye view).

Turning now to the features of the multi type systems, we hope that the reader may find them quite simple. They are derived in a principled way, following an incremental approach:

- The multi type system for Strong CbV is obtained almost effortlessly from the one given by Ehrhard for the weak and closed variant of CbV.
- The multi type system for Open CbNeed is obtained by adding a few typing rules to the one for CbNeed.
- Finally, the multi type system for Useful Open CbNeed is obtained from the one for Open CbNeed by refining the axioms. We believe this is quite remarkable, and was in fact our first intuition for understanding useful reduction from a (multi) type-theoretical perspective in the Open CbNeed setting.

Unfortunately, the same level of simplicity was not found at the rewriting-theoretical level. That is, while designing the multi type systems turned out to be relatively simple, conducting the operational studies of most of our case studies was not as easy as expected. For instance, the Open CbNeed and Useful Open CbNeed cases turned out to be extremely subtle, requiring in particular very long proofs for the characterization of their normal forms via predicates. This was somewhat surprising.

1.4 Considerations

Each one of the four case studies have two dedicated chapters, one where we study its operational aspects and another one where we study its multi type system. The distribution of contents goes as follows:

- In Chapter 4 (CbN, CbV and CbNeed) we develop an operational account on the weak and closed versions of call-by-name (CbN), call-by-value (CbV) and call-by-need (CbNeed).
- In Chapter 5 (Multi types for CbN, CbV and CbNeed) we analyze the multi type systems for CbN and for CbV given in the literature, and use them to derive the one for CbNeed.
- In Chapter 6 (Open CbNeed) we present the Open CbNeed evaluation strategy, which extends the CbNeed one by including reduction on (possibly) open terms, and reducing on arguments.
- In Chapter 7 (Multi types for Open CbNeed) we present the multi type system for the Open CbNeed evaluation strategy.
- In Chapter 8 (Useful Open CbNeed) we refine the Open CbNeed evaluation strategy to make it useful.
- In Chapter 9 (Multi types for Useful Open CbNeed) we present the multi type system for the Useful Open CbNeed evaluation strategy.
- In Chapter 10 (Strong CbV) we present the Strong CbV evaluation strategy.
- Finally, in Chapter 11 (Multi types for Strong CbV) we present the multi type system for the Strong CbV evaluation strategy.

With respect to the other chapters:

- Chapter 2 (Preliminaries) gives a relatively detailed presentation of topics. Although they are probably known to the reader, the purpose of this chapter is to agree on notation and

definitions.

- Chapter 3 (A bird's eye view) intends to extend the intuitions that we just gave here by incorporating some of the technical details introduced in Chapter 2 (Preliminaries). We also use Chapter 3 to show that the technical development in each of the case studies—in particular what concerns the multi type systems—follows the exact same structure, thus heightening the value of our approach.
- Chapter 12 (Conclusion) reviews the results attained throughout this work, and provides a list of some future lines of research that might derive from this work.

Proofs and appendices. We have made the presentation choice of keeping explanations and technical proofs apart. Thus, Chapter 4 (CbN, CbV and CbNeed) to Chapter 11 (Multi types for Strong CbV) mostly provide definitions, analyses and examples, while Chapter 13 mostly provides proofs.

In Chapter 13 (Technical appendix) we provide all the proofs for for this work. Of course, this excludes Chapter 2 (Preliminaries), Chapter 3 (A bird's eye view) and Chapter 12 (Conclusion). All other chapters, from Chapter 4 (CbN, CbV and CbNeed) to Chapter 11 (Multi types for Strong CbV), have numerous links to Chapter 13. Indeed, the complete technical development of each one of these chapters appears in Chapter 13 in the form of a dedicated section. In addition, every statement appearing in chapters from Chapter 4 (CbN, CbV and CbNeed) to Chapter 11 (Multi types for Strong CbV) is followed by links taking the reader back and forth Chapter 13 (Technical Appendix). Conversely, some (less important) statements only appear in Chapter 13, as they are considered to be relevant only as part of the technical development, and not for any meaningful conceptual understanding. We believe this should help the reader study a proof whenever he or she sees fit, while keeping matters as split as possible.

Abstract machines. The development of this thesis was considerably accompanied by a large amount of work on abstract machines. For both of Open CbNeed and Useful Open CbNeed, we simultaneously worked on them on three different levels: the operational semantics, the multi type system, and the derivation of an abstract machine.

The intention was to design these abstract machines in such a way that they would *distill* their respective evaluation strategies—in the sense presented in "Distilling Abstract Machines" [ABM14]. Additionally, they would be *bilinear* with respect to the evaluation strategy. For instance, given an Open CbNeed-normalizing term t, the number of transitions in an execution in the Open CbNeed abstract machine would have to be bilinearly related to the size of t and the number of multiplicative steps in the Open CbNeed-normalizing sequence starting in t. Similarly, the Useful Open CbNeed abstract machine would also have to satisfy a bilinear relation with respect to the evaluation strategy.

However, designing such abstract machines was too technically demanding, and so we had to focus our time on the development of the multi type systems, leaving the completion of the development of the abstract machines for future work. Despite that, let us stress the fact that the work on abstract machines was a valuable set of guiding principles both for the operational semantics and the design of the multi type systems in this work.

Contributions. This thesis contains results from two separate but connected worlds, namely the theory of operational semantics and the theory of types. Let us briefly list our contributions, especially the novelties of this work:

- *Operational semantics*: In this respect, the novelties in this work are Open CbNeed, Useful Open CbNeed and Strong CbV; the weak and closed evaluation strategies presented in Chapter 4 (CbN, CbV and CbNeed) are prior to this thesis.
  Open CbNeed is the result of joint work between my supervisor, Beniamino Accattoli, and myself. We should however say that it is actually an adaptation of the Strong Call by Need evaluation strategy from "Foundations of Strong Call by Need" [Bal+17]. Basically, the Open CbNeed evaluation strategy is a restriction of Strong Call by Need to the weak setting, where we have adjusted the derivation of its evaluation contexts in such a way as to match our needs for the design of the multi type system.
  Moreover, the development of Open CbNeed was carefully done as to, simultaneously, prepare the elements for the useful variant, the Useful Open CbNeed evaluation strategy. The isolation of needed variables is an example of this: they are presented for Open CbNeed and refined for Useful Open CbNeed.
  The latter, the Useful Open CbNeed evaluation strategy, was mostly my invention. But I had the expert and tireless help of my supervisor without whom I would not have succeeded.
  Finally, the development of the Strong CbV evaluation strategy is the result of joint work between Beniamino Accattoli, Giulio Guerrieri and myself. Andrea Condoluci and Claudio Sacerdoti Coen also contributed in the design Strong CbV, but they were mostly concerned in connecting the strategy to a bilinear abstract machine distilling it.

- *Multi type systems*: The multi type systems given for CbN and for CbV are not a novelty of our work, as we took them from "Execution time of λ-terms via denotational semantics and intersection types" [Car18] and "Collapsing non-idempotent intersection types" [Ehr12], respectively.
  On the contrary, all the other multi type systems in this work are novel. That is,
  - The CbNeed multi type system, presented in Chapter 5 (Multi types for CbN, CbV and CbNeed), was jointly developed by Beniamino Accattoli, Giulio Guerrieri and myself.
  - Both the Open CbNeed multi type system presented in Chapter 7 (Multi types for Open CbNeed), and the Useful Open CbNeed multi type system presented in Chapter 9 (Multi types for Useful Open CbNeed), were mostly developed by myself. My supervisor's close guidance was greatly helpful.
  - Finally, the Strong CbV multi type system, presented in Chapter 11 (Multi types for Strong CbV), was jointly developed by Accattoli, Guerrieri and myself.

- Finally, the work on abstract machines for Open CbNeed and for Useful Open CbNeed was done as a collaboration with Bruno Barras and Beniamino Accattoli. Although it has not made its way into this thesis, we believe the considerable effort we made on designing abstract machines shall soon bear its fruits.

Chapter 2

Preliminaries

2.1 The λ-calculus

The syntax of the λ-calculus is given by the following context-free grammar:

$$\Lambda\text{-terms } (\Lambda) \quad t, u, s, m \quad ::= \quad \mathsf{Var} \mid \lambda x.t \mid (tu)$$

where Var is taken to be any *countably infinite* set of syntactic objects, which we call *variables* and note $x, y, z, \tilde{x}, \ldots$. The same set of variables is silently used throughout the present work.

For any given $t \in \Lambda$, we define the set of its free and bound variables, respectively noted $\mathsf{fv}(t)$ and $\mathsf{bv}(t)$, as follows:

Free variables			Bound variables		
$\mathsf{fv}(x)$	$:=$	$\{x\}$	$\mathsf{bv}(x)$	$:=$	$\emptyset$
$\mathsf{fv}(tu)$	$:=$	$\mathsf{fv}(t) \cup \mathsf{fv}(u)$	$\mathsf{bv}(tu)$	$:=$	$\mathsf{bv}(t) \cup \mathsf{bv}(u)$
$\mathsf{fv}(\lambda x.t)$	$:=$	$\mathsf{fv}(t) \setminus \{x\}$	$\mathsf{bv}(\lambda x.t)$	$:=$	$\mathsf{bv}(t) \cup \{x\}$

A Λ-term with no free variables is called a *closed* term; dually, a Λ-term with free variables is called open.

Definition 1 (α-equivalence)**.**

Let $t, u \in \Lambda$. We say that t and u are "*α-equivalent*" if they only differ in the choice of bound variable names while remaining syntactically equal in any other aspect.

Remark 1 (Identifying α-equivalent terms)*.*

Throughout this work, we shall identify terms up to α-equivalence. This is a standard practice in the literature—except, of course, when α-conversion of terms is the object of study in itself—and is also known as *Barendregt's variable convention*.

That is, we shall always assume that the name of bound variables in a given term are all pairwise distinct, and that they are different from the names used for free variables. For example, $\lambda x.x$ and $\lambda y.y$ are α-equivalent terms and are thus identified as morally representing the same term, while $\lambda x.x$ and $\lambda x.x\,x$ are not.

Note that this identification concerns only the name of bound variables, and free variables are never identified unless they share the same name. Thus, we do not identify $x\,x$ and $y\,y$—since they are not α-equivalent. Nor do we identify $\lambda x.y$ and $\lambda y.y$, since in the former y is a free variable while in the latter the rightmost occurrence of y is *bound* by the leftmost occurrence, thus morally representing a different term.

2.1.1 β-reduction

The computational rule in the λ-calculus is β-reduction, whose base case is the following

Definition 2 (β-contraction)**.**
The smallest binary relation between Λ-terms satisfying

$$(\lambda x.t)u \mapsto_\beta t\{x \leftarrow u\}$$

is called the *β-reduction root step*. Here, $t\{x \leftarrow u\}$ represents the capture-avoiding, syntactic and simultaneous substitution of each occurrence of x in t by copies of u. We call this kind of substitution of variables as "*meta-level*" substitution.

Λ-terms of the form $(\lambda x.t)u$ are called *β-redexes* and we say that $(\lambda x.t)u$ *$\mapsto_\beta$-reduces to, or that it $\mapsto_\beta$-contracts to* $t\{x \leftarrow u\}$. We may also say that $t\{x \leftarrow u\}$ is the $\mapsto_\beta$-reduct of $(\lambda x.t)u$.

The following are examples of β-contraction

$$\begin{array}{rcccl} (\lambda x.x)u & \mapsto_\beta & x\{x \leftarrow u\} & = & u \\ (\lambda x.(xx))u & \mapsto_\beta & (xx)\{x \leftarrow u\} & = & (uu) \\ (\lambda x.y)u & \mapsto_\beta & y\{x \leftarrow u\} & = & y \\ (\lambda x.(\lambda y.x))y & \mapsto_\beta & (\lambda y.x)\{x \leftarrow y\} & =_\alpha & \lambda z.y \end{array}$$

In order to avoid capturing free variables, note that the last example implements α-renaming (of fresh variable z for bound variable y) before proceeding with the meta-level substitution.

Now, we would like to allow root step $\mapsto_\beta$ to be performed in any position of a Λ-term where there is a β-redex. For instance, we would like to have that $x((\lambda x.t)u)$ β-reduces to $x(t\{x \leftarrow u\})$. One simple way to do this is by means of *evaluation contexts*, which are nothing other than Λ-terms where one of its subterms is a *context hole*, noted $\langle\cdot\rangle$. This context holes play the role of placeholders for the subterms to which we perform the rewriting step $\mapsto_\beta$.

Evaluation contexts in the λ-calculus, called *general contexts* or simply *contexts*, are given in the form of a context-free grammar, much like the syntax of the λ-calculus itself:

$$C, C' ::= \langle\cdot\rangle \mid Ct \mid tC \mid \lambda x.C$$

The definitions of free and bound variables of a general context are extended from the ones for Λ-terms as expected:

Free variables			Bound variables		
$\mathsf{fv}(\langle\cdot\rangle)$	$:=$	$\emptyset$	$\mathsf{bv}(\langle\cdot\rangle)$	$:=$	$\emptyset$
$\mathsf{fv}(Ct)$	$:=$	$\mathsf{fv}(C) \cup \mathsf{fv}(t)$	$\mathsf{bv}(Ct)$	$:=$	$\mathsf{bv}(C) \cup \mathsf{bv}(t)$
$\mathsf{fv}(tC)$	$:=$	$\mathsf{fv}(t) \cup \mathsf{fv}(C)$	$\mathsf{bv}(tC)$	$:=$	$\mathsf{bv}(t) \cup \mathsf{bv}(C)$
$\mathsf{fv}(\lambda x.C)$	$:=$	$\mathsf{fv}(C) \setminus \{x\}$	$\mathsf{bv}(\lambda x.C)$	$:=$	$\mathsf{bv}(C) \cup \{x\}$

We write $C\langle t\rangle$ for the Λ-term obtained by replacing the hole $\langle\cdot\rangle$ in C by t. This *plugging* operation may capture variables—as it is usual with contexts in the literature. For instance, consider $(\lambda x.(t\langle\cdot\rangle))\langle x\rangle = \lambda x.(tx)$. Conversely, we write $C\langle\!\langle t\rangle\!\rangle$ when we want to stress that the context C does not capture the free variables of t.

Remark 2 (α-equivalence and evaluation contexts). Since contexts may capture free variables, we do not consider them up to α-equivalence. However, once a context C has been plugged with a $t \in \Lambda$, and potential captures have been established, then $C\langle t\rangle \in \Lambda$; consequently, α-equivalence considerations apply on $C\langle t\rangle$ as for any other Λ-term.

We now have all the needed concepts to finally define the operational semantics for the λ-calculus:

Definition 3 (β-reduction)**.**
The reduction relation $\to_\beta$ between Λ-terms, called *β-reduction*, is defined as follows:

$$\frac{t \mapsto_\beta u}{C\langle t\rangle \to_\beta C\langle u\rangle}$$

for all $t, u \in \Lambda$ and general context C.

Alternatively, we may write $\to_\beta := C\langle \mapsto_\beta \rangle$, saying that $\to_\beta$ is the *contextual closure* of $\mapsto_\beta$. When $C\langle t\rangle \to_\beta C\langle u\rangle$, we say that $C\langle u\rangle$ is the result of *contracting* the *β-redex* t in the evaluation context C.

Finally, we write $t \to_\beta^* u$ to express that there exists a (potentially empty) *$\to_\beta$-reduction sequence from t to u*. That is, t $\to_\beta$-reduces to u if there exists $n \geq 0$ such that

$$t \underbrace{\to_\beta \ \ldots \ \to_\beta}_{n} u$$

In particular, when $n = 0$ we have that $t = u$.

2.1.2 Normalization in the λ-calculus

As is standard, we say that $t \in \Lambda$ is in *$\to_\beta$-normal form*, or that t *is a $\to_\beta$-normal form*, when there exists no other $u \in \Lambda$ such that $t \to_\beta u$.

Normal forms in the λ-calculus are known to be unique, and we say that it is *confluent*. As proven by Church and Rosser, if $t \to_\beta^* u$ and $t \to_\beta^* s$, where both u and s are $\to_\beta$-normal forms, then $u = s$.

The notion of normal forms is not exclusive to the λ-calculus, as it may be generalized to any rewriting system. Thus, the following notions, defined in terms of the λ-calculus, may also be defined for every reduction relation in this work:

Definition 4 (Weakly normalizing, strongly normalizing and diverging Λ-terms)**.**
Let $t \in \Lambda$. Then,

- We say that t is *$\to_\beta$-weakly normalizing* if *there exists* $u \in \Lambda$ such that $t \to_\beta^* u$ and u is in $\to_\beta$-normal form.
- We say that t is *$\to_\beta$-strongly normalizing* if *all* $\to_\beta$-reduction sequences starting in t reach a $\to_\beta$-normal form.
- We say that t is *$\to_\beta$-diverging* if it is not $\to_\beta$-weakly normalizing. That is, if *there exists no* $u \in \Lambda$ such that $t \to_\beta^* u$ and u is in $\to_\beta$-normal form[1].

[1] Although it is not used in this work, there exists a further refinement on the notion of $\to_\beta$-divergence—akin to the refinement defined for $\to_\beta$-normalization: we say that t is *$\to_\beta$-weakly diverging* if there exists a $\to_\beta$-reduction sequences that diverges, and that t is *$\to_\beta$-strongly diverging* if all $\to_\beta$-reduction sequences diverge. Note that the notion of $\to_\beta$-divergence described above matches that of $\to_\beta$-strong divergence described here.

The typical example of a $\to_\beta$-diverging term is $\Omega \coloneqq \delta\delta$, with $\delta \coloneqq \lambda x.(xx)$. Note that Ω can only $\to_\beta$-reduce to itself—*i.e.*, $\Omega \ \to_\beta \ \Omega$—and so there cannot be $\to_\beta$-reduction sequences starting in Ω and reaching a $\to_\beta$-normal form.

2.1.3 Evaluation strategies

As we can see in the definition of general contexts, given $(tu) \in \Lambda$ we can proceed to contract a β-redex either in t or in u. In this respect, the λ-calculus does not force redexes to be contracted in any particular order, thus making it a non-deterministic model of computation.

The ordinary, non-deterministic λ-calculus enjoys many interesting properties by itself, and serves as a common framework and foundational background to all of its variants. However, we shall not work directly on the λ-calculus here. Instead, we shall be interested in some of its *evaluation strategies*, which are variants satisfying a relaxed notion of determinism called the *diamond property*:

Definition 5 (Diamond property)**.**

Let $\to$ is a sub-relation of $\to_\beta$ composed of $\to_1$, ..., $\to_n$. That is, let $\to_1 \ \cup \ldots \cup \ \to_n =: \to \subseteq \to_\beta$.

We say that $\to$ satisfies the *diamond property* if for every $t, u, s \in \Lambda$ and $1 \leq i, j \leq n$, the fact that $s \ {}_j\!\leftarrow \ t \ \to_i \ u$ implies the existence of $m \in \Lambda$ such that $s \ \to_i \ m \ {}_j\!\leftarrow \ u$.

The diamond property implies *uniform normalization*: if there is a normalizing sequence from t, then there are no diverging sequences from t. Additionally, it implies the *random descent property*: all normalizing sequences from t have the same length and, in our current case, the diamond diagram preserves also the kind of step (and so all normalizing sequences have the same number of each kind of steps).

Definition 6 (Evaluation strategies)**.**

Given $\to \subseteq \to_\beta$, we say that $\to$ is an *evaluation strategy* of $\to_\beta$ if it satisfies the diamond property.

While Definition 6 shall be adapted to formalisms other than the λ-calculus when considering our different case studies, its essence shall remain the same.

Let us stress the fact that considering sub-relations that satisfy the diamond property is enough to cover our purposes[2]. Indeed, if we consider all possible reduction sequences starting at a given term, we see that if any one of them reaches a normal form, then all of them do, because the diamond property is a strong form of confluence. This property also ensures that the *number* and *kind* of steps among the reduction sequences starting at a given term are *exactly the same*.

Thus, and since our quantitative results shall only concern the number and kind of steps in normalizing reduction sequences, it is enough that the reduction relations in this work satisfy the diamond property.

[2]In the theory of the λ-calculus, evaluation strategies are usually defined as deterministic sub-relations of β-reduction. Hence, our definition of evaluation strategies is a more relaxed one, where we replace determinism for the diamond property.

2.1.4 Different flavors of evaluation strategies

As explained in Chapter 1 (Introduction), all our evaluation strategies fall into one of the following three categories, depending on the conditions imposed on arguments:

Evaluation strategies implementing *call-by-value* are such that, upon encountering a β-redex, they first evaluate the argument until reaching a (partial or full) normal form before contracting the redex. Thus, we say that call-by-value only contracts redexes when the argument is a *value*, hence the name.

Evaluation strategies implementing *call-by-name* do not impose any conditions on the shape of the argument of a redex, and proceed directly to contract the redex. Because of this lack of restriction on arguments, call-by-name is known to reach a normal form more often than call-by-value.

Finally, evaluation strategies implementing *call-by-need* do not impose restrictions on arguments either, and are thus said to be termination-equivalent to their call-by-name versions. But unlike call-by-name, arguments needed in call-by-need reduction sequences are *reduced only once*. Moreover, while call-by-value also reduces arguments only once, it does so even if the argument is not needed. Hence, call-by-need is said to be a "call-by-value-like efficient implementation of call-by-name", and is thus placed in between the two worlds. We shall extensively explore this feature of call-by-need in Chapter 4 (CbN, CbV and CbNeed).

2.1.5 Degrees of generality in β-reduction

Recall that $\to_\beta$-reduction is defined as the contraction of a β-redex *in any subterm* of the overall Λ-term, including under λ-abstraction. This degree of generality and non-determinism is sometimes too much, especially in technological applications, which require a deterministic reduction relation to perform their computations. Moreover, the different technological applications that use the λ-calculus to model their computations may have quite different needs in terms of generality.

For example, functional programming languages take closed terms as initial programs, proceed with their computations in a deterministic fashion, and stop evaluating a program when it reaches the form of a λ-abstraction.

On the one hand, and regarding free variables, note that since β-reduction does not create new free variables[3], we note that the hypothesis of closed terms suffices to model functional programming languages.

On the other hand, and regarding which subterms may be reduced, we say that functional programming languages perform *weak* reduction, as they stop execution upon reaching a λ-abstraction, even if the latter contains β-redexes in its body. For example, take $t \coloneqq \lambda x.\Omega$, where Ω is the usual diverging Λ-term, and note that while t is not in $\to_\beta$-normal form—and could never be so, because Ω diverges—it is in normal form when considering weak reduction.

On the opposite side, proof assistants impose no restrictions on the set of free variables of a term. Moreover, they contract each and all of the β-redexes, even those contained in the body of a λ-abstraction. Following our running example, proof assistants would continue reducing $t \coloneqq \lambda x.\Omega$—indefinitely.

[3]That is, if $t \to_\beta u$, then $\mathsf{fv}(u) \subseteq \mathsf{fv}(t)$. This is trivially provable by induction on the evaluation context C such that $t = C\langle s\rangle \to_\beta C\langle s'\rangle = u$, with $s \mapsto_\beta s'$.

These features may be expressed in the form of two separate axes, and used to categorize reduction relations as follows[4]:

1. A reduction relation that does not reduce under λ-abstractions is called *weak*, while one that does is called *strong*.
2. A reduction relation defined for closed terms is called *closed*. Conversely, one that is defined for (possibly) open terms is called *open*.

Remark 3 (Weakly/strongly normalizing terms and weak/strong reduction are not to be confused). Since there is a clash in the names to express these concepts, it should be remarked here that weakly and strongly normalizing are properties of terms, while weak and strong refer to reduction relations.

2.2 The Linear Substitution Calculus

2.2.1 Syntax

The Linear Substitution Calculus, or LSC for short, plays a central role throughout this work and is used to define virtually all of our evaluation strategies. Its syntax is given by extending the one for the λ-calculus with a fourth production, as follows:

$$(\Lambda_{\mathsf{L}}) \quad t, u, s, m \quad ::= \quad \mathsf{Var} \mid \lambda x.t \mid (tu) \mid t[x{\leftarrow}u]$$

Note that every Λ-term is also a Λ_{L}-term—*i.e.*, $\Lambda \subset \Lambda_{\mathsf{L}}$. The production $t[x{\leftarrow}u]$ should be taken to be a simple syntactic sugar presentation of the traditional let $x = u$ in t from the literature. We call $[x{\leftarrow}u]$ an *explicit substitution*, or *ES* for short.

As for let-constructs, ESs are a form of binding: variable x is bound in t in closure $t[x{\leftarrow}u]$. Given a $t \in \Lambda_{\mathsf{L}}$, the set of its free and bound variables, respectively noted $\mathsf{fv}(t)$ and $\mathsf{bv}(t)$, are defined as expected:

Free variables			Bound variables		
$\mathsf{fv}(x)$	$:=$	$\{x\}$	$\mathsf{bv}(x)$	$:=$	$\emptyset$
$\mathsf{fv}(tu)$	$:=$	$\mathsf{fv}(t) \cup \mathsf{fv}(u)$	$\mathsf{bv}(tu)$	$:=$	$\mathsf{bv}(t) \cup \mathsf{bv}(u)$
$\mathsf{fv}(\lambda x.t)$	$:=$	$\mathsf{fv}(t) \setminus \{x\}$	$\mathsf{bv}(\lambda x.t)$	$:=$	$\mathsf{bv}(t) \cup \{x\}$
$\mathsf{fv}(t[x{\leftarrow}u])$	$:=$	$(\mathsf{fv}(t) \setminus \{x\}) \cup \mathsf{fv}(u)$	$\mathsf{bv}(t[x{\leftarrow}u])$	$:=$	$\mathsf{bv}(t) \cup \{x\} \cup \mathsf{bv}(u)$

The concept of *closed* and *open* Λ_{L}-terms is defined as expected as well.

2.2.2 Where do explicit substitutions come from?

Explicit substitutions, as the name suggests, make the process of substitution explicit. Recall that the process of substitution, central to β-reduction—see Subsect. 2.1.1 (β-reduction)—relies on a kind of substitution that we call *implicit* substitution: $t\{x{\leftarrow}u\}$ represents the result of the syntactical and *simultaneous* substitution of *every* occurrence of x in t by copies of u.

The definition of meta-level substitutions is in fact unsuited for technology-related applications (whose procedures need to be concrete enough in order to perform computations). Thus, explicit

[4]In Chapter 3 (A bird's eye view), we shall explain how each of the evaluation strategies in this work combines these axes in a meaningful way.

substitutions may be considered a tool bridging the gap between the formal presentation of the λ-calculus and its concrete applications.

Historically, this has been achieved via the implementation of *abstract machines*. Peter J. Landin designed the first-ever abstract machine—called SECD [Lan64]—intended as a target for functional programming languages compilers—*i.e.*, for the λ-calculus and its higher-order evaluation. Since then, a considerable number of abstract machines have been studied, all of which are more or less explicit in how the substitution process should be carried out.

Abstract machines are, evidently, implementation-oriented, so their specific incarnation of the substitution process is somewhat ad-hoc rather than fundamental to the theory of substitutions. To address this theoretical lack, an alternative is resorting to an operational-semantics approach, consisting in defining the substitution process via some notion of a *sharing mechanism*.

A first such notion was proposed by de Bruijn, dating back to as far as the 1970's. In his [Bru72], he proposed the use of terms with *indices*, introducing his famous "*de Bruijn notation*" to the community. He later defined, in his [Bru78], a set of reduction rules that depend themselves on the indices of terms.

De Bruijn's work eventually evolved to what we know today as *explicit substitutions*. Many formulations of explicit substitutions were conceived and adapted to a considerable number of λ-calculi. For example, in their "Explicit Substitutions"[Aba+91] Abadi, Cardelli, Curien and Lévy define a so-called $\lambda\sigma$-calculus which is presented as a "useful bridge between ordinary λ-calculus and concrete implementations" [Aba+91].

2.2.3 Origin of the LSC

While the LSC is yet another calculus with explicit substitutions, among many other, there actually are certain subtle features that made it our formalism of choice for defining reduction relations. In particular, the LSC provides a sharing mechanism suitable for the call-by-need reduction relations defined in this work while, at the same time, keeping syntax bookkeeping (arguably) to a minimum of complexity. Let us now emphasize what we consider to be the two main operational features of the LSC:

- *Linear substitutions*. The LSC is particularly appropriate for making the substitution mechanism fine-grained enough to allow for substitutions for *single* variable occurrences. Note that this is in contrast with meta-level substitutions, where *all* variable occurrences get *simultaneously* replaced.
  Substitutions for a single variable occurrence—originally called *partial substitutions* but unsurprisingly called "*linear substitutions*" in the LSC—were envisioned by de Bruijn in his [Bru87], and then by Severi and Poll in their [SP87]. Robin Milner then took the idea of partial substitutions and derived the λ_{sub}-calculus [Mil06]. Milner's λ_{sub}-calculus is particularly relevant to the LSC because the latter may be seen as an extension of the former with the added notion of *distance* β.
- *Distance* β. As it was later shown by Accattoli [Acc18], the LSC is operationally isomorphic to (an abstract variant of) Linear Logic proof nets [Gir87].
  This property of the LSC is partly due to the fact that the cut-elimination process defined on proof nets acts "*at a distance*". Said property is embodied by the LSC in the way it contracts (a generalization of) β-redexes, where the λ-abstraction and its corresponding argument may be

separated by a (possibly empty and) finite sequence of explicit substitutions. Said differently, the notion of β-redex in the λ-calculus gets generalized in the LSC into redexes of the following form:

$$(\lambda x.t)\,[y_1 \leftarrow u_1]\ \ldots\ [y_b \leftarrow u_n]s$$

noting in particular that if $n = 0$ then $(\lambda x.t)\,u$ is just a β-redex from the λ-calculus.
Originally known as "*distance* β", the first reduction rules contracting this generalized kind of redexes were used in Accattoli and Kesner's *structural* λ*-calculus* [AK10].

Historically speaking, the LSC was derived by Accattoli and Kesner as the evolution of the structural λ-calculus combined with Milner's λ_{sub}-calculus, thus obtaining a calculus with explicit substitutions implementing both *linear substitutions* and *distance* β.

The LSC has been proven to enjoy interesting rewriting properties, including (but not limited to) having a residual system and a theory of standardization —see *e.g.* [Acc12; Acc18; Acc+14]. Additionally, we claim that the LSC has a closer resemblance with environment-based abstract machines[5] than other calculi with sharing mechanisms. This can be justified by Accattoli, Barenbaum and Mazza's [ABM14], where they prove that certain evaluation strategies formalized in the LSC and their corresponding environment-based abstract machines can simulate each other; even more remarkably, they do so while proving that the length of a given reduction sequence and the associated time-complexity in its abstract machine are linearly related.

Before we conclude, let us stress again the connections between the LSC and Linear Logic proof nets. Linear Logic, introduced by Jean-Yves Girard in his famous "Linear Logic" [Gir87] constitutes a useful tool for performing fine resource-consumption analyses of higher-order evaluation. In other words, Linear Logic can be thought of as a kind of resource-refinement applied to different proof systems, which therefore can be exploited to obtain a finer-grained cut-elimination process—particularly in terms of resources.

For all these reasons, we use the LSC to define our evaluation strategies. The only exception is the Open CbNeed evaluation strategy—and its *useful* variant; see Chapter 6 (Open CbNeed) and Chapter 8 (Useful Open CbNeed), respectively—which we formalize in a variant of the LSC.

2.2.4 The λ-calculus as a LSC

Just like β-reduction for the λ-calculus was defined by using *general* evaluation contexts based on Λ-terms, we now define the LSC evaluation contexts, with which we define the reduction relation in the LSC setting. LSC evaluation contexts are based on Λ_{L}, and are obtained via the following context-free grammar:

$$\text{LSC evaluation contexts} \qquad D, D' \ ::= \ \langle\cdot\rangle \mid Dt \mid tD \mid \lambda x.D \mid D[x \leftarrow t] \mid t[x \leftarrow D]$$

That is, LSC evaluation contexts are an extension to the notion of general evaluation contexts, where we add two productions to encompass the presence of ESs in the calculus.

In addition, we adapt the definition of free and bound variables of LSC evaluation contexts as expected; namely, by extending the one for general contexts with the following equations regarding ESs:

[5] Landin's SECD: in their [ABM14], Accattoli, Barenbaum and Mazza carry out an interesting approach to pairing the reduction relation in several evaluation strategies are formalized as LSC-calculi

Free variables			Bound variables		
$\mathsf{fv}(D[x{\leftarrow}t])$	$:=$	$(\mathsf{fv}(D) \setminus \{x\}) \cup \mathsf{fv}(t)$	$\mathsf{bv}(D[x{\leftarrow}t])$	$:=$	$\mathsf{bv}(D) \cup \{x\} \cup \mathsf{bv}(t)$
$\mathsf{fv}(t[x{\leftarrow}D])$	$:=$	$(\mathsf{fv}(t) \setminus \{x\}) \cup \mathsf{fv}(D)$	$\mathsf{bv}(t[x{\leftarrow}D])$	$:=$	$\mathsf{bv}(t) \cup \{x\} \cup \mathsf{fv}(D)$

Moreover, the following subclass of LSC evaluation contexts is used in almost every evaluation strategy:

$$\text{Substitution contexts} \qquad S, S' \ ::= \ \langle\cdot\rangle \mid S[x{\leftarrow}t]$$

That is, substitution contexts are nothing but a context hole followed by a possibly empty list of ESs.

We can now define the reduction relation for the λ-calculus in this LSC setting, which follows the same definition schema as for β-reduction. A major difference between the ordinary λ-calculus and its LSC variant is that the root-step $\mapsto_\beta$ used in the ordinary λ-calculus gets split in two[6]:

$$\begin{array}{lrcl} \text{Multiplicative root-step} & S\langle\lambda x.t\rangle u & \mapsto_{\mathsf{m}} & S\langle t[x{\leftarrow}u]\rangle \\ \text{Exponential root-step} & D\langle\!\langle x\rangle\!\rangle[x{\leftarrow}t] & \mapsto_{\mathsf{e}} & D\langle\!\langle t\rangle\!\rangle[x{\leftarrow}t] \end{array}$$

where the double angle-bracketing on the definition of exponential root-steps means that $x \notin \mathsf{bv}(D)$, as explained above.

Note that in multiplicative root-steps we perform *distance* β as explained above, while in exponential root-steps we perform *linear substitutions*. Moreover, note that performing a multiplicative root-step does not involve performing the meta-level substitution that would be triggered in the ordinary λ-calculus setting upon contraction of a β-redex. Instead, we simply add the explicit substitution $[x{\leftarrow}u]$, delaying the substitutions. In this way, the LSC implements distance β and linear substitutions as separate "*components*" of the reduction relation.

Remark 4 (α-renaming of bound variables).

As explained in Remark 1 (Identifying α-equivalent terms), in this text we consider terms up to α-equivalence. This affects in particular the duplication mechanism in the exponential root step defined above, where the bound variables of the duplicated term t become different from the ones in the original term inside the explicit substitution.

Technically speaking, the exponential root step should rather be defined as

$$D\langle\!\langle x\rangle\!\rangle[x{\leftarrow}t] \mapsto_{\mathsf{e}} D\langle\!\langle t^\alpha\rangle\!\rangle[x{\leftarrow}t]$$

where t^α represents term t after its bound variables have been changed for new ones.

That said, it turns out that this implementation is not completely necessary, since Barendregt's variable convention lets us assume that the bound variables in each of the copies of t are different from each other.

We shall however use the t^α notation in the remainder of the thesis when we want to stress the fact that the bound variables of a given duplicated term are not the same as the ones in the original term. This is particularly relevant in the proofs of the multi type systems, where we must meticulously consider the membership of a given variable to the domain of a given type context.

Let us define the reduction relation $\to_L$ of the LSC variant of the λ-calculus as the contextual closure of the multiplicative and exponential root-steps by LSC evaluation contexts. Formally,

$$\to_L := \ D\langle \mapsto_{\mathsf{m}} \cup \mapsto_{\mathsf{e}} \rangle \ = \ D\langle \mapsto_{\mathsf{m}} \rangle \cup D\langle \mapsto_{\mathsf{e}} \rangle$$

[6]The terminology "*multiplicative*" and "*exponential*" is taken from Linear Logic—see *e.g.* [Acc15].

Given a $\to_L$-reduction sequence $d : t \to_L^* u$, we note with $|d|$ the length of d, and with $|d|_{\mathsf{m}}$ and $|d|_{\mathsf{e}}$ the number of multiplicative and exponential steps in d, respectively.

When using the LSC as the base formalism for the definition of an evaluation strategy, we shall say that we are giving it a "*micro-step* semantics". This expression is in opposition to the usual notion of *small-steps* semantics that one usually gives to sub-relations of the ordinary λ-calculus. Micro-step semantics should be considered as a decomposition of small-steps semantics consisting in *linearly substituting variable occurrences*; that is, substituting variables *one at a time*, instead of performing meta-level substitution like in small-steps semantics.

2.3 Usefulness

Let us shortly discuss what we call *usefulness*, a property about the substitution process of a given sub-relation. To understand the value of usefulness, we first need to overview how the λ-calculus relates to the complexity theory branch of mathematics. That is, we need to discuss reasonability of cost models.

2.3.1 Reasonable time cost models of the λ-calculus

For each one of the case studies in this work, we shall present an operational semantics in the form of an evaluation strategy, and a multi type system characterizing its normalization. Furthermore, type derivations shall contain quantitative information regarding the normalization process of their subject. We then say that the quantitative information of each one of the normalization processes admits a type-theoretical representation. However, what does this information really mean, in the broader sense? In particular, how does any of the quantitative results in this work relate to more general notions of time complexity?

In the Complexity Theory community, the standard is to define complexity classes in terms of more machine-oriented models of computation, like Turing machines or random access machines. Indeed, most complexity classes consist on decision problems solvable by a Turing machine. These complexity classes are differentiated by the time or space a Turing machine consumes to solve the decision problem. For instance, the time complexity class $\mathcal{P}$ of polynomially decidable problems consists, precisely, of all those decision problems which can be decided by a Turing machine in polynomial time with respect to the size of the input.

The λ-calculus being a rather abstract model of computation, it is not clear how a reduction relation defined in the λ-calculus may be implemented in a Turing machine, in particular regarding its overhead. In terms of complexity, may reduction sequences in the λ-calculus be related with a polynomial overhead in time in a Turing machine? In fact, this problem is known to have been an arduous task. The complication came from the fact that it was not evident at all that β-steps, which are called the *unitary* cost model of the λ-calculus, were polynomially related to transition steps in Turing machines.

A positive answer to the question of the polynomial relation between the λ-calculus and Turing machines, expressed by saying that β-reduction is a *reasonable*, or *invariant*, time cost model of the λ-calculus, would mean that polynomial and super-polynomial complexity classes may be defined model-independently. That is, that they do not depend on the model of computation to be defined. On the other hand, a negative answer would somewhat severely undermine the relevance of the

λ-calculus as a model of computation, since the cost of computing programs whose base formalism is the λ-calculus could thus be too big to be worth using the λ-calculus at all.

In their 1984 paper [SE84], Slot and van Emde Boas introduced a complexity-theoretical version of the Church-Turing thesis, which they called the *invariance thesis*:

> *Invariance thesis*: Reasonable computational models simulate each other with polynomially bounded overhead in time, and constant factor overhead in space.

and gave a space-agnostic version to it:

> *Weak invariance thesis*: Reasonable computational models simulate each other with polynomially bounded overhead in time.

If the weak invariance thesis holds, then, as far as the λ-calculus is concerned, one should be able to prove that the λ-calculus has a unitary time cost model which is polynomially related to the unitary time cost model of a reasonable model of computation, like Turing machines. More precisely, and since in its most general case the λ-calculus is a non-deterministic model of computation, proving its invariance requires fixing a deterministic reduction sub-relation and proving the invariance result for its unitary time cost model. Of course, it cannot be a randomly-picked evaluation strategy; rather, it should be one that always reaches the normal form if there exists one—which means that it is a *normalizing* evaluation strategy, in particular because of the confluence of λ-calculus. The natural candidate for this is the standard evaluation strategy of the λ-calculus, leftmost-outermost reduction.

2.3.2 The size-explosion problem in the λ-calculus

Recently, Accattoli and Dal Lago gave, in their "(Leftmost-outermost) Beta Reduction is Invariant, Indeed"[AL16], a positive result for the reasonability of the unitary time cost model of leftmost-outermost reduction. Of course, results similar to those in [AL16] had been given before. For example, Blelloch and Greiner's "Parallelism in sequential functional languages" [BG95] is the very first publication on the topic, where they prove the reasonability of a weak and closed evaluation strategy in the λ-calculus performing call-by-value reduction; Sands, Gustavsson and Moran prove a similar result in their "Lambda Calculi and Linear Speedups" [SGM02]. Other examples include Dal Lago and Martini's "Derivational Complexity is an Invariant Cost Model" [DM09], where the invariance of innermost and of outermost reduction in orthogonal term rewriting systems is proven.

The result in [AL16] was a somewhat unexpected result. In fact, the community used to believe that strong evaluation strategies in the λ-calculus did not have a unitary time cost model, especially considering Asperti and Mairson's result that Lèvy's optimal strategy—which performs strong reduction—is not reasonable. Moreover, it is relatively easy to derive counter-examples of invariance for the unitary time cost model for virtually any evaluation strategy of the λ-calculus. In particular in its original formulation, where the substitution process is realized via meta-level substitutions and so there is no sharing of subterms.

A typical such counter-example is the one known as the *size-explosion problem*, which consists of a family of Λ-terms whose normalizing sequence involves substituting subterms that do not contribute in any way to the creation of β-redexes, but whose substitution increases the size of the term at each β-step, thus producing an exponentially-large gap between the size of the initial term and the size of the normal form.

We now turn to present the size-explosion problem as given in [AL16], using the notation introduced therein. We begin by formalizing some of the basic notions involved in its presentation:

- The *size* of a Λ-term t, noted $|t|$, is defined by the following set of equations:

$$\begin{array}{rcl} |x| & \coloneqq & 1 \\ |u\,s| & \coloneqq & |u| + |s| + 1 \\ |\lambda x.u| & \coloneqq & |u| \end{array}$$

- A *variable tree of height* n, noted $x^{@n}$, is given by the following set of equations:

$$\begin{array}{rcl} x^{@0} & \coloneqq & x \\ x^{@(n+1)} & \coloneqq & x^{@n} x^{@n} \end{array}$$

Remark. By a simple induction on n, we obtain that $|x^{@n}| = 2^{n+1} - 1$.

Proposition 2.3.1 (Size-explosion in the λ-calculus)**.**
Let $\{t_n\}_{n\geq 1}$ be inductively defined by the following equations:

$$\begin{array}{rcccl} t_1 & \coloneqq & \lambda x_1.(x_1\,x_1) & = & \lambda x_1.x_1^{@1} \\ t_{n+1} & \coloneqq & \lambda x_{n+1}.(t_n\,(x_{n+1}\,x_{n+1})) & = & \lambda x_{n+1}.x_{n+1}^{@1} \end{array}$$

noting that $|t_n| = 4n - 1$, for every n. Moreover, let $y \in \mathsf{Var}$ and $m \in \mathbb{N}$.
Then, $|t_n\,y^m| = \mathcal{O}(n + 2^m)$ and $t_n\,y^{@m} \to_\beta^n y^{@(n+m)}$, where $|y^{@(n+m)}| = \Omega(2^n\,2^m)$.

Proof. First, recall that $|x^n| = 2^{n+1} - 1$. Then, for every n and m we have that $|y^{@(n+m)}| = 2^{n+m+1} - 1 = 2\,2^n\,2^m - 1 = \Omega(2^n\,2^m)$.

Next, we prove that $|t_n\,y^m| = \mathcal{O}(n + 2^m)$ by induction on n:

- *Base case*: Let $n \coloneqq 1$. Then,

$$|t_1\,y^{@m}| = |t_1| + |y^{@m}| + 1 = (4n - 1) + (2^{m+1} - 1) + 1 = \mathcal{O}(n + 2^m).$$

- *Inductive case*: Let $n > 1$. Then,

$$|t_{n+1}\,y^{@m}| = |t_{n+1}| + |y^m| + 1 = (4(n+1) - 1) + (2^{m+1} - 1) + 1 = \mathcal{O}(n + 2^m).$$

Finally, we prove by induction on n the part concerning the reduction sequence:

- *Base case*: Let $n \coloneqq 1$. Then,

$$t_1\,y^{@m} = (\lambda x_1.x_1\,x_1)\,y^{@m} \to_\beta y^{@m}\,y^{@m} = y^{@(1+m)} = y^{@(n+m)}$$

- *Induction case*: Let $t_n\,y^{@m} \to_\beta^n y^{@(n+m)}$. Then,

$$\begin{array}{rcl} t_{n+1}\,y^{@m} & = & (\lambda x_{n+1}.(t_n\,({x_{n+1}}^{@1})))\,y^{@m} \\ & \to_\beta & t_n\,\left((y^{@m})^{@1}\right) \\ & = & t_n\,(y^{@(m+1)}) \\ & \to_\beta^n & y^{@(n+m+1)} \\ & = & y^{@((n+1)+m)} \end{array}$$

□

That is, Proposition 2.3.1 shows that there exists a family $\{t_j\}_{j\geq 1}$ of Λ-terms such that $t_j\, y^{@m}$ reduces in j $\to_\beta$-steps to the $\to_\beta$-normal form $y^{@(j+m)}$ whose size is *exponentially* larger than size of the initial term $t_j\, y^m$.

This would seem to be a counter-example to the weak invariance thesis, namely because the output term takes an exponential time to be written down, thus seemingly canceling any possibilities for the λ-calculus to be polynomially related to Turing machines. Furthermore, the size-explosion problem is known to be adaptable to virtually every reduction relation in the λ-calculus.

However, there exists a unitary cost model for the λ-calculus which is polynomially related to the one for Turing machines. As shown in [AL16], the number of reduction steps in the useful variant of leftmost-outermost reduction is both polynomially related to the number of β-steps in the λ-calculus and polynomially related to the time cost model for Turing machines. This result may seem counter-intuitive when one considers the size-explosion problem, but it can be decomposed in two simple elements that explain the success in [AL16], namely:

- The leftmost-outermost reduction is defined in [AL16] in the form of an *evaluation strategy in the LSC*, where it is moreover shown to be polynomially related to the λ-calculus.
- The substitution process in the leftmost-outermost strategy is restricted to be useful, and this useful variant is shown to be polynomially related to Turing machines.

Thus, usefulness is the key novelty used in [AL16] to achieve the invariance result for the λ-calculus. But the LSC is not the only formalism where usefulness is shown to be the key element in proving the invariance result for the most general case of the λ-calculus. For instance, in his "The Useful MAM, a Reasonable Implementation of the Strong λ-Calculus"[Acc16] Accattoli provides an abstract machines performing useful substitutions.

Needless to say, usefulness need not be restricted to a particular notion of reduction relation, as it may be applied to other settings. An interesting example of such an application is Yoshida's [Yos93], where she defines a weak λ-calculus with explicit substitutions morally implementing useful sharing.

Following this same vein—that is, extending usefulness to other reduction relations—the goal of the Useful Open CbNeed evaluation strategy presented in Chapter 8 (Useful Open CbNeed) is to study usefulness in a call-by-need setting. Additionally, in Chapter 9 (Multi types for Useful Open CbNeed) we study a type-theoretical interpretation of usefulness in this same call-by-need setting.

2.4 Multi types

In the early 90's, the scientific community became interested in non-idempotent variants to intersection types. One such case is Philippa Gardner's [Gar94] (Discovering Needed Reductions Using Type Theory), where she adapts intersection types to make them incorporate explicit information about resources, thus allowing her to identify all needed redexes. This work by Gardner is considered the first envisioning of what was latter known as non-idempotent intersection types. We shall call them "*multi types*" instead, to avoid the long name.

Since the publication of [Gar94], numerous other results involving multi types have emerged, including (but not limited to)

- Several strong normalization[BR13; BG13] and head and weak-normalization[Car07] results of the λ-calculus.

- The synthesis of a resource-aware semantics for the λ-calculus[BCL99].
- The proof of an exact correspondence between the size of type derivations—in a suitable non-idempotent intersection type system—and the execution time in head and weak-normalizing Λ-terms [Car07]. This work in particular has had a big influence in ours.

For a more general survey on multi types, we suggest Bucciarelli, Kesner and Ventura's [BKV17] (Non-idempotent intersection types for the Lambda-Calculus) to the reader. As thoroughly explained in [BKV17], there exists a technical improvement over idempotent intersection types when proving normalization results via the non-idempotent variants: while the idempotent ones traditionally make use of a *reducibility* technique [Kri93], the quantitative information present in multi type derivations is enough to prove the adequacy of the resulting denotational semantics via a simple *combinatorial* argument, as pioneered by [Ehr12]. Thus, although still somewhat technically involved, all the proofs of adequacy of our non-idempotent intersection type systems make use of a combinatorial argument.

Multi types and multiset notation. In this work, we shall use multi type systems as a considerably flexible tool that allows us to characterize normalization in a given reduction relation. Consequently, we shall be able to infer an *interpretation* specific to each reduction relation.

Multi types enjoy such a high degree of flexibility, that even the very notion of types depends on the corresponding reduction relation that the multi type system is meant to characterize normalization of. These differences in the definitions of multi types is thoroughly explained in Chapter 5 (Multi types for CbN, CbV and CbNeed).

In spite of that, let us momentarily resort to the *CbNeed* evaluation strategy—properly introduced in Chapter 4 (CbN, CbV and CbNeed)—for the sake of presenting a definition of multi types that may help the unacquainted reader get a first grasp of multi types. Thus, CbNeed multi types are defined mutually inductively as follows

$$\begin{array}{rl} \textsc{CbNeed linear types} & L, L' ::= \mathsf{norm} \mid M \multimap N \\ \textsc{CbNeed multi types} & M, N ::= [L_i]_{i \in I} \text{ (with I finite)} \end{array}$$

That is, CbNeed linear types are either the constant base type norm or a *linear arrow* type of the form $M \multimap N$, where M and N are multi types. Moreover CbNeed multi types are finite collections of CbNeed linear types[7].

Note that we may safely represent multi types using the multi set notation, and hence the name *multi* types. Then, we may adapt the multi set union operator $\uplus$ to multi types as expected. For example, $[L_1, L_2] \uplus [L_1] = [L_1, L_2, L_1]$.

Finally, the empty multi type $[]$ is noted with the special symbol $\mathbf{0}$, and plays a remarkably important role in many of our case studies. This is analyzed, for example, in Chapter 5 (Multi types for CbN, CbV and CbNeed).

[7]Keep in mind that in Chapter 5 we shall see that CbN and CbV have specific definitions for their notions of linear and multi types. Nevertheless, the common point among all of them is that multi types are always finite collections of (the appropriate notion of) linear types, and that linear types are either a constant base type, or are built using the linear arrow $\multimap$ type constructor and a pair of types.

Type contexts and type judgements. As usual in Gentzen-style type systems, our multi type systems implement the notion of type contexts and of type judgements. Let us first define the former, again in terms of the CbNeed multi type system:

Definition 7 (Type contexts)**.**

A *type context* Γ for the CbNeed multi type system is a total function, from Var to the class of CbNeed multi types, such that only finitely many variables are not mapped to the empty multi type $\mathbf{0}$. The *domain* of type context Γ is then defined as $\mathsf{dom}(\Gamma) \coloneqq \{x \mid \Gamma(x) \neq \mathbf{0}\}$, and we write $\Gamma = \emptyset$ when $\mathsf{dom}(\Gamma) = \emptyset$.

The multiset union operator $\uplus$ is extended to type contexts point-wise and noted $\biguplus$. That is,

$$\left(\Gamma \biguplus \Pi\right)(x) \coloneqq \Gamma(x) \uplus \Pi(x)$$

for each $x \in \mathsf{Var}$. This can then be further extended to a finite family of type contexts as expected, so that $\biguplus_{i \in J} \Gamma_i$ denotes a finite union of type contexts, with the convention that $\biguplus_{i \in J} \Gamma_i = \emptyset$ when $J = \emptyset$.

Moreover, a type context Γ is written as $x_1 : M_1; \ldots; x_n : M_n$, with $n \in \mathbb{N}$, if $\mathsf{dom}(\Gamma) = \{x_1, \ldots, x_n\}$ and $\Gamma(x_i) = M_i$ for all $1 \leq i \leq n$.

Finally, given type contexts Γ and Π such that $\mathsf{dom}(\Gamma) \cap \mathsf{dom}(\Pi) = \emptyset$, we define type context $\Gamma; \Pi$ as follows

$$(\Gamma; \Pi)(x) \coloneqq \begin{cases} \Gamma(x) & x \in \mathsf{dom}(\Gamma) \\ \Pi(x) & x \in \mathsf{dom}(\Pi) \\ \mathbf{0} & \text{otherwise} \end{cases}$$

As a minor remark, note that we do not need to resort to $\bot$ to express that a type context Γ is undefined on a particular variable x. Instead, we may write have that $\Gamma(x) = \mathbf{0}$; that is, $\mathbf{0}$ is the uniform way in which we represent that $\Gamma(x) = \bot$.

Last, let us define the notion of type judgement, again for the CbNeed type system:

Definition 8 (Type judgements in the CbNeed type system)**.**

Given type context Γ and multi type or linear type T, the syntactic object $\Gamma \vdash^{(i_1,\ldots,i_n)} t : T$ is called a *type judgement*, where Γ is the type context of the type judgement, T is the *type*, and $(i_1, ..., i_n)$ is a vector of natural numbers called the *indices*.

The quantitative information contained in type derivations shall be represented by the indices of the final type judgement in a type derivation. Since the quantitative information for each of the case studies may vary, the cardinality and interpretation of indices shall only be considered when presenting each of the multi type systems.

We do not give further details or insights about multi types here. Like we did with respect to CbNeed, we believe that this is better done when considering a concrete case. Indeed, many of the details of the multi type systems unavoidably depend on the operational features of the reduction relation that they characterize.

Chapter 3

A bird's eye view

We have concluded the preliminary studies and covered the required base concepts to be able to give a general view of what this work is about. This chapter intends to summarize the results we obtained in the preparation of this thesis, hopefully allowing the reader to comprehend the motivations and developments of the upcoming chapters.

We refrain from going into deep technical details here. Instead, insights, further arguments on the motivation for certain technical choices, examples and explanations shall be separately presented in due time—that is, in the appropriate chapter where each of these technicalities are relevant.

In any case, this chapter serves as a common ground where we discuss about the differences and similarities between the case studies presented here. We recommend the reader to revisit it after reading the other chapters, to get a more panoramic perspective on the overall subject.

Introduction

Using type derivations for quantitative analyses. Throughout this work, we present several evaluation strategies, each one of them associated with a different multi type system. We also use these multi type systems to provide quantitative information about the normalization process in the evaluation strategy.

But, in which way exactly do multi types provide the tools for quantitative analyses of typed expressions? The answer may be stated in general terms as follows: given a multi type system $\mathcal{X}$, associated to an evaluation strategy $\to_{\mathcal{X}}$, and a type derivation Φ for an expression e, then the indices of Φ contain quantitative information regarding a $\to_{\mathcal{X}}$-normalizing reduction sequence starting at e. Conversely, given a $\to_{\mathcal{X}}$-normalizing reduction sequence starting at expression e, we are able to derive a type derivation Φ in $\mathcal{X}$ for e such that the indices of Φ contain precise quantitative information regarding the $\to_{\mathcal{X}}$-normalizing reduction sequence starting at expression e.

A natural follow-up questions arises: what *kind* of quantitative information do the indices of type derivations contain? Unfortunately, this cannot be answered here in full detail, as we lack the technical details of each of the multi type systems. Nevertheless, we can say that the nature of the quantitative information contained in indices falls into one of the two following categories:

- The number of reduction steps from the typed expression to its normal form in the corresponding evaluation strategy. This in turn is split in the different kinds of reduction steps that said evaluation strategy is composed of. Thus, every evaluation strategy performing linear substitutions—see Sect. 2.2 (The Linear Substitution Calculus)—has a multiplicative and an exponential index, respectively counting multiplicative and exponential reduction steps.

- The size of the normal form in the corresponding evaluation strategy.

Furthermore, there exists a direct correspondence between indices appearing in a final type judgement of a type derivation, and the number of applications of the different typing rules in said type derivation. More specifically, each index appearing at the end of a type derivation actually counts the number of applications of a certain typing rule in said type derivation. Hence, indices are a concise representation of the quantitative information in a type derivation.

Lax or tight upper bounds. Generally speaking, however, not every index provides *exact* quantitative information. In fact, the general rule is that indices provide *upper bounds* to the different measures of interest.

In order to obtain *exact* quantitative information, we shall define a class of type derivations, specific to each one of the multi type systems, providing exact upper bounds, or simply *measures*. These classes of type derivations providing measures on the normalization process shall all be defined in terms of the final type judgement of type derivations.

3.1 Properties of evaluation strategies

The evaluation strategies that we cover in this work are all based on some kind of explicit-substitutions formalism, either on the LSC or on some derivative. There are some common features among them that we would like to sketch now, where $\to_{\mathcal{X}}$ represents any given evaluation strategy:

- $\to_{\mathcal{X}}$ is given by a combination of the appropriate notions of evaluation contexts and of $\to_{\mathcal{X}}$-root steps.
- $\to_{\mathcal{X}}$-normal forms are characterized by syntactically-driven predicates, which are invaluable in the connection between the $\to_{\mathcal{X}}$ and its multi type system.
- We prove that $\to_{\mathcal{X}}$ enjoys a relaxed form of determinism, either by showing that it is deterministic or that it satisfies the diamond property.

3.2 Properties of multi type systems.

Our study of multi types mimics Accattoli, Graham-Lengrand and Kesner's "Tight Typings and Split Bounds"[AGK20]. In particular, because all our multi type systems characterize normalization in a certain evaluation strategy, because the indices in type derivations provide upper bounds on the normalization process, and because for each of the multi type systems we characterize a class of type derivations whose indices give exact quantitative information.

Tight constants. Some of our multi type systems adapt moreover a technique from [AGK20] that we call *tightness technique*.

This technique has very recently been rephrased in Kesner and Vial's "Consuming and persistent types for Classical Logic" [KV20], where persistent types morally represent the category of our *tight* types. The intuition is that consuming types are used to type constructors involved in redexes and consumed when said redexes are fired, while persistent types are used to type those subterms that are never consumed and shall end up in the normal form of the expression. Thus, consuming types

are represented by an arrow type $M \to N$, while persistent types are represented by a set of linear constant types—in our case, the *tight* constants.

This distinction between consuming and persistent types has later been applied to calculi extending the λ-calculus. An example of this is Alves, Kesner and Ventura's [AKV19], where a type system with consistent and persistent types is derived for head reduction with pattern matching, proving this technique to be also relevant in variants of the λ-calculus with added features.

When the goal is to obtain a class of type derivations giving exact bounds, the tight types technique is known to be *much simpler* than those which do not use constants at all. For instance, the analysis of the leftmost-outermost reduction in [AGK20] first considers the design of a multi type system implementing the tight types technique, and only then extend it to isolate a class of type derivations giving exact bounds *without* using the tight types technique—namely, by using the *shrinking types technique*, which shall be thoroughly explained in due time.

In this text, we derive two multi type systems based on the tight types technique, namely the Open CbNeed and for Useful Open CbNeed ones. The justification for this is two-fold:

- Implementing usefulness in the call-by-need setting is significantly complex. Consequently, deriving a multi type system for it is also very complex. But the type-theoretical analyses in Chapter 9 (Multi types for Useful Open CbNeed) are considerably simplified by the use of tight constants. Therefore, we claim that our multi type-theoretical presentation of Useful Open CbNeed should be considered as a highly complex, but absolutely necessary, intermediate step towards a more general type system[1].
- The original motivation for the development of the Useful Open CbNeed system came from [AGK20], as we realized that tight constants were a natural candidate for establishing the difference, in terms of characterization, between the useful and non-useful variants of a given evaluation strategy—like the Useful Open CbNeed and Open CbNeed ones, for example.

Semantical properties. Additionally, we shall infer *semantics* specific to each one of the multi type systems. Roughly speaking, given a multi type system $\mathcal{X}$, the semantics of an expression e given by $\mathcal{X}$ is noted $[\![\mathsf{e}]\!]_{\mathcal{X}}$, and shall be given by the following equation:

$$[\![\mathsf{e}]\!]_{\mathcal{X}} := \{(\Gamma, M) : \exists\, \Phi \triangleright_{\mathcal{X}} \Gamma \vdash \mathsf{e} : M\}$$

We shall prove the *invariance* and *adequacy* of the semantics provided by the multi type systems in this work[2]. Moreover, and although we shall not explicitly study it, some of our type systems also satisfy *compositionality*. Let us briefly discuss these three concepts, taking as reference a generic evaluation strategy $\to_{\mathcal{X}}$ and its corresponding multi type system $\mathcal{X}$:

- *Invariance*: Typability in $\mathcal{X}$ is invariant by $\to_{\mathcal{X}}$-reduction and by $\to_{\mathcal{X}}$-expansion.
  This is proven, respectively, by showing that the *Subject Reduction* and *Subject Expansion* properties hold for $\mathcal{X}$. In fact, we shall give *quantitative* versions of Subject Reduction and Subject Expansion, showing how the quantitative information contained in indices changes by reduction or expansion.
- *Adequacy*: This property states that typability in $\mathcal{X}$ equals $\to_{\mathcal{X}}$-normalizability. Formally, $[\![\mathsf{e}]\!]_{\mathcal{X}} \neq \emptyset$ *if and only if* e is $\to_{\mathcal{X}}$-normalizable.

[1] This claim is further reinforced by the fact that the shrinking types technique might need to be adapted to Useful Open CbNeed.

[2] Unfortunately, some of our systems shall satisfy these properties *provided that we restrict the semantics* to a certain class of type derivations. Each case is different, and shall be covered in due time.

One of the directions of the proof is given by the Correctness theorem for $\mathcal{X}$, showing that typability in $\mathcal{X}$ implies $\to_{\mathcal{X}}$-normalizability. In fact, we shall refine the Correctness theorems with *quantitative* information, showing specifically how indices in type derivations relate to the quantities of $\to_{\mathcal{X}}$-reduction sequences.
Similarly, the other direction of the proof is given by the Completeness theorem for $\mathcal{X}$, showing that $\to_{\mathcal{X}}$-normalizability implies typability in $\mathcal{X}$. Again, we shall refine the Completeness theorems with quantitative information, showing how we can derive a type derivation in $\mathcal{X}$ whose indices measure precisely the quantities of the $\to_{\mathcal{X}}$-reduction sequence.

- *Compositionality*: This third property is usually considered when studying a given semantics. We do not explicitly do so in this work, as the proofs of invariance and adequacy are already complex enough. Nonetheless, let us give some remarks about it.
  Although compositionality can have many definitions, the basic way in which one would like to compose programs is by *modus ponens*. That is, given a pair of type derivations whose final type judgements are $\Gamma \vdash t : N \multimap M$ and $\Pi \vdash u : N$, we would like to obtain type derivation whose final type judgement is $\Gamma \uplus \Pi \vdash t\, u : M$.
  Moreover, note that since $[\![t]\!]_{\mathcal{X}}$ and $[\![u]\!]_{\mathcal{X}}$ are sets of type derivations, then this kind of compositionality can be easily extended to the interpretations of t and u. In our multi type systems where type derivations provide upper bounds, this extension to interpretations becomes particularly interesting when tu is a redex in $\to_{\mathcal{X}}$, because the compositionality of the interpretation then allows us to reason *quantitatively* about tu by means of the interpretations of $[\![t]\!]_{\mathcal{X}}$ and of $[\![u]\!]_{\mathcal{X}}$, recalling that the type derivations in $[\![t]\!]_{\mathcal{X}}$ (resp. $[\![u]\!]_{\mathcal{X}}$) contain upper bounds on the $\to_{\mathcal{X}}$-normalization process for t (resp. u).
  Regarding the systems in this thesis, the semantics inferred from the multi type systems in Chapter 5 (Multi types for CbN, CbV and CbNeed) satisfy compositionality.
  The case of the semantics inferred from the multi type system in Chapter 11 (Multi types for Strong CbV) is slightly different, in that invariance, adequacy and, in particular, compositionality are only satisfied by a subset of the type derivations.
  On the contrary, compositionality of the semantics inferred from the Open CbNeed and Useful Open CbNeed multi type systems is incompatible with the way in which we prove their adequacy and invariance. In fact, the interpretations $[\![\mathsf{e}]\!]_{\mathcal{X}}$, with $\mathcal{X} \in \{$Open CbNeed, Useful Open CbNeed$\}$, shall be restricted to final type judgements *composed only of tight constants*. That is, we shall exclude, from the semantics given by the Open CbNeed and Useful Open CbNeed systems, those type derivations whose type contexts assign variables to multi types containing linear types of the form $N \multimap M$, as well as those whose right-hand side types are of the form $N \multimap M$.
  Although this clearly affects compositionality of these semantics, the reason for giving these restricted semantics for the Open CbNeed and Useful Open CbNeed systems is that the Subject Reduction and Subject Expansion properties—used to prove invariance—only hold under these restricted conditions. In addition, since the Correctness and Completeness theorems—used to prove adequacy—depend, respectively, on the Subject Reduction and Subject Expansion properties, then they also only hold under the same conditions.

Proof technique. Let us now turn to a more concrete analysis on the proof technique used to prove the Correctness and Completeness theorems. Once again, $\mathcal{X}$ represents any given multi type system in this work and $\to_{\mathcal{X}}$ represents the corresponding evaluation strategy associated with $\mathcal{X}$.

Moreover, we use the expression "*minimal*" to represent the specific class of type derivations in $\mathcal{X}$ which should contain minimal quantitative information[3]

Let us begin by analyzing the proof technique used in the Correctness theorems:

- First, we prove that the indices in minimal type derivations for $\to_{\mathcal{X}}$-normal forms are *exact*. The propositions concerning this property are called "*Typing properties of $\to_{\mathcal{X}}$-normal forms*".
- Recall that $\mathcal{X}$ is an evaluation strategy based on the LSC formalism, and as such its definition contains an exponential part implementing the substitution of variables. For this reason, we then need to prove that $\mathcal{X}$ mimics $\to_{\mathcal{X}}$-exponential steps at the level of type derivations, and that this produces the expected quantitative information. These lemmas are called "*(Linear) Substitution for $\to_{\mathcal{X}}$*"[4].
- Next, we prove the quantitative version of the Subject Reduction property—called "*(Shrinking) Quantitative Subject Reduction for $\to_{\mathcal{X}}$*"[5].
  The proof is split between multiplicative and exponential $\to_{\mathcal{X}}$-steps, where the exponential base case requires the "(Linear) Substitution for $\to_{\mathcal{X}}$" lemma. Typically, Subject Reduction holds if for every time that $\mathsf{e} \to_{\mathcal{X}} \mathsf{e}'$ and there exists type derivation Φ whose final type judgement is $\Gamma \vdash^{(i_1,...,i_n)} \mathsf{e} : T$, then there exists type derivation Ψ whose final type judgement is $\Gamma \vdash^{(i'_1,...,i'_n)} \mathsf{e}' : T$. While the exact relation between $(i_1, ..., i_n)$ and $(i'_1, ..., i'_n)$ strongly depends on the operational specifics, the general property is that there is a certain index in $(i'_1, ..., i'_n)$ that is strictly smaller than the corresponding one in $(i_1, ..., i_n)$.
- Finally, we prove the Correctness theorem. This is done by induction on the sum of the indices of type derivation Φ, where the base case is given by the "Typing properties of $\to_{\mathcal{X}}$-normal forms" proposition, and the inductive case is given by the "(Shrinking) Quantitative Subject Reduction for $\to_{\mathcal{X}}$" proposition.
  We also prove that if the type derivation is minimal, then the indices in Φ provide exact quantitative information.

Let us analyze now the proof technique used in the Completeness theorems. Keep in mind that although it should be regarded as the dual of the Correctness theorem, there exist subtle differences between one another.

- We begin by showing that every $\to_{\mathcal{X}}$-normal form has a minimal type derivation, whose indices contain exact quantitative information. These propositions are called "*(Tight/Shrinking) typability of $\to_{\mathcal{X}}$-normal forms*".
- Next, we turn to prove the dual of the "(Linear) Substitution for $\to_{\mathcal{X}}$" lemma, namely the "*(Linear) Removal for $\to_{\mathcal{X}}$*" lemma.
- The quantitative version of the Subject Expansion property, called the "*(Shrinking) Quantitative Subject Expansion for $\to_{\mathcal{X}}$*" propositions, is the dual of the "(Shrinking) Quantitative Subject Reduction for $\to_{\mathcal{X}}$" proposition. Their proofs follow a very similar but dual structure. In particular, the exponential base case requires the "(Linear) Removal for $\to_{\mathcal{X}}$" lemma.
  Typically, Subject Expansion holds if for every time that $\mathsf{e} \to_{\mathcal{X}} \mathsf{e}'$ and there exists type derivation Ψ whose final type judgement is $\Gamma \vdash^{(i'_1,...,i'_n)} \mathsf{e}' : T$, then there exists type derivation

[3]By "*minimal*" type derivations we mean those whose indices provide *minimal* upper bounds with respect to a certain operational quantity. As a matter of fact, the minimality is such that the indices actually *match* the corresponding operational quantity. Thus, we may also talk of type derivations providing *exact quantitative information* or even *measurements*.

[4]The only case study with a non-linear Substitution lemma is covered in Chapter 11 (Multi types for Strong CbV).

[5]"*Shrinking*" makes reference to a typing technique which is only implemented in Chapter 11 (Multi types for Strong CbV).

Φ whose final type judgement is $\Gamma \vdash^{(i_1,...,i_n)} \mathsf{e} : T$. Quantitatively speaking, there is a certain index in $(i_1, ..., i_n)$ that is strictly larger than the corresponding one in $(i'_1, ..., i'_n)$.

- Finally, the Completeness theorem. It states that if $d : \mathsf{e} \to^*_{\mathcal{X}} \mathsf{e}'$ where e' is a $\to_{\mathcal{X}}$-normal form, then there exists a minimal type derivation Φ for e. Moreover, the indices in the final type judgement of Φ provide exact quantitative information concerning d. This is done by induction on the the length of d, where the base case is given by the '*(Tight/Shrinking) typability of $\to_{\mathcal{X}}$-normal forms*" proposition, and the inductive case is obtained expanding to e by the "(Shrinking) Quantitative Subject Expansion for $\to_{\mathcal{X}}$" proposition.

This concludes the analysis of the general proof technique. Let us now turn to give a short description of each of the case studies in this thesis.

3.3 Case study: CbNeed

We begin by covering a well-known evaluation strategy, namely the weak and closed version of the "*call-by-need*" evaluation strategy—or "*CbNeed*" for short. For the original formulation of weak and closed CbNeed, see [AF97; MOW98].

Unlike in the original formulations, however, our choice has been to formalize CbNeed as an evaluation strategy in the LSC, as first proposed in [ABM14]. There are several arguments in favor of this choice, one of them being the close connections the LSC bears to the abstract machines world—as analyzed in depth in [ABM14]. But most importantly, we chose the LSC because reduction relations defined in this setting need to present multiplicative and exponential reduction steps in a separate way. It is precisely this feature of LSC that allows us to disentangle the notions of *duplication* and *erasure*.

We give a proper analysis of these two features in Chapter 4 (CbN, CbV and CbNeed). Nonetheless, we would like to mention here that the weak and closed variant of CbN and CbV—when given in the LSC setting—are dual with respect to duplication and erasure. Efficiency wise, both evaluation strategies have drawbacks, either in the way they handle erasure or in the way they handle duplication. Moreover, we shall see that CbNeed takes the best of both worlds. We present these facts in the form of the following slogans:

$$\begin{gathered}\text{CbN} = \text{silly duplication} + \text{wise erasure},\\ \text{CbV} = \text{wise duplication} + \text{silly erasure, and}\\ \text{CbNeed} = \text{wise duplication} + \text{wise erasure}\end{gathered}$$

With these insights in mind, we face our first major goal in this work: namely, deriving a multi type system characterizing CbNeed-normalization, containing moreover a subclass of type derivations that provide exact measures of the CbNeed-normalization process.

Such a goal requires first covering the operational account of all three evaluation strategies—that is, of CbN, CbV and CbNeed. This is done in Chapter 4 (CbN, CbV and CbNeed) by formalizing them as evaluation strategies in the LSC. It is precisely the separate treatment of multiplicative and exponential reduction steps in a LSC setting that allows us to define CbNeed as the combination of CbV duplication and CbN erasure, namely by picking the right notions of multiplicative and exponential root steps, together with the right notion of CbNeed evaluation context.

Having established the relations CbNeed has with CbN and CbV from the operational standpoint, we present in Chapter 5 (Multi types for CbN, CbV and CbNeed) a specific multi type system for each of these three evaluation strategies. On the one hand, the CbN multi type system is

essentially a reformulation into the LSC setting of de Carvalho's system $\mathcal{R}$[Car18]. In turn, system $\mathcal{R}$ is itself a type-based presentation of the relational model of the CbN λ-calculus, which is induced by the relational model of Linear Logic via the CbN translation of the λ-calculus into Linear Logic. On the other hand, the CbV multi type system is essentially a reformulation into the LSC setting of Ehrhard's CbV multi type system[Ehr12]. Similarly, Ehrhard's CbV multi type system is itself a type-based presentation of the relational model of the CbV λ-calculus, which is induced by the relational model of Linear Logic via the CbV translation of the λ-calculus into Linear Logic. Thus, we obtain the CbNeed by carefully combining the right typing rules from the CbN and CbV type systems, chosen in such a way as to match the operational presentation given for CbNeed, in particular in its management of duplication and erasure.

Next, we give separate semantics for each of CbN, CbV and CbNeed. The interpretations are given directly in terms of type derivations.

Finally, we provide a proof that "*CbNeed is as efficient as CbV*, namely by showing that the normalization bounds given by the CbV multi type system also apply to CbNeed. This is all achieved via a type-theoretical approach, constituting a novelty of our work[6]. From the perspective of duplication and erasure, the slogan we give to this result is that "CbNeed duplicates as wisely as CbV".

3.4 Case study: Open CbNeed

When modelling programming languages, it is natural to take an evaluation strategy that performs weak reduction on closed terms as the underlying model of computation. For this purpose, the weak and closed evaluation strategies studied in Chapter 4 (CbN, CbV and CbNeed) suffice.

But in some technological applications, the weak and closed restrictions are not enough as underlying model of computation. Take, for instance, proof assistants based on dependent types—like Coq or Agda—whose modeling requires a strong evaluation strategy acting on (possibly) open terms. Unfortunately, it is well-known that the operational semantics of most evaluation strategies gets considerably more complex when going from weak reduction on closed terms to strong reduction on (possibly) open terms.

In their [GL02], Grégoire and Leroy propose implementing strong reduction by iterating a kind of *intermediate* reduction under λs. Said intermediate reduction is placed somewhere between weak reduction on closed terms and strong reduction on (possibly) open terms. Many technicalities need to be considered to give the right definition of such an intermediate reduction relation; details aside, what is important to remark here is that this intermediate reduction may act on open terms, while remaining weak.

With a similar goal in mind, in Chapter 6 (Open CbNeed), we study an evaluation strategy placed between the weak and closed CbNeed , and a strong evaluation strategy implementing call-by-need that acts on (possibly) open terms. Following the terminology used in "Open Call-by-Value"[AG16][7] we call this evaluation strategy "*Open CbNeed*".

The reason why Open CbNeed is relevant is that it presents most of the subtleties appearing in the strong case, while remaining simpler than the latter. We shall give further, more concrete

[6] A dual result to this is given in [Kes16], where CbNeed is shown to be termination-equivalent to CbN, also via the use of a multi type system. From the perspective of duplication and erasure, this result can be read as "CbNeed erases as wisely as CbN".

[7] Where a similar evaluation strategy implementing call-by-value on open terms is studied.

reasons as to why Open CbNeed is valuable in Sect. 3.5 (Case study: Useful Open CbNeed) below, specifically concerning the reasonability of the strong variant of call-by-need.

As far as relations to the literature is concerned, in Balabonski, Barenbaum, Bonelli and Kesner present in their "Foundations of Strong Call by Need' [Bal+17] a strong evaluation strategy implementing call-by-need. As it turns out, our Open CbNeed evaluation strategy takes considerable inspiration from the *Strong Call by Need* strategy appearing therein, and might even be shown to represent a sub-relation of Strong Call by Need, one where reduction does not enter λ-abstractions. We do not prove such result here. Instead, we isolate the weak reduction sub-relation in [Bal+17] and adapt it to our purposes, obtaining our Open CbNeed evaluation strategy.

To finish, let us see some of the proofs and concepts that are required for the open case:

- *Determinism*: Being a variant of the Strong Call by Need strategy in [Bal+17] and having no other point of reference in the literature to the best of our knowledge, the determinism of Open CbNeed needs proving.
- *Characterizing normal forms*: While the set of CbNeed-normal forms is (absurdly) easy to characterize, the one of Open CbNeed-normal forms requires a delicate treatment.
- *Needed variables*: The theory of Open CbNeed is highly centered on the notion of *needed variables*. Roughly put, the needed variables of an expression are the set of its free variables restricted to those that appear out of all λ-abstractions. For example, x is a needed variable of $x\lambda y.z$, while z is a free variable but not a needed one.
  We believe that our Open CbNeed theory would not have been so easily proven without the isolation of the notion of needed variables. Just to give an example of the centrality of needed variables in the type theory of Open CbNeed, we show in the Correctness and Completeness theorems for Open CbNeed in Chapter 7 (Multi types for Open CbNeed) that the domain of the type context is exactly the set of needed variables of the typed expression. This was an unexpected result, and it showed us that the approach we followed was consistent all along.
- *Size of normal forms*: Type judgements in the Open CbNeed multi type system contains a new index, which is not present in the CbNeed multi type system. This index provides an upper bound to the size of Open CbNeed-normal forms—or an exact measure of the normal form, if the type derivation is minimal. This index is unnecessary in the (weak and closed) CbNeed case because all CbNeed-normal forms has morally the same size.

3.5 Case study: Useful Open CbNeed

In Sect. 2.3 (Usefulness), we stressed the importance of refraining from performing *useless* substitutions—that is, exponential steps that do not contribute to a future multiplicative step down the reduction sequence—if one is to attain reasonability of the proposed time cost model. Said differently, and as far as the results in the literature are concerned, the size-explosion problem—which may be stated in virtually every calculus—cannot be overcome unless substitutions for variables are only triggered in a *smart* way: namely, by only performing *useful* substitutions.

In Chapter 8 (Useful Open CbNeed), we explore a useful variant of Open CbNeed. Let us give a general explanation of the importance of and roles played by the call-by-need variants we have presented so far, noting in particular that they are increasingly demanding in terms of techniques for reasonable implementations:

- *Weak and Closed*: This variant of call-by-need—which we call CbNeed—only requires micro-step evaluation (that is, performing linear substitutions) for it to be polynomially related to Turing machines. Hence, it is enough to use the LSC definition of CbNeed that we provide in

Chapter 4 (CbN, CbV and CbNeed). Since this property is required in each one of the cases below, let us assume that they are all implemented as evaluation strategy in the LSC and hence have a multiplicative and exponential part.

- *Weak and open*: In order to prove that this variant of call-by-need—called Open CbNeed—is polynomially related to Turing machines, we also need to apply one of the two useful optimizations required in the strong case. Namely, Open CbNeed should not substitute Λ-terms in normal form if they are not a λ-abstraction. These kind of normal forms do not exist in the weak and closed case, but are an intrinsic characteristic of the open setting. They are called in the literature as *neutral* or *inert* terms—we chose the latter for our work—and their substitution is easily shown to not contribute in the creation of new multiplicative steps.
- *Weak, open and useful*: In order to prove that the strong and open variant of call-by-need is invariant, we need to further restrict Open CbNeed by implementing the second useful optimization, which consists in avoiding the substitution of λ-abstractions that do not contribute to the creation of new multiplicative steps. This is the evaluation strategy we present in Chapter 8 (Useful Open CbNeed), where knowing which variable occurrences should be substituted and which should not requires the notions of *applied* and *unapplied* occurrences or *positions*.
- *Strong and open*: Although it has not yet been proven in the literature, we believe that the invariance result for a call-by-need evaluation strategy may be obtained by taking the Useful Open CbNeed as basis and, once a Useful Open CbNeed-normal form has been reached, iterating the same evaluation strategy under λ-abstractions.
  Note that, regarding reasonability of strong call-by-need, we may then say that isolating a fully-useful, weak and open version of call-by-need—namely, our Useful Open CbNeed evaluation strategy—is likely the hardest part in the road down to a reasonable and strong variant for call-by-need. We believe this would be a remarkable achievement in the theory of the λ-calculus.

Adapting the base formalism. As we just hinted, implementing the Useful Open CbNeed evaluation strategy involves defining applied and unapplied positions. For some technical reasons that we shall only consider in Chapter 8 (Useful Open CbNeed), this is more easily achievable in a variant of the LSC that we call the *split LSC*. Since Open CbNeed serves as basis for Useful Open CbNeed, both of them shall be formalized as evaluation strategies in the split LSC.

Refining needed variables. We saw in Sect. 3.4 (Case study: Open CbNeed) that the notion of needed variables is of remarkable importance in the operational and type-theoretical studies of Open CbNeed. Switching to the Useful Open CbNeed evaluation strategy implies an even deeper understanding of the needed variables of an expression. In this sense, we shall refine the notion of needed variables, discriminating between needed variable appearing in applicative positions, which we call *applied* variables, from the needed variables which appear in non-applicative positions, which we call *unapplied* variables.

These two new notions of applied and unapplied variables provide us with the right tools needed to prove the operational and type-theoretical properties of Useful Open CbNeed, much like the notion of needed variables was the key element in the theory of Open CbNeed. Therefore, we would like to stress here the fact that the study of Open CbNeed and Useful Open CbNeed are done in a remarkably consistent way, and switching from the former to the latter requires refining the central concept of the theory, namely switching from needed to applied and unapplied variables,

and analyzing the consequences both from the operational and type-theoretical perspectives.

3.6 Case study: Strong CbV

In Accattoli, Graham-Lengrand and Kesner's 'Tight Typings and Split Bounds'[AGK20], the authors present a multi type system providing exact measures on the normalization process of the standard evaluation strategy in the λ-calculus; namely, leftmost-outermost reduction.

It is well-known that leftmost-outermost reduction implements substitutions in a call-by-name fashion; that is, it imposes no restrictions on the arguments of β-redexes. A consequence of the lack of restrictions on arguments in the call-by-name setting is that turning from weak reduction on closed terms to strong reduction on open terms is harmless.

On the contrary, doing turning to strong reduction in a call-by-value setting is delicate. While some fundamental properties such as confluence and standardization still hold—as shown by Plotkin in his [Plo75]—others break as soon as one considers open terms—these properties are typically of a semantical nature.

In Chapter 10 (Strong CbV), we define strong reduction in a call-by-value setting, which we call "*Strong CbV*". For such purpose, we adopt the "*Value Substitution Calculus*"[8], and endow it with the Strong CbV evaluation strategy, satisfying the diamond property. Although not present here, our study of Strong CbV is actually complementary to the work of co-authors—namely, Beniamino Accattoli, Claudio Sacerdoti Coen and Andrea Condolucci—on a bilinear[9] abstract machine implementing $\to_{\mathsf{s}}$. These results for Strong CbV, ranging from operational semantics, to (multi) type-theoretical interpretations and abstract machines, are all connected at the quantitative level, and have not yet been published.

Then, we give a multi type system characterizing Strong CbV-normalization in Chapter 11 (Multi types for Strong CbV). As usual, we characterize a class of type derivations that provide exact measures with respect to the corresponding Strong CbV -normalization sequence. Remarkably, we do so by using a standard typing technique in the literature, expressed in terms of what are known as "*shrinking types*".

Finally, we use the Strong CbV multi type system to show that the Strong CbV evaluation strategy is normalizing for the VSC.

3.7 Design principles for multi type systems

The understanding of Useful Open CbNeed from the operational and type-theoretical perspectives is, in our opinion, the major contribution of this work. Furthermore, we believe that there exists an additional main contribution in this work, namely the collection of reasonings by which we have obtained the multi type systems. In particular, considering their capability to provide exact normalization measures for several evaluation strategies having varied operational features.

Of course, proving the desired properties for each of these multi type systems requires correlating operational and type-theoretical features in a highly technical way. Nonetheless, we have followed some general design principles throughout this work that we would like to remark here. As a whole, they form a consistent approach to designing multi type systems providing exact measures.

[8] This calculus has a clean Linear Logic background and was first presented in Accattoli and Paolini's [AP12]

[9] See *e.g.* [ABM14] for a definition of bilinearity.

As starting point, we would like to mention the fact that our multi type systems have a significant control over the use of the weakening and contraction structural rules, much like what happens in Linear Logic[Gir87]. More concretely, in our multi type systems,

- The *weakening* rule is disallowed altogether in our multi type systems. This is expressed by the fact that, in the conclusion of a typing rule, the type context is the result of collecting the type contexts of the premises; that is, no new variables are added. The sole exception to this are the axiom rules, which create type contexts containing *only* the subject variable of the conclusion.
- Conversely, the *contraction* rule in our multi type systems is implemented by applying the sum operator—namely, $\uplus$ for multi types and $\biguplus$ for type contexts—to collect the types appearing in the premises of a typing rule into the conclusion.

We believe this management of the structural rules to be crucial points in the design of the multi type systems, in particular regarding the exact measures that we would like to extract from type derivations. The reason is simply that the multi type system should be designed to be as tight as possible in its management of resources/types. Moreover, some of the case studies—see Chapter 7 (Multi types for Open CbNeed) and Chapter 9 (Multi types for Useful Open CbNeed) below—require delicate analyses on the variables appearing in the domain of type contexts, and so many of our results concerning Open CbNeed and Useful Open CbNeed would not hold if we allowed weakening, for example.

Each of our multi type systems contains a key typing rule that we call the many rule. Roughly, its shape is as follows

$$\frac{(\Gamma_i \vdash^{(i_1,\dots,i_n)} \mathsf{e} : L_i)_{i\in I}}{\biguplus_{i\in I} \Gamma_i \vdash^{(\sum_{i\in I} i_1,\dots,\sum_{i\in I} i_n)} \mathsf{e} : [L_i]_{i\in I}}\ \mathsf{many}$$

Note that the indices in the conclusion are the sum of the indices in the premises, that the type context in the conclusion is the sum of the type contexts in the premises, and that the *linear* types assigned to e in the premises are all collected in a single multi type $[L_i]_{i\in I}$ in the conclusion. Thus, the many rule is used to give controlled access to the multiplicities of linear types when collected into multi types. This suggests that the many rule acts analogously to the promotion rule of the ! connective in Linear Logic.

The many rule is a key typing rule for managing multiplicative redexes in the LSC. Therefore, in each case study we shall apply appropriate and precise restrictions to the many rule, thus implementing central operational features of the corresponding evaluation strategy at the level of its multi type system. These restrictions act on the syntactic shape of the expression as well as on the set of indices I:

- *Syntactic shape of the subject expression*: In the multi type systems of any of the three call-by-need variants, as well as the multi type systems of any of the two call-by-value variants, we only allow the subject of the many-rule to be a λ-abstraction. This is in accordance with the fact that these evaluation strategies only substitute λ-abstractions.
  Instead, the CbN multi type system imposes no restrictions on the subject of the many-rule, in accordance with the fact that all call-by-name variants impose no restrictions on arguments.
- Note that if $I = \emptyset$, then the many rule takes no premises:

$$\frac{}{\emptyset \vdash \mathsf{e} : \mathbf{0}}\ \mathsf{many}$$

This special case may pose a series of problems, especially depending on what the impact of the empty multi type **0** is in each of the multi type systems. Generally speaking, **0** represents the type of erasable expressions, which are handled differently depending on the evaluation strategy under consideration. We thoroughly explore erasure—and duplication, its dual concept—in Chapter 4 (CbN, CbV and CbNeed) from an operational perspective, and from a type-theoretical perspective in Chapter 5 (Multi types for CbN, CbV and CbNeed), where the role of the many rule is made explicit with respect to these concepts.

Linear Logic as inspiration for the linear arrow. We briefly mentioned in Sect. 2.4 (Multi types) that the CbNeed linear type $M \multimap N$ is made of a pair of CbNeed *multi* types, M and N.

But this is not the case for every type system in this work. Indeed, CbN linear and multi types are defined as follows:

CbN linear types	$L, L' ::= \mathsf{norm} \mid M \multimap L$
CbN multi types	$M, N ::= [L_i]_{i \in J}$ (with I finite)

While the CbN linear arrow type $M \multimap L$ take a multi type on the left, it differs with respect to the CbNeed definition in that it takes a *linear* type on the right. This is purposeful, since the approach that we have taken in this respect comes from Girard's call-by-name and call-by-value translations of Intuitionistic Propositional Logic into Linear Logic:

- On the one hand, the call-by-name translation maps $(A \Rightarrow B)^{\mathrm{CbN}}$ to $!(A^{\mathrm{CbN}}) \multimap B^{\mathrm{CbN}}$; hence, linear arrow types of evaluation strategies implementing call-by-name are of the form $M \multimap L$.
- On the other hand, the call-by-value translation maps $(A \Rightarrow B)^{\mathrm{CbV}} = (!A^{\mathrm{CbV}}) \multimap (!B^{\mathrm{CbV}})$; hence, the linear arrow types of evaluation strategies implementing call-by-value are of the form $M \multimap N$. Consequently, evaluation strategies implementing call-by-need—which duplicate arguments in the same way as those implementing call-by-value, as we shall see in Chapter 4 (CbN, CbV and CbNeed)—also has linear arrow types of the form $M \multimap N$.

3.7.1 Axes-based analysis of the multi type systems

Now that each of the case studies has been introduced, and that the more general commonalities between the multi type systems have been made explicit, let us move on to the more specific design principles determining the shape of the multi type systems.

We give here an axes-based presentation of said design principles. Each one of the axes proposes a categorization of the multi type systems, either according to the operational features of the evaluation strategy it is associated with—and the impact this has at the level of the type system—or according to the particular way in which the quantitative information is extracted from its type derivations.

- *Base formalism.* On one hand, the CbN, CbV, CbNeed, and Strong CbV are all based on the LSC. Hence, their multi type systems have ES-dedicated typing rules.
  On the other hand, Open CbNeed and Useful Open CbNeed are both formulated in terms of the *split* LSC. This means their multi type systems have a lifting rule, that turns Λ-terms into programs with an empty environment, as well as ES-dedicated typing rules, that append ESs to the environment.

- *Family of evaluation strategies.* In this work, we chose to study only evaluation strategies implementing either call-by-name, call-by-value or call-by-need. This precise categorization of our evaluation strategies is actually reflected in the design of multi type systems as follows:
  - *The empty multi type*: The role played by $\mathbf{0}$ in the CbV system is central. In fact, it is *the* type used to characterize CbV-normalization.
    On the contrary, the use of $\mathbf{0}$ is deliberately restricted in each of the multi type systems that we give for the CbNeed, Open CbNeed and for Useful Open CbNeed cases. As one of the consequences of this, the axiom rules in these systems shall only involve non-empty multi types.
    The case of Strong CbV is more delicate, as the use of $\mathbf{0}$ is neither restricted nor does it characterize normalization.
  - *Mixing CbN and CbV*: We mentioned above that CbNeed, both as an evaluation strategy and as a multi type system, is defined as an explicit combination of CbN and CbV features, by implementing the erasure mechanisms from CbN and the duplication mechanisms from CbV.
    These operational traits are implemented at the level of the CbN and CbV multi type systems as follows:
    * *CbV duplication*: Once again, the many rule in the CbV multi type system is restricted to λ-abstractions, as CbV only duplicates this kind of terms.
      In addition, the many rule applies with zero premises, thus taking into account the fact that CbV only erases normal forms. This is because erasable and duplicable terms are the same in the weak and closed variant of call-by-value.
    * *CbN erasure*: CbN duplicates terms of any shape, and so the many rule in its multi type system assumes no restrictions on the kind of term being typed.
      In addition, the many rule is applicable with zero premises, thus taking into account the fact that CbN erases any kind of term, even diverging or untypable ones.

    The CbNeed multi type system should be considered as a combination of the CbN and CbV ones, modulo some technicalities. For instance, the many rule in the CbNeed type system is also restricted to λ-abstractions. Unfortunately, we shall see in Chapter 5 (Multi types for CbN, CbV and CbNeed) the handling of erasure in the CbNeed multi type system is delicate and requires dedicated typing rules. We present in Chapter 5 the principles followed in the derivation of the CbNeed multi type system, in particular with respect to erasure.
- *Reduction depth.* The evaluation strategies we consider in this work range from weak and closed ones, to weak and open ones, to strong ones. Recalling that multi type systems are meant to characterize normalization of a certain evaluation strategy, note that the multi type systems targeting a weak strategy should allow us to type λ-abstractions without having to type their bodies. The CbN, CbNeed, Open CbNeed and Useful Open CbNeed multi type systems achieve this via the use of a dedicated typing rule, called norm:

$$\frac{}{\emptyset \vdash \lambda x.t : \mathsf{norm}}\ \mathsf{norm}$$

  where norm is a linear constant type serving precisely this purpose.
  Similarly, the CbV multi type system uses the empty multi type $\mathbf{0}$ to achieve this, covered in the case where the many rule has zero premises:

$$\frac{}{\emptyset \vdash^{(0,0)} \lambda x.t : \mathbf{0}}\ \mathsf{many}$$

This is a possibility in the CbV multi type system because, as we explained above, $\mathbf{0}$ is used to characterize both normalizable and erasable terms in CbV. This is in sheer contrast with the CbN and CbNeed cases, where normalizable terms are those typable with a type different from $\mathbf{0}$.
Of course, since Strong CbV is a strong evaluation strategy, its multi type system does not have a norm linear constant or typing rule. Additionally, normalizable terms in Strong CbV do not equal erasable ones. Therefore, proving normalization of a λ-abstraction in the Strong CbV system unavoidably requires typing its body.

The reduction depth of the evaluation strategy characterized by the type system has another important interesting consequence. Namely, that the conditions under which the Subject Reduction property is proven to hold in each of the case studies depends on the reduction depth of the strategy as follows:

- *No conditions required.* In Chapter 5 (Multi types for CbN, CbV and CbNeed), we shall see that weak and *closed* evaluation strategies do not assume anything in particular on the final type judgement for the corresponding Subject Reduction property to hold.
- *Only the type context.* As we shall see in Chapter 7 (Open CbNeed) and Chapter 9 (Useful Open CbNeed), weak and *open* evaluation strategies require the type context of the final type judgement to satisfy certain properties in order for the the Subject Reduction property to hold. However, no conditions on the right-hand side type of the final type judgement are required.
- *Both the type context and the right-hand side type.* The strong setting, being the more general one, requires both the type context *and* the right-hand side type of the final type judgement to satisfy certain conditions in order for the Subject Reduction property to hold. This is thoroughly analyzed in Chapter 11 (Multi types for Strong CbV)[10].

- *Counting technique for exponential steps.* The Open CbNeed and Useful Open CbNeed systems make use of an exponential-steps counting technique based on the types appearing in axioms.
  To the best of our knowledge, this simple technique is the only one that seems to work. In Sect. 7.2 (Counting techniques for exponential steps) we explore the possibility of applying a different technique to count exponential steps, namely one in "Tight Typings and Split Bounds"[AGK20] used to count exponential steps in the Linear Head reduction case. We then arrive at the conclusion that said technique only works for evaluation strategies implementing call-by-name.
  In fact, we shall see that the Open CbNeed and Useful Open CbNeed type systems differ only in their axioms, precisely because switching from (non-useful) Open CbNeed to Useful Open CbNeed requires analyzing the different (typed) variable occurrences and distinguishing between applied and unapplied ones. We believe this to be a simple and elegant *type-theoretical* distinction between useful and non-useful exponential-steps.
  Furthermore, by adding the axioms of the CbNeed system to this comparison, we get three different ways of counting exponential steps in call-by-need, corresponding to the reasonability requirements for the three different levels described above:
  - *Weak and closed*: In this setting, proving the reasonability of evaluation strategies re-

[10] In addition, we claim that this reasoning should also apply to an eventual multi type system for a strong call-by-need evaluation strategy.

quires implementing sharing mechanisms. Regarding the substitution process, we shall see that exponential steps are shared in call-by-need and in call-by-value, and are not shared in call-by-name. This gets reflected, for example, in the difference between axioms in the multi type systems in Chapter 5 (Multi types for CbN, CbV and CbNeed): typing a variable with a multi type $[L_1, \ldots, L_n]$ in the CbN system takes the following form

$$\frac{\left(\dfrac{}{x : L_i \vdash^{(0,1)} x : L_i}\ \mathsf{ax}\right)_{i=1}^{n}}{x : [L_1, \ldots, L_n] \vdash^{(0,n)} x : [L_1, \ldots, L_n]}\ \mathsf{many}$$

while the same final type judgement may be obtained in the CbV and CbNeed systems by a single application of the axiom rule, as follows:

$$\frac{}{x : [L_1, \ldots, L_n] \vdash^{(0,1)} x : [L_1, \ldots, L_n]}\ \mathsf{ax}$$

Note the difference in the exponential indices in the final type judgements of these type derivations.

– *Weak and open*: Besides implementing sharing mechanisms, attaining reasonability in the weak and open setting additionally requires avoiding substitutions of non-value normal forms. In Open CbNeed, these are the *inert* terms. For this reason, axioms in the Open CbNeed are split as follows:

$$\frac{M \neq [\mathsf{inert}, \ldots, \mathsf{inert}]}{x : M \vdash^{(0,1)} x : M}\ \mathsf{ax} \qquad \frac{M := [\mathsf{inert}, \ldots, \mathsf{inert}]}{x : M \vdash^{(0,0)} x : M}\ \mathsf{ax}_I$$

To be consistent with these axioms, we shall see that λ-abstractions are not typable in the Open CbNeed system with the inert constant. This means that a variable is substituted (by a λ-abstraction) in an Open CbNeed-reduction sequence only if it is typed with a multi type *containing at least a non-*inert *type*.

– *Strong*: Besides the two features explained above, proving the reasonability of a strong evaluation strategy requires that it avoids substitutions of λ-abstractions for *unapplied* variable occurrences. This last point is better implemented first in the weak and open setting and then incorporated into the strong setting, as explained above; this is our Useful Open CbNeed evaluation strategy.
Therefore, axioms in the Useful Open CbNeed system are as follows

$$\frac{M \neq [\mathsf{tight}, \ldots, \mathsf{tight}]}{x : M \vdash^{(0,1)} x : M}\ \mathsf{ax} \qquad \frac{M := [\mathsf{tight}, \ldots, \mathsf{tight}]}{x : M \vdash^{(0,0)} x : M}\ \mathsf{ax}_T$$

where $\mathsf{tight} := \mathsf{inert} \mid \mathsf{abs}$.
To see why this is correct, let us first say that Useful Open CbNeed linear types are either one of the tight constants or the linear arrow type $N \multimap O$. Thus, only axioms typed with a multi type *containing at least a linear arrow type* are part of an exponential step. Indeed, while λ-abstractions shall be typable both with abs and with types of the form $N \multimap O$, the intuition is that abs is the type of *unapplied* variable occurrences (which are not substituted for in Useful Open CbNeed), while $N \multimap O$ is the type of *applied* variable occurrences (which are substituted, as they contribute to the creation of a multiplicative redex).

- *Granularity.* Except for Strong CbV, all the other evaluation strategies in this work implement linear substitutions—that is, the substitution of variable occurrences one at a time. This fact gets reflected in their type-theoretical theories by the presence of a *Linear* Substitution lemma, required to prove the Subject Reduction property, and the presence of a *Linear* Removal lemma, required to prove the Subject Expansion property.
  Conversely, Strong CbV performs meta-level substitutions—which are of a non-linear nature—and so the Strong CbV multi type system requires a (*non-linear*) Substitution lemma to prove its Subject Reduction property, and a (*non-linear*) Removal lemma to prove its Subject Expansion property.
- *Tightness technique vs the unitary shrinkingness technique.* In Chapter 10 (Strong CbV), we use the "*unitary shrinkingness*" technique to characterize a class of type derivations whose indices provide minimal quantities with respect to the corresponding Strong CbV-normalizing sequence. The unitary shrinkingness technique has been successfully used to produce exact measures in other works in the literature, such as [Car18] or [AGK20].

  Conversely, we use a technique called "*tightness*" to achieve the same kind of characterization in each one of the other multi type systems in this work. That is, we take the constant linear types and use them to characterize a class of type derivations whose indices provide minimal quantities with respect to the corresponding normalizing sequence. The tightness technique is explained in detail in [AGK20], which we took inspiration from.

  Since the publishing of [AGK20], the tightness technique has been applied to multiple settings. It has been proven to be a useful and consistent tool for distinguishing, in a given expression, the (syntactic) constructors that are *consumed* along a reduction sequence from the constructors that *persist* along the reduction sequence and make their way to the normal form. In some of these systems, this ability to distinguish constructors has even been used to provide exact upper bounds for the different reduction steps and for the size of the normal form in a *split* fashion—*i.e.*, type derivations provide exact and *separate* upper bounds for each kind of reduction steps as well as for the size of the normal form. In particular, this is the approach adopted in [AGK20].

  Among the multi type systems for which the tightness technique has been used to implement this distinction between consuming and persistent constructors, we would like to mention the following:
  - Bucciarelli, Kesner and Viso's "The Bang Calculus Revisited" [Buc+20], where the authors derive a multi type *system* $\mathcal{U}$ characterizing normalization in the Call-by-Push-Value paradigm—which subsumes the call-by-name and call-by-value operational semantics—and then refine it into a *system* $\mathcal{E}$ by implementing the tightness technique, thus distinguishing consuming and persistent constructors.
  - Kesner and Viso's unpublished work called "The Power of Tightness for Call-by-Push-Value" [KV21], where the authors take *system* $\mathcal{E}$ from [Buc+20] and derive a *system* $\mathcal{N}$ and a *system* $\mathcal{V}$, respectively characterizing normalization for head reduction (*i.e.*, a call-by-name reduction relation) and for open call-by-value reduction [AG16]. Needless to say, these systems distinguish consuming and persistent constructors by implementing the tightness technique.

Chapter 4

CbN, CbV and CbNeed

4.1 Duplication and erasure

The multiplicity of substitutions of a function argument—*i.e.*, given a β-redex $(\lambda x.t)\,u$, the number of times x is going to be replaced by u—has always been considered one of the key phenomena in the λ-calculus. Indeed, many interesting results have historically been attained by imposing (partial) restrictions on the multiplicities of function arguments. To explain what some of these restrictions are, let us first note that, in λ-abstraction $\lambda x.t$, variable x may appear in t

- 0 times, in which case we say that $\lambda x.t$ *erases* the function argument.
- exactly once, in which case we say that $\lambda x.t$ is *linear* on x.
- more than once, in which case we say that $\lambda x.t$ *duplicates* the function argument.

Thus, *duplication* refers to the way in which a particular evaluation strategy treats function arguments needed more than once, while *erasure* corresponds to the way it deals with unneeded function arguments.

There are interesting examples regarding this notion of partial restrictions on duplication and erasure. Take, for instance, the family of Λ-terms $(t_i\mathsf{I})_{i\in\mathbb{N}}$ defined in Sect. 2.3.2—whose duplication induces the so-called *size-explosion*. Or the λI-calculus [Bar84; Kri93]—which disallows erasure by only taking into consideration λ-abstractions $\lambda x.t$ where $x \in \mathsf{fv}(t)$; *i.e.*, $\lambda x.xz$ is a λI-term while $\lambda y.xz$ is not. Among all possible restrictions on duplication and erasure, the notion of *linearity* corresponds to disallowing both duplication and erasure; that is, in the linear λ-calculus we only take under consideration λ-abstractions of the form $\lambda x.t$ where $x \in \mathsf{fv}(t)$ and x appears *exactly once* in t.

Unsurprisingly, restricting these very basic features of the λ-calculus comes at a high price, especially in terms of expressive power. Nonetheless, these restrictions are interesting per se. For example, partial recursive functions—a Turing-complete model of computation equivalent to the λ-calculus according to the Church-Turing thesis—can be defined using only λI-terms[Bar84].

As another example along these lines, consider the fact that every linear Λ-term t strongly $\to_\beta$-normalizes in linear time on $|t|$. We believe that, although the expressiveness of the linear λ-calculus is seriously restricted compared to the one of the λ-calculus, having such a control over the complexity of the normalization process is undoubtedly valuable.

The advent of Linear Logic [Gir87] has given duplication and erasure a prominent logical status. It has reintroduced duplication and erasure into the linear λ-calculus via the non-linear modality !, recovering the full expressive power of cut-elimination while allowing a fine-grained analysis of resource consumption. One could argue that duplication and erasure are then key ingredients for

logical expressiveness, and—via the Curry-Howard correspondence—for the expressiveness of the λ-calculus.

Evaluation strategies may also be seen from a multiplicities point of view. Arguably the most natural evaluation strategies—and probably the most implemented in real-world programming languages—are CbN and CbV evaluation strategies in their weak and closed versions. We call them from now on simply CbN and CbV, and see that they have a dual behavior with respect to duplication and erasure.

We shall cover a third weak and closed evaluation strategy, implementing call-by-need reduction and which we call CbNeed. It can be seen as a combination of CbN and CbV, in that it takes the best of both worlds with respect to duplication and erasure.

Let us see how all these evaluation strategies can be factored in terms of duplication and erasure.

4.2 CbN = silly duplication + wise erasure

CbN *never* evaluates function arguments of β-redexes before the redexes themselves. As a consequence of this, it never reduces subterms that are erased; that is, when a function argument is simply discarded. This is what we call *wise erasure*, and makes CbN a *normalizing strategy*: it reaches a result whenever one exists[1].

A second consequence of prioritizing contracting β-redexes over evaluating the function arguments is that, if the argument of the redex should be duplicated, then it is evaluated more than once. We call this *silly duplication*, as it repeats work already done.

The definition of all three evaluation strategies in this chapter follows a remarkably uniform approach, which we took from Accattoli, Barenbaum and Mazza's "Distilling Abstract Machines" [ABM14]. Since we use the weak restriction of LSC evaluation contexts as the common framework, and use substitution contexts to define the exponential parts of the evaluation strategies, let us recall the definitions:

$$\begin{array}{rrcl} \text{Weak LSC evaluation contexts} & D_W, D'_W & ::= & \langle\cdot\rangle \mid D_W\, t \mid t\, D_W \mid D_W[x{\leftarrow}t] \mid t[x{\leftarrow}D_W] \\ \text{Substitution contexts} & S, S' & ::= & \langle\cdot\rangle \mid S[x{\leftarrow}t] \end{array}$$

Note that since they are given in terms of the LSC, the CbN, CbV and CbNeed defined here are said to have a *micro-step* semantics.

Let us begin by formalizing the CbN evaluation contexts:

$$\text{CbN evaluation contexts} \qquad C \quad ::= \quad \langle\cdot\rangle \mid C\, t \mid C\, [x{\leftarrow}t]$$

Similarly to the definition given in Subsect. 2.2.4 (The λ-calculus as a LSC), we need to define the root-steps of the CbN evaluation strategy. Namely, its notion of multiplicative and exponential root-steps:

$$\begin{array}{rrcl} \text{CbN multiplicative root-step} & S\langle\lambda x.t\rangle u & \mapsto_{\mathsf{m}} & S\langle t\,[x{\leftarrow}u]\rangle \\ \text{CbN exponential root-step} & C\langle\!\langle x\rangle\!\rangle\,[x{\leftarrow}t] & \mapsto_{\mathsf{e}_{\mathrm{CbN}}} & C\langle\!\langle t\rangle\!\rangle\,[x{\leftarrow}t] \end{array}$$

such that in $\mapsto_{\mathsf{m}}$, if $S := \langle\cdot\rangle[y_1{\leftarrow}s_1]...[y_n{\leftarrow}s_n]$, with $n \in \mathbb{N}$, then $y_1, ..., y_n$ and $\mathsf{fv}(u)$ are all pairwise disjoint. Note that this condition can always be fulfilled by α-equivalence[2].

[1] If a term t admits both converging and diverging evaluation sequences in the ordinary λ-calculus, we can see that the diverging sequences occur in erasable subterms of t, which is what CbN avoids reducing.

[2] We also assume this for CbV and CbNeed, as they share the same notion of multiplicative root-step.

The CbN evaluation strategy, which we note $\to_{\mathrm{CbN}}$, is defined as the closure of the CbN root-steps by CbN evaluation contexts. That is, $\to_{\mathrm{CbN}}$ is defined by (any of the two following alternative ways):

$$\to_{\mathrm{CbN}} := C\langle \mapsto_{\mathsf{m}} \cup \mapsto_{\mathsf{e}_{\mathrm{CbN}}} \rangle = C\langle \mapsto_{\mathsf{m}} \rangle \cup C\langle \mapsto_{\mathsf{e}_{\mathrm{CbN}}} \rangle$$

Similarly to the LSC presentation of the λ-calculus, given an evaluation sequence $d\colon t \to^*_{\mathrm{CbN}} u$, we note with $|d|$ the length of d, with $|d|_{\mathsf{m}}$ the number of multiplicative steps in d, and with $|d|_{\mathsf{e}}$ the number of exponential steps in d.

4.2.1 Comparative example of reduction sequences - CbN

Let us present an example of reduction in the CbN evaluation strategy that we shall later recover for the CbV and CbNeed cases. Let $t := ((\lambda x.\lambda y.xx)(II))(II)$ where $I := \lambda z.z$ is the identity combinator. In CbN, t evaluates with 5 multiplicative steps and 5 exponential steps to its $\to_{\mathrm{CbN}}$-normal form, as follows:

$$\begin{array}{ll}
t \to_{\mathsf{m}_{\mathrm{CbN}}} (\lambda y.xx)[x{\leftarrow}II](II) & \to_{\mathsf{m}_{\mathrm{CbN}}} (xx)[y{\leftarrow}II][x{\leftarrow}II] \\
\to_{\mathsf{e}_{\mathrm{CbN}}} ((II)x)[y{\leftarrow}II][x{\leftarrow}II] & \to_{\mathsf{m}_{\mathrm{CbN}}} (z[z{\leftarrow}I]x)[y{\leftarrow}II][x{\leftarrow}II] \\
\to_{\mathsf{e}_{\mathrm{CbN}}} (I[z{\leftarrow}I]x)[y{\leftarrow}II][x{\leftarrow}II] & \to_{\mathsf{m}_{\mathrm{CbN}}} \tilde{x}[\tilde{x}{\leftarrow}x][z{\leftarrow}I][y{\leftarrow}II][x{\leftarrow}II] \\
\to_{\mathsf{e}_{\mathrm{CbN}}} x[\tilde{x}{\leftarrow}x][z{\leftarrow}I][y{\leftarrow}II][x{\leftarrow}II] & \to_{\mathsf{e}_{\mathrm{CbN}}} (II)[\tilde{x}{\leftarrow}x][z{\leftarrow}I][y{\leftarrow}II][x{\leftarrow}II] \\
\to_{\mathsf{m}_{\mathrm{CbN}}} x'[x'{\leftarrow}I][\tilde{x}{\leftarrow}x][z{\leftarrow}I][y{\leftarrow}II][x{\leftarrow}II] & \to_{\mathsf{e}_{\mathrm{CbN}}} I[x'{\leftarrow}I][\tilde{x}{\leftarrow}x][z{\leftarrow}I][y{\leftarrow}II][x{\leftarrow}II]
\end{array}$$

We shall put these quantities in perspective when considering the CbV and CbNeed reduction sequences.

4.3 CbV = wise duplication + silly erasure

On the other hand, CbV *always* evaluates function arguments before contracting the β-redex. Consequently, arguments are evaluated *exactly once*, be it to be duplicated, copied only once, or erased in the contraction of the β-redex. For instance, CbV evaluation diverges on $t := (\lambda x.\lambda y.y)\Omega$—where Ω is the famous looping Λ-term—as it keeps evaluating Ω unsuccessfully. This is in sheer contrast with CbN , as t reduces to $\lambda y.y$ in one single step—simply erasing Ω. We call the way CbV treats erasure *silly erasure*, as it makes it a non-normalizing strategy. But the way CbV treats duplication is called *wise duplication*, as it avoids having to reduce the function argument every time it is needed thereafter.

Like we did in the case of CbN, the CbV is defined as a weak sub-relation of the LSC. Let us begin with its notion of evaluation contexts:

$$\text{CbV evaluation contexts} \qquad V, V' \quad ::= \quad \langle\cdot\rangle \mid Vt \mid tV \mid V[x{\leftarrow}t] \mid t[x{\leftarrow}V]$$

That is, CbV evaluation contexts are nothing but the weak restriction of LSC evaluation contexts. Before seeing how this affects the determinism of CbV, let us continuing defining the evaluation strategy.

The CbV multiplicative and exponential root-steps are defined as follows:

$$\begin{array}{rrcl} \text{CbV multiplicative root-step} & S\langle \lambda x.t\rangle u & \mapsto_{\mathsf{m}} & S\langle t\,[x{\leftarrow}u]\rangle \\ \text{CbV exponential root-step} & V\langle\!\langle x\rangle\!\rangle\,[x{\leftarrow}S\langle v\rangle] & \mapsto_{\mathsf{e}_{\mathrm{CbV}}} & S\langle V\langle\!\langle v\rangle\!\rangle\,[x{\leftarrow}v]\rangle \end{array}$$

Note that CbV multiplicative root-steps are defined exactly as the CbN ones.

Finally, the CbV evaluation strategy, which we note $\to_{\mathrm{CbV}}$, is defined as the closure of the CbV root-steps by CbV evaluation contexts. That is, $\to_{\mathrm{CbV}}$ is defined by (any of the two following alternative ways):

$$\to_{\mathrm{CbN}} := V\langle \mapsto_{\mathsf{m}} \cup \mapsto_{\mathsf{e}_{\mathrm{CbV}}}\rangle = V\langle \mapsto_{\mathsf{m}}\rangle \cup V\langle \mapsto_{\mathsf{e}_{\mathrm{CbV}}}\rangle$$

Given an evaluation sequence $d\colon t \to^*_{\mathrm{CbV}} u$, the definitions of $|d|$, $|d|_{\mathsf{m}}$ and $|d|_{\mathsf{e}}$ are analogous to the CbN case.

4.3.1 Non-determinism of CbV

The only difference between the version of CbV we present here and the one in "Distilling Abstract Machines"[ABM14] is that ours subsumes both the left-to-right and right-to-left versions of CbV therein. This is harmless, because the CbV evaluation strategy enjoys the diamond property:

Proposition 4.3.1 (Diamond property for CbV)**.**
$\to_{\mathrm{CbV}}$ *is diamond.*

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on t.

□

4.3.2 Comparative example of reduction sequences - CbV

Let us come back to the example studied in Sect. 4.2 (CbN = silly duplication + wise erasure). Recall that $t := ((\lambda x.\lambda y.xx)(II))(II)$ takes 5 multiplicative steps and 5 exponential steps to reach its $\to_{\mathrm{CbN}}$-normal form. As it turns out, it also takes 5 multiplicative steps and 5 exponential steps to reach its $\to_{\mathrm{CbV}}$-normal form:

$$\begin{array}{ll} t \to_{\mathsf{m}_{\mathrm{CbV}}} (\lambda x.\lambda y.xx)(II)(z[z{\leftarrow}I]) & \to_{\mathsf{e}_{\mathrm{CbV}}} (\lambda x.\lambda y.xx)(II)(I[z{\leftarrow}I]) \\ \to_{\mathsf{m}_{\mathrm{CbV}}} (\lambda x.\lambda y.xx)(\tilde{x}[\tilde{x}{\leftarrow}I])(I[z{\leftarrow}I]) & \to_{\mathsf{e}_{\mathrm{CbV}}} (\lambda x.\lambda y.xx)(I[\tilde{x}{\leftarrow}I])(I[z{\leftarrow}I]) \\ \to_{\mathsf{m}_{\mathrm{CbV}}} (\lambda y.xx)[x{\leftarrow}I[\tilde{x}{\leftarrow}I]](I[z{\leftarrow}I]) & \to_{\mathsf{m}_{\mathrm{CbV}}} (xx)[y{\leftarrow}I[z{\leftarrow}I]][x{\leftarrow}I[\tilde{x}{\leftarrow}I]] \\ \to_{\mathsf{e}_{\mathrm{CbV}}} (xI)[y{\leftarrow}I[z{\leftarrow}I]][x{\leftarrow}I][\tilde{x}{\leftarrow}I] & \to_{\mathsf{e}_{\mathrm{CbV}}} (II)[y{\leftarrow}I[z{\leftarrow}I]][x{\leftarrow}I][\tilde{x}{\leftarrow}I] \\ \to_{\mathsf{m}_{\mathrm{CbV}}} x'[x'{\leftarrow}I][y{\leftarrow}I[z{\leftarrow}I]][x{\leftarrow}I][\tilde{x}{\leftarrow}I] & \to_{\mathsf{e}_{\mathrm{CbV}}} I[x'{\leftarrow}I][y{\leftarrow}I[z{\leftarrow}I]][x{\leftarrow}I][\tilde{x}{\leftarrow}I] \end{array}$$

However, it should clear that the fact that CbN and CbV take the same number of steps is by chance, as they reduce different redexes: CbN never reduces the unneeded redex II associated to y, but it reduces twice the needed II redex associated to x, while CbV reduces both, but each one only once.

4.4 CbNeed = wise duplication + wise erasure

We think that the "silly erasure" mechanism implemented in CbV is as much of a drawback to CbV as the "silly duplication" mechanism is to CbN. One would like to have an evaluation strategy providing a combination of the advantages of CbN and CbV, discarding their drawbacks. We present here a weak and closed call-by-need evaluation strategy, which we call CbNeed for short and was first introduced by Wadsworth [Wad71].

Despite being at the core of Haskell, one of the most-used functional programming languages, and being at work as a strong evaluation strategy in the kernel of Coq as designed by Bruno Barras [Bar99], the theory of CbNeed is much less developed than that of CbN or CbV. We believe one of the reasons for this is that it cannot be defined inside the λ-calculus without some hacking. Manageable presentations of CbNeed indeed require first-class sharing and micro-step operational semantics.

Another reason is the less natural logical interpretation. We refer the interested reader to the comparison among CbN, CbV and CbNeed from a Curry-Howard perspective presented in "Call-by-name, Call-by-value, Call-by-need, and the Linear Lambda Calculus"[Mar+99]. Therein, the authors interpret the CbN and CbV evaluation orders via the so-called *Girard's CbN and CbV translations*[3] of Intuitionistic Propositional Logic into Linear Logic. They then use the results for CbN and CbV to design a CbNeed translation of Intuitionistic Propositional Logic into an affine calculus. That is, CbNeed can be seen from a logical perspective as the application of Girard's CbV translation of Intuitionistic Logic into an affine calculus, where duplication is controlled and erasure is allowed unrestrictedly—that is, not only on the terms under the scope of a ! modality. Hence, CbNeed can be considered to be a sort of *affine* CbV. This interpretation, however, is rather unusual, given that the CbNeed translation does not provide as exact a match between CbNeed evaluation and cut-elimination in the target affine calculus, as for CbN evaluation (resp. CbV evaluation) and cut-elimination in Linear Logic via Girard's CbN translation (resp. CbV translation).

The reader should keep in mind that, while we present all three evaluation strategies simultaneously in this chapter, it is the CbNeed one that the majority of this work is centered around. Indeed, we only presented the CbN and CbV evaluation strategies—as well as their respective multi type systems in Chapter 5 (Multi types for CbN, CbV and CbNeed), for that matter—in order to develop the CbNeed theory. In particular, considering that CbNeed may be seen as the result of combining features from CbN and CbV.

Let us define the CbNeed evaluation strategy now. First, we require the notion of evaluation contexts:

Definition 9 (CbNeed evaluation contexts)**.**
The class of CbNeed evaluation contexts is given by the following grammar:

$$\text{CbNeed evaluation contexts} \qquad E, E' \ ::= \ \langle\cdot\rangle \mid E\,t \mid E\,[x{\leftarrow}t] \mid E\langle\!\langle x\rangle\!\rangle\,[x{\leftarrow}E']$$

The fourth production, namely $E\langle\!\langle x\rangle\!\rangle\,[x{\leftarrow}E']$, is of particular interest in the CbNeed evaluation strategy, as it implements the *neededness* part of the evaluation strategy. This concept of neededness

[3]The CbN translation $(.)^{\mathrm{CbN}}$ maps $(A \Rightarrow B)^{\mathrm{CbN}}$ to $(!A^{\mathrm{CbN}}) \multimap B^{\mathrm{CbN}}$, while the CbV one, $(.)^{\mathrm{CbV}}$, maps $(A \Rightarrow B)^{\mathrm{CbV}}$ to $!A^{\mathrm{CbV}} \multimap !B^{\mathrm{CbV}}$ or, equivalently, to $!(A^{\mathrm{CbV}} \multimap B^{\mathrm{CbV}})$.

may be related to the concept of erasure in that they are opposite. That is, an argument is erasable if it is never needed[4].

Typically, note that if $x \notin \mathsf{fv}(t)$ then there cannot exist an evaluation context E such that we may write $t = E\langle\langle x \rangle\rangle$. Then, even if $(\lambda x.t)u$ CbNeed-reduces to $t[x \leftarrow u]$, the latter cannot CbNeed-reduce to $E\langle\langle x \rangle\rangle[x \leftarrow u]$, which is the only way in which reduction on argument u may be triggered.

Coming back to the definition of the CbNeed evaluation strategy, let us define its root-steps:

Definition 10 (CbNeed root-steps)**.**
The multiplicative and exponential root-steps for the CbNeed evaluation strategies are

$$\begin{array}{rrcl} \text{CbNeed multiplicative root-step} & S\langle \lambda x.t \rangle u & \mapsto_{\mathsf{m}} & S\langle t\,[x \leftarrow u] \rangle \\ \text{CbNeed exponential root-step} & E\langle\langle x \rangle\rangle\,[x \leftarrow S\langle v \rangle] & \mapsto_{\mathsf{e}_{\text{CbNeed}}} & S\langle E\langle\langle v \rangle\rangle\,[x \leftarrow v] \rangle \end{array}$$

Finally,

Definition 11 (CbNeed evaluation strategy)**.**
The CbNeed evaluation strategy, which we note $\to_{\text{CbNeed}}$, is defined as the closure of the CbNeed root-steps by CbNeed evaluation contexts. That is, $\to_{\text{CbNeed}}$ is defined by (any of the two following alternative ways):

$$\to_{\text{CbNeed}} := E\langle \mapsto_{\mathsf{m}} \cup \mapsto_{\mathsf{e}_{\text{CbNeed}}} \rangle = E\langle \mapsto_{\mathsf{m}} \rangle \cup E\langle \mapsto_{\mathsf{e}_{\text{CbNeed}}} \rangle$$

Given an evaluation sequence $d\colon t \to^*_{\text{CbNeed}} u$, the definitions of $|d|$, $|d|_{\mathsf{m}}$ and $|d|_{\mathsf{e}}$ are analogous to the CbN and CbV cases.

4.4.1 Comparative example of reduction sequences - CbNeed

Let us come back to the example studied in the CbN and CbV cases. Recall that $t := ((\lambda x.\lambda y.xx)(II))(II)$ takes 5 multiplicative steps and 5 exponential steps to reach its $\to_{\text{CbN}}$-normal form or its $\to_{\text{CbV}}$-normal form. CbNeed thus presents an improvement over each of these evaluation strategies, since it takes 4 multiplicative steps and 4 exponential steps to reach is $\to_{\text{CbNeed}}$-normal form:

$$\begin{array}{ll} t \to_{\mathsf{m}_{\text{CbNeed}}} (\lambda y.xx)[x \leftarrow II](II) & \to_{\mathsf{m}_{\text{CbNeed}}} (xx)[y \leftarrow II][x \leftarrow II] \\ \to_{\mathsf{m}_{\text{CbNeed}}} (xx)[y \leftarrow II][x \leftarrow z[z \leftarrow I]] & \to_{\mathsf{e}_{\text{CbNeed}}} (xx)[y \leftarrow II][x \leftarrow I[z \leftarrow I]] \\ \to_{\mathsf{e}_{\text{CbNeed}}} (Ix)[y \leftarrow II][x \leftarrow I][z \leftarrow I] & \to_{\mathsf{m}_{\text{CbNeed}}} (\tilde{x}[\tilde{x} \leftarrow x])[y \leftarrow II][x \leftarrow I][z \leftarrow I] \\ \to_{\mathsf{e}_{\text{CbNeed}}} \tilde{x}[\tilde{x} \leftarrow I][y \leftarrow II][x \leftarrow I][z \leftarrow I] & \to_{\mathsf{e}_{\text{CbNeed}}} I[\tilde{x} \leftarrow I][y \leftarrow II][x \leftarrow I][z \leftarrow I] \end{array}$$

Such an improvement comes from the fact that CbNeed *never* reduces the unneeded redex II associated to y, while it also reduces the needed redex II associated to x *only once*.

[4]The converse does not hold as simply as that. That is, while needed arguments are not erased *right after the multiplicative step*, they are erasable after the last substitution has been performed. More on this in the following Sect. 4.5 (Erasing steps).

4.4.2 Different flavors of weak linear head reduction

In the next Chapter 5 (Multi types for CbN, CbV and CbNeed), we shall see that there are important semantical connections between the three evaluation strategies introduced above and (the relational semantics given to) Linear Logic; such connections serve as a logical foundation for the evaluation strategies we study in this work.

But, from a rewriting-theoretic point of view, there exists also a remarkable logical foundation to our weak and closed evaluation strategies. Let us first point out that our presentation of CbNeed was first studied by Accattoli, Barenbaum and Mazza in their [ABM14], with the intention of giving a uniform presentation of several evaluation strategies within the LSC framework. They also prove that these evaluation strategies correspond to the executions of certain abstract machines—or rather *distill* said abstract machines, following their terminology—both in a qualitative and quantitative way[5].

The authors in [ABM14] claim that these LSC versions of CbN, CbV and CbNeed are the call-by-name, call-by-value and call-by-need versions (respectively) of *weak linear head reduction*. They cite Danos and Regnier [DR04] as the first who ever realized this connection[6].

4.5 Erasing steps

The reader may be surprised by our evaluation strategies, as none of them includes erasing steps, despite the absolute relevance of erasures pointed out previously. There are no contradictions: in the LSC—in contrast to the λ-calculus—erasing steps can always be postponed, and so they are often omitted—see *e.g.* "An abstract factorization theorem for explicit substitutions"[Acc12].

This is actually close to programming language practice, as the garbage collector acts asynchronously with respect to the evaluation flow. For the sake of clarity, let us spell out the erasing rules—they are nonetheless ignored in the remainder of the work. In CbN and CbNeed every Λ-term is erasable, so the erasing root-step takes the following form

$$t[x \leftarrow u] \mapsto_{\mathsf{gc}} t \qquad \text{if } x \notin \mathsf{fv}(t)$$

and it is then closed by weak evaluation contexts.

Since only values are erasable in CbV , the root erasing step in CbV is:

$$t[x \leftarrow S\langle v\rangle] \mapsto_{\mathsf{gc}} S\langle t\rangle \qquad \text{if } x \notin \mathsf{fv}(t)$$

and it is then closed by weak evaluation contexts.

4.6 Characterizing closed normal forms

Throughout this work, we give syntactic characterizations of normal forms in each of the evaluation strategies. There are two alternative ways of doing this: either by giving a context-free grammar—which should correspond exactly to the normal forms in a particular evaluation strategy—or by

[5]Qualitatively, the strategies are sound and complete with respect to the abstract machines. Quantitatively, the number and kind of transitions of executions in each of these abstract machines have a bilinear relation with the evaluation strategies that distill them.

[6]In particular, they proved an exact correspondence between the *KAM* abstract machine and weak head reduction, which we here present in its *linear* variant as CbN.

defining predicates—satisfiable by the normal forms and not satisfied by any other Λ-term. For CbN, CbV and CbNeed we have decided to use predicates, mainly to avoid including yet another syntactic category and especially due to the simple characterizations that these weak and close evaluation strategies have.

It is known that call-by-name and call-by-need reduction relations are termination-equivalent. That is, their sets of normal forms are exactly the same. Hence, we only need two predicates: one for CbV and another one both for CbN and CbNeed.

The difference between these two predicates is that the terms that satisfy the one for CbN and CbNeed may contain any Λ-term inside the ESs, while the terms satisfying the CbV predicate may only contain CbV-normal forms inside the ESs. That is,

$$\frac{}{\mathsf{norm}(\lambda x.t)} \qquad \frac{\mathsf{norm}(t)}{\mathsf{norm}(t[x{\leftarrow}u])} \qquad \frac{}{\mathsf{norm}_{\mathrm{CbV}}(\lambda x.t)} \qquad \frac{\mathsf{norm}_{\mathrm{CbV}}(t) \quad \mathsf{norm}_{\mathrm{CbV}}(u)}{\mathsf{norm}_{\mathrm{CbV}}(t[x{\leftarrow}u])}$$

Proposition 4.6.1 (Syntactic characterization of closed normal forms)**.**
Let t be a closed term.

1. CbN and CbNeed*: For* $\mathsf{r} \in \{\mathrm{CbN}, \mathrm{CbNeed}\}$*, t is* r*-normal if and only if* $\mathsf{norm}(t)$*.*
2. CbV*: t is* CbV*-normal if and only if* $\mathsf{norm}_{\mathrm{CbV}}(t)$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)

- *CbN and CbNeed*: Given t in $\to_{\mathrm{CbN}}$-normal form (resp. $\to_{\mathrm{CbNeed}}$-normal form), the proof of $\mathsf{norm}(t)$ is by structural induction on t, proceeding by case analysis on its shape. Conversely, if $\mathsf{norm}(t)$, then we prove that t is in $\to_{\mathrm{CbN}}$-normal form (resp. $\to_{\mathrm{CbNeed}}$-normal form) by induction on the derivation of $\mathsf{norm}(t)$.
- *CbV*: This proof follows the same structure as the one used for the $\to_{\mathrm{CbN}}$ and $\to_{\mathrm{CbNeed}}$ cases. □

It follows immediately from Proposition 4.6.1 and from the definition of the predicates that normal forms for CbN and CbNeed coincide, while normal forms for CbV are a subset of them. Additionally, this is a proper inclusion since, for instance, the closed term $I[x{\leftarrow}\delta\delta]$, where $I \coloneqq \lambda z.z$ and $\delta \coloneqq \lambda y.yy$, is normal for CbN and CbNeed but not for CbV.

In any case, note that the normal forms of all three evaluation strategies are of the form $S\langle v\rangle$, where S is a substitution context and v is a λ-abstraction. This kind of Λ_{L}-terms is called *answers*.

Chapter 5

Multi types for CbN, CbV and CbNeed

Introduction

In this chapter, we give separate multi type systems for CbN, CbV and CbNeed. Moreover, from each of these type systems we infer an invariant, adequate and compositional *semantics* for its corresponding evaluation strategy. The systems are designed to reflect the characteristics of the strategies that we studied in Chapter 4 (CbN, CbV and CbNeed)—in particular, their implementations of duplication and erasure.

The systems are so finely adjusted to the strategies, that we are able to extract *exact* quantitative information regarding the typed term from (a particular class of) its type derivations. Such a degree of exactness was first accomplished by de Carvalho in 2007 [Car07], when he separately obtained exact bounds for the length of head-normalizing reduction sequences and weak-normalizing reduction sequences. Needless to say, obtaining exact bounds requires a delicate treatment of the features of the systems; our type systems are a proof of this.

Therefore, a thorough presentation for each one of them is necessary. With that being said, we suggest the reader to go through the section on the CbN multi type system first, as many of the intuitions presented therein are useful guidelines for understanding the CbV and CbNeed ones, as well as other type systems in the remainder of the work. Moreover, the proof technique used for the CbN system is, roughly speaking, the same we use throughout.

5.1 Different flavors of multi types

As is the case for each one of our evaluation strategies, the concept of multi types is based on two layers of types, defined in a mutually recursive way: the *linear types* L, and the *multi types* M, which are represented as finite multisets of linear types—as explained in Sect. 2.4 (Multi types). With the idea of gaining useful insights on these notions, let us give intuitive interpretations:

- Linear types correspond to a single use of a term.
- Multi types correspond to the *maximum* number of times a function argument is going to be needed in the body of the function it is an argument of: Suppose we have a type derivation for $S\langle\lambda x.t\rangle u$ in the CbN type system, in such a way that the sub-type derivation for u assigns multi type $M = [L_1, ..., L_n]$ as the type of u. Then we can be sure that, after contracting $S\langle\lambda x.t\rangle u$ in a CbN reduction sequence, u is going to end up (at most) n times in evaluation position.

In Sect. 2.4 (Multi types) on page 24, we presented the definition of linear and multi types for CbNeed, with the intention of introducing the reader to their general structure. But, as we mentioned there, there are subtle differences in the definitions of linear and multi types for each of the case studies. Let us present the definitions corresponding to the CbN, CbV and CbNeed cases:

Definition 12 (Linear and multi types for CbN, CbV and CbNeed)**.**
Let us first recall the definitions of linear and multi types for CbNeed

$$\begin{array}{rl} \textsc{CbNeed linear types} & L, L' ::= \mathsf{norm} \mid M \multimap N \\ \textsc{CbNeed multi types} & M, N ::= [L_i]_{i \in I} \qquad \text{(with I finite)} \end{array}$$

and the CbV linear and multi types, mutually recursively defined as follows:

$$\begin{array}{rl} \textsc{CbV linear types} & L, L' ::= M \multimap N \\ \textsc{CbV multi types} & M, N ::= [L_i]_{i \in I} \qquad \text{(with I finite)} \end{array}$$

While these definitions are structurally the same as the one for CbNeed, one should keep in mind that CbV linear and multi types are with respect to one another, and not with respect to the CbNeed ones.

Finally, linear and multi types for CbN are defined as follows:

$$\begin{array}{rl} \textsc{CbN linear types} & L, L' ::= \mathsf{norm} \mid M \multimap L \\ \textsc{CbN multi types} & M, N ::= [L_i]_{i \in I} \qquad \text{(with I finite)} \end{array}$$

That is, CbN multi types are finite collections of CbN linear types, while the latter are either the constant type norm or a linear arrow type $M \multimap L$. Note that this last production takes a CbN multi type on the left and a CbN *linear* type on the right, which is structurally different than the linear arrow type defined for CbNeed or for CbV. This particular form of the linear arrow type is known in the literature as *strict* types—and was introduced by van Bakel in his [Bak92].

As pioneered in [Car07] by de Carvalho, the size of type derivations in certain multi type systems represent (exact) measurements regarding (or related to) the typed term.

Since each of our multi type systems is designed to characterize termination of a corresponding evaluation strategy, type derivations in our multi type systems provide quantitative information on the size of the normal form of the typed term and the length of the normalizing reduction sequence. This information takes the form of *upper bounds*, which are represented as indices in the final type judgement of a type derivation.

Nonetheless, all three weak and closed evaluation strategies have a very particularly simple kind of normal forms. That is, their normal forms are answers, as we saw in Sect. 4.6 (Characterizing closed normal forms) on page 49. And since reduction does not go under λs in weak settings, the size of normal forms in any of these strategies is always the same. Hence, type derivations in the CbN, CbV and the CbNeed multi type systems do not provide any information regarding the size of normal forms.

On the contrary, the length of the normalizing reduction sequence is of great value, and we extract indeed this information from type derivations, presenting it in the form of 2 indices of

natural numbers, namely (m, e), where m provides information about the number of multiplicative steps and e provides information about the number of exponential steps in the normalizing reduction sequence.

Thus,

Definition 13 (Type judgements)**.**

For every $X \in \{\text{CbN}, \text{CbV}, \text{CbNeed}\}$, we write $\Phi \triangleright_X \Gamma \vdash^{(m,e)} t : M$ to express that type derivation Φ in multi type system X ends in type judgement $\Gamma \vdash^{(m,e)} t : M$, where indices m and e are intended to count the number of multiplicative and exponential steps from t to its X-normal form, respectively.

The same is defined for when the type derivation ends in a type judgement $\Gamma \vdash^{(m,e)} t : L$ instead.

5.2 Multi type system for CbN

Here we introduce the CbN multi type system, together with general intuitions about multi types for the three evaluation strategies of this chapter. We also prove that derivations provide exact bounds on CbN evaluation sequences, and finally induce a semantics.

$$\frac{}{x : [L] \vdash^{(0,1)} x : L}\ \mathsf{ax} \qquad \frac{}{\emptyset \vdash^{(0,0)} \lambda x.t : \mathsf{norm}}\ \mathsf{norm}$$

$$\frac{\Gamma; x : M \vdash^{(m,e)} t : L}{\Gamma \vdash^{(m,e)} \lambda x.t : M \multimap L}\ \mathsf{fun} \qquad \frac{\left(\Gamma_i \vdash^{(m_i,e_i)} t : L_i\right)_{i\in I} \quad I : \text{finite}}{\biguplus_{i\in I} \Gamma_i \vdash^{(\sum_{i\in I} m_i, \sum_{i\in I} e_i)} t : [L_i]_{i\in I}}\ \mathsf{many}$$

$$\frac{\Gamma \vdash^{(m,e)} t : M \multimap L \quad \Pi \vdash^{(m',e')} u : M}{\Gamma \uplus \Pi \vdash^{(m+m'+1, e+e')} tu : L}\ \mathsf{app} \qquad \frac{\Gamma; x : M \vdash^{(m,e)} t : L \quad \Pi \vdash^{(m',e')} u : M}{\Gamma \uplus \Pi \vdash^{(m+m', e+e')} t[x \leftarrow u] : L}\ \text{ES}$$

Figure 5.1: Type system for CbN evaluation

5.2.1 CbN types

The typing rules of the CbN type system are in Fig. 5.1. It is essentially a reformulation of de Carvalho's system $\mathtt{R}$ [Car18], itself being a type-based presentation of the relational model of the CbN λ-calculus induced by the relational model of Linear Logic via the CbN translation of Intuitionistic Propositional Logic into Linear Logic—which, given the Curry-Howard correspondence, means the CbN translation of the λ-calculus into Linear Logic.

Let us overview some subtleties and easy facts about our presentation of the type system:

- Recall that CbN linear and multi types are defined as follows:

CbN linear types	$L, L' ::= \mathsf{norm} \mid M \multimap L$
CbN multi types	$M, N ::= [L_i]_{i\in J}$ (with I finite)

  In particular, note that the linear constant norm is used to type abstractions, which are in CbN-normal form, via typing rule norm it plays a crucial role in our quantitative analysis of CbN evaluation.

- The many has as many premises as the elements in the (possibly empty) set of indices I; when $I = \emptyset$, the rule has no premises, and it types t with the empty multi type $\mathbf{0}$.
  This rule is needed to derive the right-hand-side premises of rules app and ES, that have a multi type M on their right-hand side.
  Essentially, it corresponds to the promotion rule of Linear Logic, that, in the CbN representation of the λ-calculus, is indeed used for typing the right subterm of applications and the content of explicit substitutions.
- *Introduction and destruction of multisets.* Multisets are introduced on the right by the many rule and on the left by ax. Moreover, they are summed on the left by app and ES.
- *Vacuous abstractions.* The abstraction rule fun can always abstract a variable x; note that if $M = \mathbf{0}$, then $\Gamma; x : M$ is equal to Γ.
- *Relevance.* No weakening is allowed in axioms. An easy induction on type derivations shows that

  Lemma 5.2.1 (Relevance of the CbN type system)**.**
  Let $t \in \Lambda_{\mathsf{L}}$ and let $\Phi \triangleright_{CbN} \Gamma \vdash^{(m,e)} t : L$ be a type derivation. If $x \notin \mathsf{fv}(t)$ then $x \notin \mathsf{dom}(\Gamma)$.

  (Click here to see the complete proof in the Technical Appendix)
  Note that Lemma 5.2.1 implies that derivations of closed terms have empty type context. Note that there can be free variables of t not in $\mathsf{dom}(\Gamma)$: the ones occurring only in subterms not touched by the evaluation strategy.
- *Erasable terms and* $\mathbf{0}$. The empty multi type $\mathbf{0}$ is the type of erasable terms. Indeed, abstractions that erase their argument—whose paradigmatic example is $\lambda x.y$—can only be typed with $\mathbf{0} \multimap L$, because of Lemma 5.2.1 (Relevance of the CbN type system). Note that in CbN every term—even diverging ones—can be typed with $\mathbf{0}$ by rule many (taking 0 premises), because, correctly, in CbN every term can be erased.
- *Adequacy and linear types.* All CbN typing rules except for many assign linear types. And many is used only as right premise of the rules app and ES, to derive M. It is with respect to linear types, in fact, that the adequacy of the system is going to be proved: a term is CbN normalizing if and only if it is typable with a linear type, given by Theorem 5.2.5 (Tight Correctness for CbN) and Theorem 5.2.9 (Tight Completeness for CbN) below.

5.2.2 Tight type derivations

A Λ_{L}-term may have several derivations in the CbN type system, indexed by different pairs (m, e). The latter always provide upper bounds on CbN evaluation lengths; specifically, m and e provide bounds on the number of multiplicative and exponential steps, respectively, from the typed term to its CbN-normal form.

One of the interesting aspects of the type systems in this chapter is that there is a simple description of a class of type derivations that provide *exact* upper bounds for these quantities, as we shall show. Their definition relies on the norm type constant.

Definition 14 (Tight derivations for CbN)**.**
A derivation $\Phi \triangleright_{\mathrm{CbN}} \Gamma \vdash^{(m,e)} t : L$ is *tight* if $L = \mathsf{norm}$ and Γ is empty.

5.2.3 Comparative example - derivation in the CbN type system

Let us return to the term $t \coloneqq ((\lambda x.\lambda y.xx)(\mathsf{II}))(\mathsf{II})$, for which a CbN normalizing sequence is given in Sect. 4.2 (CbN: silly duplication + wise erasure). Let us give a derivation for it in the CbN type

system:

First, let us shorten norm to n. Then, we define Φ as the following derivation for the subterm $\lambda x.\lambda y.xx$ of t:

$$\dfrac{\dfrac{\dfrac{}{x:[[\mathsf{n}]\multimap\mathsf{n}]\vdash^{(0,1)} x:[\mathsf{n}]\multimap\mathsf{n}}\,\mathsf{ax} \quad \dfrac{\dfrac{}{x:[\mathsf{n}]\vdash^{(0,1)} x:\mathsf{n}}\,\mathsf{ax}}{x:[\mathsf{n}]\vdash^{(0,1)} x:[\mathsf{n}]}\,\mathsf{many}}{x:[\mathsf{n},[\mathsf{n}]\multimap\mathsf{n}]\vdash^{(1,2)} xx:\mathsf{n}}\,\mathsf{app}}{\dfrac{x:[\mathsf{n},[\mathsf{n}]\multimap\mathsf{n}]\vdash^{(1,2)} \lambda y.xx:\mathbf{0}\multimap\mathsf{n}}{\emptyset\vdash^{(1,2)} \lambda x.\lambda y.xx:[\mathsf{n},[\mathsf{n}]\multimap\mathsf{n}]\multimap(\mathbf{0}\multimap\mathsf{n})}\,\mathsf{fun}}\,\mathsf{fun}$$

Now, we need two derivations for II, one of type n, given by Ψ as follows

$$\dfrac{\dfrac{\dfrac{}{z:[\mathsf{n}]\vdash^{(0,1)} z:\mathsf{n}}\,\mathsf{ax}}{\emptyset\vdash^{(0,1)} \lambda z.z:[\mathsf{n}]\multimap\mathsf{n}}\,\mathsf{fun} \quad \dfrac{\dfrac{}{\emptyset\vdash^{(0,0)} \lambda\tilde{x}.\tilde{x}:\mathsf{n}}\,\mathsf{norm}}{\emptyset\vdash^{(0,0)} \lambda\tilde{x}.\tilde{x}:[\mathsf{n}]}\,\mathsf{many}}{\emptyset\vdash^{(1,1)} \mathsf{II}:\mathsf{n}}\,\mathsf{app}$$

and one of type $[\mathsf{n}]\multimap\mathsf{n}$, given by Ξ as follows

$$\dfrac{\dfrac{\dfrac{}{z:[[\mathsf{n}]\multimap\mathsf{n}]\vdash^{(0,1)} z:[\mathsf{n}]\multimap\mathsf{n}}\,\mathsf{ax}}{\emptyset\vdash^{(0,1)} \lambda z.z:[[\mathsf{n}]\multimap\mathsf{n}]\multimap([\mathsf{n}]\multimap\mathsf{n})}\,\mathsf{fun} \quad \dfrac{\dfrac{\dfrac{}{\tilde{x}:[\mathsf{n}]\vdash^{(0,1)} \tilde{x}:\mathsf{n}}\,\mathsf{ax}}{\emptyset\vdash^{(0,1)} \lambda\tilde{x}.\tilde{x}:[\mathsf{n}]\multimap\mathsf{n}}\,\mathsf{fun}}{\emptyset\vdash^{(0,1)} \lambda\tilde{x}.\tilde{x}:[[\mathsf{n}]\multimap\mathsf{n}]}\,\mathsf{many}}{\emptyset\vdash^{(1,2)} \mathsf{II}:[\mathsf{n}]\multimap\mathsf{n}}\,\mathsf{app}$$

Finally, we put Φ, Ψ and Ξ together in the following derivation Θ for $t=(u(\mathsf{I}))(\mathsf{I})$, where $u \coloneqq \lambda x.\lambda y.xx$:

$$\dfrac{\dfrac{\vdots\,\Phi \qquad \dfrac{\vdots\,\Psi \quad \vdots\,\Xi}{\emptyset\vdash^{(1,1)} \mathsf{II}:\mathsf{n} \quad \emptyset\vdash^{(1,2)} \mathsf{II}:[\mathsf{n}]\multimap\mathsf{n}}\,\mathsf{many}}{\emptyset\vdash^{(2,3)} \mathsf{II}:[\mathsf{n},[\mathsf{n}]\multimap\mathsf{n}]}\,\mathsf{app}}{\emptyset\vdash^{(4,5)} (\lambda x.\lambda y.xx)(\mathsf{II}):\mathbf{0}\multimap\mathsf{n}} \quad \dfrac{}{\emptyset\vdash^{(0,0)} \mathsf{II}:\mathbf{0}}\,\mathsf{many}}{\emptyset\vdash^{(5,5)} ((\lambda x.\lambda y.xx)(\mathsf{II}))(\mathsf{II}):\mathsf{n}}\,\mathsf{app}$$

Note that Θ is a tight derivation and the indices $(5,5)$ correspond to the number of $\mathsf{m}_{\mathrm{CbN}}$-steps and $\mathsf{e}_{\mathrm{CbN}}$-steps, respectively, from t to its CbN-normal form, as shown in Sect. 4.2 (CbN: silly duplication + wise erasure).

Forthcoming Theorem 5.2.5 (Tight Correctness for CbN) shows that this is not by chance: tight derivations for CbN are minimal and provide exact bounds to evaluation lengths in CbN.

The next two subsections prove the two halves of the properties of the CbN type system, namely correctness and completeness.

5.2.4 CbN correctness

Correctness is the fact that *every typable term is CbN-normalizing*. In our setting it comes with additional quantitative information: the indices m and e of a derivation $\Phi \triangleright_{\mathrm{CbN}} \Gamma \vdash^{(m,e)} t : L$ provide bounds for the length of the CbN evaluation of t, which are proved to be exact when the derivation is tight.

The proof technique we use is standard. Moreover, the correctness theorems for all the other evaluation strategies follow *exactly* the same structure. The proof relies on a Quantitative Subject Reduction property showing that the m index decreases by *exactly one* at each $\mathsf{m}_{\mathrm{CbN}}$-step, and that there exists an analogous relation between e and $\mathsf{e}_{\mathrm{CbN}}$-steps.

In turn, Quantitative Subject Reduction relies on a Linear Substitution lemma. Last, while correctness in itself could be proved without anything else, proving the *tight* part—that is, proving that the indices in tight type derivations provide *exact* bounds—requires a further property satisfied by tight type derivations of CbN-normal forms.

Let us point out that correctness is stated with respect to closed terms only, but the auxiliary results have to deal with open terms, since they are proved by induction (over predicates defined by induction) over the structure of terms.

Tightness and Normal Forms. Since indices are (non-negative) natural numbers, note that the decrease in the indices in the Subject Reduction property implies that evaluation must terminate.

Thus, indices are upper bounds to evaluation lengths. But these bounds need not be exact, as evidenced, for example, by the fact that derivations of CbN-normal forms can have strictly positive indices. Consider for instance the following type derivation for $\lambda x.x$, which does not $\mathsf{e}_{\mathrm{CbN}}$-reduce but still the e index is greater than 0:

$$\dfrac{\dfrac{}{x : [L] \vdash^{(0,1)} x : L}\ \mathsf{ax}}{\emptyset \vdash^{(0,1)} \lambda x.x : [L] \multimap L}\ \mathsf{fun}$$

On the contrary, if a type derivation in the CbN system is tight, then it is indexed by $(0, 0)$, as stated in:

Proposition 5.2.2 (Typing properties of CbN-normal forms)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be such that $\mathsf{norm}(t)$, and $\Phi \triangleright_{CbN} \Gamma \vdash^{(m,e)} t : \mathsf{norm}$ be a type derivation. Then $\Gamma = \emptyset$ and $(m, e) = (0, 0)$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{norm}(t)$.

□

Linear Substitution. The Linear Substitution lemma states that substituting over a variable occurrence as in the exponential rule consumes exactly one linear type and decreases of one the exponential index e.

Lemma 5.2.3 (Linear Substitution for CbN)**.**
Let $\Phi_{C\langle\langle x\rangle\rangle} \triangleright_{CbN} \Gamma; x : M \vdash^{(m,e)} C\langle\langle x\rangle\rangle : L$ be a type derivation. Then $e \geq 1$ and there exists a splitting $M = [L'] \uplus N$ such that for every derivation $\Psi \triangleright_{CbN} \Pi \vdash^{(m',e')} t : L'$ there is a derivation

$$\Phi_{C\langle\langle t\rangle\rangle} \triangleright_{CbN} (\Gamma \biguplus \Pi); x : N \vdash^{(m+m',e+e'-1)} C\langle\langle t\rangle\rangle : L$$

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on the CbN evaluation context C.

□

Quantitative Subject Reduction. A key point of multi types is that the size of type derivations shrinks after every evaluation step, which is what allows to bound evaluation lengths. If we take the size of a type derivation to be the sum of the indices in the final type judgement, then this shrinking is obtained by the Quantitative Subject Reduction property, where one of the indices decreases by *exactly* 1; the index decreased depends on the kind of reduction step.

Proposition 5.2.4 (Quantitative Subject Reduction for CbN)**.**
Let $\Phi \rhd_{CbN} \Gamma \vdash^{(m,e)} t : L$ *be a type derivation.*

1. Multiplicative*: If* $t \to_{\mathsf{m}_{\mathrm{CbN}}} u$*, then* $m \geq 1$ *and there exists a derivation*

$$\Psi \rhd_{CbN} \Gamma \vdash^{(m-1,e)} u : L$$

2. Exponential*: If* $t \to_{\mathsf{e}_{\mathrm{CbN}}} u$*, then* $e \geq 1$ *and there exists a derivation*

$$\Psi \rhd_{CbN} \Gamma \vdash^{(m,e-1)} u : L$$

Proof. (Click here to see the complete proof in the Technical Appendix)
The multiplicative case is proved by induction on the CbN evaluation context C such that $t = C\langle s\rangle \to_{\mathsf{m}_{\mathrm{CbN}}} C\langle s'\rangle = u$. The exponential case is proven by induction on the CbN evaluation context C such that $t = C\langle s\rangle \to_{\mathsf{e}_{\mathrm{CbN}}} C\langle s'\rangle = u$, using Lemma 5.2.3 (Linear Substitution for CbN) for the root-step.

□

Finally,

Theorem 5.2.5 (Tight Correctness for CbN)**.**
Let $t \in \Lambda_{\mathsf{L}}$ *be closed and* $\Phi \rhd_{CbN} \Gamma \vdash^{(m,e)} t : L$ *be a type derivation. Then there exists* $u \in \Lambda_{\mathsf{L}}$ *such that*

1. $\mathsf{norm}(u)$*,*
2. *there exists a reduction sequence* $d : t \to^*_{\mathrm{CbN}} u$*, and*
3. $|d|_{\mathsf{m}} \leq m$ *and* $|d|_{\mathsf{e}} \leq e$*.*

Moreover, if Φ *is tight then* $(m, e) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}})$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the sum $m + e$, and proceeding by case analysis on whether $t \to_{\mathrm{CbN}}$-reduces or not:

- If t is in $\to_{\mathrm{CbN}}$-normal form, then we only need to prove the case where Φ is *tight*, which follows by Proposition 5.2.2 (Typing properties of CbN-normal forms).
- If $t \to_{\mathrm{CbN}} u$, then the statement follows by application of Proposition 5.2.4 (Quantitative Subject Reduction for CbN), distinguishing whether $t \to_{\mathrm{CbN}} u$ is a multiplicative or exponential step, and then by application of the *i.h.* on the type derivation for u giving by Proposition 5.2.4. Note that the "Moreover" part follows by the fact that Proposition 5.2.4 preserves the type context and the right-hand side type.

□

Note that Theorem 5.2.5 implicitly states that tight type derivations have *minimal* indices among all derivations.

5.2.5 CbN completeness

Completeness is the fact that *every CbN-normalizing Λ_{L}-term has a (tight) type derivation.* As for correctness, the completeness theorems are always obtained via three intermediate steps, dual to those for correctness: Roughly, one first shows that every normal form has a (tight) derivation. Typability is then extended to CbN-normalizing terms by pulling it back through CbN-evaluation using a Quantitative Subject Expansion property, itself based on a Linear Removal lemma.

Normal Forms. We must first prove that each CbN-normal form is typable; in fact, it is typable with a tight type derivation.

Proposition 5.2.6 (Tight typability of CbN-normal forms)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be such that $\mathsf{norm}(t)$. Then there exists a tight type derivation $\Phi \triangleright_{CbN} \emptyset \vdash^{(0,0)} t : \mathsf{norm}$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{norm}(t)$.

□

Linear Removal. In order to prove the Quantitative Subject Expansion property, we have to first show that typability can also be pulled back along substitutions—*i.e.*, $\mathsf{e}_{\mathrm{CbN}}$-steps—via a Linear Removal lemma dual to the Linear Substitution lemma:

Lemma 5.2.7 (Linear Removal for CbN)**.**
Let $\Phi \triangleright_{CbN} \Gamma; x : M \vdash^{(m,e)} C\langle\langle u \rangle\rangle : L$ be a type derivation, where $x \notin \mathsf{fv}(u)$. Then there exist type derivations

$$\begin{array}{lll} \Psi & \triangleright_{CbN} & \Pi \vdash^{(m',e')} u : L' \\ \Theta & \triangleright_{CbN} & \Delta; x : (M \uplus [L']) \vdash^{(m'',e'')} C\langle\langle x \rangle\rangle : L \end{array}$$

such that $\Gamma = \Pi \biguplus \Delta$ and $(m, e) = (m' + m'', e' + e'' - 1)$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on the CbN evaluation context C.

□

Quantitative Subject Expansion. This property is dual to Proposition 5.2.4 (Quantitative Subject Reduction for CbN):

Proposition 5.2.8 (Quantitative Subject Expansion for CbN)**.**
Let $\Phi \triangleright_{CbN} \Gamma \vdash^{(m,e)} u : L$ be a type derivation.

1. Multiplicative*: If $t \to_{\mathsf{m}_{\mathrm{CbN}}} u$, then there exists a derivation*
$$\Psi \triangleright_{CbN} \Gamma \vdash^{(m+1,e)} t : L$$
2. Exponential*: If $t \to_{\mathsf{e}_{\mathrm{CbN}}} u$, then there exists a derivation*
$$\Psi \triangleright_{CbN} \Gamma \vdash^{(m,e+1)} t : L$$

Proof. (Click here to see the complete proof in the Technical Appendix)
The multiplicative case is proven by induction on the CbN evaluation context C such that $t = C\langle s \rangle \to_{\mathsf{m}_{\mathrm{CbN}}} C\langle s' \rangle = u$. The exponential case is proven by induction on the CbN evaluation context C such that $t = C\langle s \rangle \to_{\mathsf{e}_{\mathrm{CbN}}} C\langle s' \rangle = u$, using Lemma 5.2.7 (Linear Removal for CbN) for the root-step.

□

The Tight Completeness Theorem. The theorem is proved by a straightforward induction on the evaluation length relying on Proposition 5.2.8 (Quantitative Subject Expansion for CbN) for the inductive case, and Proposition 5.2.6 (Tight typability of CbN-normal forms) for the base case—via Proposition 4.6.1.1 (Syntactic characterization of closed normal forms - CbN).

Theorem 5.2.9 (Tight Completeness for CbN)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be closed. If there exists $d : t \to^*_{\mathrm{CbN}} u$ for some $u \in \Lambda_{\mathsf{L}}$ in $\to_{\mathrm{CbN}}$-normal form, then there exists a type derivation $\Phi \rhd_{CbN} \emptyset \vdash^{(|d|_{\mathsf{m}},|d|_{\mathsf{e}})} t : \mathsf{norm}$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the length of reduction sequences starting in t:

- *Base case*: By Proposition 4.6.1.1 (Syntactic characterization of closed normal forms - CbN), we have that $\mathsf{norm}(t)$. We then get Φ by application of Proposition 5.2.6 (Tight typability of CbN-normal forms) on t.
- *Inductive case*: Let $t \to_{\mathsf{m}_{\mathrm{CbN}}} u$. By *i.h.*, we get the expected type derivation for u. We may then obtain the one for t by an application of Proposition 5.2.8 (Quantitative Subject Expansion for CbN), taking into account the kind of reduction step in $t \to_{\mathsf{m}_{\mathrm{CbN}}} u$—*i.e.*, whether it is exponential or multiplicative.

□

5.2.6 Counting erasing steps in the CbN type system

Our system can be easily adapted to measure also garbage collection steps—the CbN erasing rule is shown in Sect. 4.5 (Erasing steps) on page 49. We first extend the quantitative information in type judgements to include a third index g. Second, one needs to distinguish the erasing cases from the non-erasing ones in typing rules app and ES. This is done by splitting each of them in 2, say app into $\mathsf{app}_{\mathrm{gc}}$ and $\mathsf{app}_{\neg\mathrm{gc}}$, and ES into $\mathrm{ES}_{\mathrm{gc}}$ and $\mathrm{ES}_{\neg\mathrm{gc}}$. The discriminating element between these new split rules is the empty multi type $\mathbf{0}$. In the case of the app, we should split it into the following:

$$\frac{\Gamma \vdash^{(m,e,g)} t : \mathbf{0} \multimap L}{\Gamma \uplus \Pi \vdash^{(m,e,g+1)} tu : L}\ \mathsf{app}_{\mathrm{gc}}$$

$$\frac{\Gamma \vdash^{(m,e,g)} t : M \multimap L \quad \Pi \vdash^{(m',e',g')} u : M \quad M \neq \mathbf{0}}{\Gamma \uplus \Pi \vdash^{(m+m'+1,e+e',g+g')} tu : L}\ \mathsf{app}_{\neg\mathrm{gc}}$$

while the ES rule should be split into the following:

$$\frac{\Gamma \vdash^{(m,e,g)} t : L \quad \Gamma(x) = \mathbf{0}}{\Gamma \uplus \Pi \vdash^{(m,e,g+1)} t[x \leftarrow u] : L}\ \mathrm{ES}_{\mathrm{gc}}$$

$$\frac{\Gamma; x : M \vdash^{(m,e,g)} t : L \quad \Pi \vdash^{(m',e',g')} u : M \quad M \neq \mathbf{0}}{\Gamma \uplus \Pi \vdash^{(m+m',e+e',g+g')} t[x \leftarrow u] : L}\ \mathrm{ES}_{\neg\mathrm{gc}}$$

The remaining typing rules—*i.e.*, ax, norm, fun and many—should be extended with a third index, all of them simply adding together the third indices of the premises into the third index of the conclusion. In other words, just like the ax rule is the only typing rule incrementing the e index and the (original) app rule is the only typing rule incrementing the m index, it should be that rules $\mathsf{app}_{\mathrm{gc}}$ and $\mathrm{ES}_{\mathrm{gc}}$ are the only ones incrementing the g index.

Note that the right premise of rules $\mathsf{app}_{\mathrm{gc}}$ and $\mathrm{ES}_{\mathrm{gc}}$ have been removed. This is because the only way to introduce $\mathbf{0}$ in the CbN type system is via a many rule with no premises, whose conclusion has an empty type context and is indexed by $(0,0,0)$.

Given a type derivation $\Phi \rhd_{\mathrm{CbN}} \Gamma \vdash^{(m,e,g)} t : L$ in the CbN type system with extended indices, g happens to be an upper bound on the number of erasing steps. While this upper bound may not be exact in general terms, we believe their analysis here is justified by the fact that these typing rules are a significant part of the CbNeed type system—presented in Sect. 5.4 (Multi type system for CbNeed) on 66.

5.2.7 CbN semantics

The idea used to build the semantics from the multi type system is that the interpretation of a term is simply the set of its type assignments; that is, the set of its derivable types together with their type contexts.

Let $t \in \Lambda_{\mathsf{L}}$ and $x_1, \dots, x_n$ (with $n \geq 0$) be pairwise distinct variables. If $\mathsf{fv}(t) \subseteq \{x_1, \dots, x_n\}$, we say that the list $\vec{x} = (x_1, \dots, x_n)$ is *suitable for* t. Given a suitable list $\vec{x} = (x_1, \dots, x_n)$ for t, the *semantics of* t *for* $\vec{x}$ with respect to the CbN type system is

$$[\![t]\!]^{\mathrm{CbN}}_{\vec{x}} := \{((M_1, \dots, M_n), L) \mid \exists\, \Phi \rhd_{\mathrm{CbN}} x_1 : M_n; \dots; x_n : M_n \vdash^{(m,e)} t : L\}$$

Note that Proposition 5.2.4 (Quantitative Subject Reduction for CbN) and Proposition 5.2.8 (Quantitative Subject Expansion for CbN) guarantee that the semantics $[\![t]\!]^{\mathrm{CbN}}_{\vec{x}}$ of t—for *any* term t, possibly open—is *invariant* by CbN evaluation.

Theorem 5.2.5 (Tight Correctness for CbN) and Theorem 5.2.9 (Tight Completeness for CbN) guarantee that given a *closed* $t \in \Lambda_{\mathsf{L}}$, its interpretation $[\![t]\!]^{\mathrm{CbN}}_{\vec{x}}$ is non-empty if and only if t is CbN-normalizable. That is, Theorem 5.2.5 and Theorem 5.2.9 guarantee that the semantics is *adequate*.

In fact, adequacy also holds with respect to open Λ_{L}-terms. The issue in that case is that the characterization of tight derivations is more involved—see for example Accattoli, Graham-Lengrand and Kesner's "Tight Typings and Split Bounds" [AGK20]. Said differently, weaker correctness and completeness theorems without exact bounds also hold in the open case, but the exactness of bounds requires the Λ_{L}-term to be closed; the same applies in the CbV and CbNeed type systems introduced below.

Just to give a superficial explanation as to why the class of tight type derivations defined for the CbN the multi type system needs to be refined for the open setting, consider the fact that assuming that the type context is empty does not provide a *complete* characterization anymore. Indeed, take any type derivation for $x \in \mathsf{Var}$ and notice that the type context must contain x in its domain.

Finally, note that the CbN semantics is compositional, as explained in Sect. 3.2 (Properties of multi type systems).

5.3 Multi type system for CbV

Here we present Ehrhard's CbV multi type system [Ehr12]—adapted to our presentation of CbV in the LSC—and prove its properties. The system is similar, and yet in many aspects dual, to the CbN one. In particular, their grammars of types is different; recall that CbV linear and multi types

are defined mutually recursively as follows:

CbV linear types	$L, L' ::= M \multimap N$
CbV multi types	$M, N ::= [L_i]_{i\in J}$ (for any finite set J)

Note that CbV linear types have a multi type both as source and as target of the linear arrow $\multimap$, and that the norm constant is absent. This is purposeful, as the role of the norm constant in CbN is played in CbV by the empty multi type $\mathbf{0}$.

The typing rules are in Fig. 5.2. It is an adaptation—to our version of CbV in the LSC —of the type-based presentation of the relational model of the CbV λ-calculus induced by the relational model of Linear Logic via the CbV translation of Intuitionistic Propositional Logic into Linear Logic; *i.e.*, via the Curry-Howard correspondence, the CbV translation of the λ-calculus into Linear Logic.

$$\frac{}{x : M \vdash^{(0,1)} x : M}\ \mathsf{ax}$$

$$\frac{\Gamma; x : N \vdash^{(m,e)} t : M}{\Gamma \vdash^{(m,e)} \lambda x.t : N \multimap M}\ \mathsf{fun} \qquad \frac{\left(\Gamma_i \vdash^{(m_i,e_i)} \lambda x.t : L_i\right)_{i\in I} \quad I : \text{finite}}{\biguplus_{i\in I} \Gamma_i \vdash^{(\sum_{i\in I} m_i, \sum_{i\in I} e_i)} \lambda x.t : [L_i]_{i\in I}}\ \mathsf{many}$$

$$\frac{\Gamma \vdash^{(m,e)} t : [N \multimap M] \quad \Pi \vdash^{(m',e')} u : N}{\Gamma \uplus \Pi \vdash^{(m+m'+1, e+e')} tu : M}\ \mathsf{app} \qquad \frac{\Gamma; x : N \vdash^{(m,e)} t : M \quad \Pi \vdash^{(m',e')} u : N}{\Gamma \uplus \Pi \vdash^{(m+m', e+e')} t[x \leftarrow u] : M}\ \mathsf{ES}$$

Figure 5.2: Type system for CbV evaluation

Let us consider some remarks:

- *Right-hand types.* All rules but fun assign a multi type to the term on the right-hand side, and not a linear type as in the CbN system.
- *Abstractions and* many. The many rule has a restricted form with respect to the CbN one: it can only be applied to abstractions, which happen to be the only terms that can be typed with a linear type.
- *Indices.* Note that the indices are incremented—in rules ax and app—and summed—in rules many and ES—exactly as in the CbN system.

Just like in the CbN case, the CbV multi type system satisfies the following property:

Lemma 5.3.1 (Relevance of the CbV type system)**.**
Let $t \in \Lambda_{\mathsf{L}}$ and let $\Phi \triangleright_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} t : L$ be a type derivation. If $x \notin \mathsf{fv}(t)$ then $x \notin \mathsf{dom}(\Gamma)$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on $\Phi \triangleright_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} t : M$ and proceeding by case analysis on the last derivation rule of Φ.

□

5.3.1 Intuitions: The Empty Type 0

The empty multi type $\mathbf{0}$ plays a special role in CbV. As in CbN, it is the type of terms that can be erased, but, in contrast to CbN, not every term is erasable in CbV.

In the CbN multi type system every term, even a diverging one, is typable with $\mathbf{0}$. On the one hand, this is correct, because in CbN every term can be erased, and erased terms can also be divergent, because they are never evaluated. On the other hand, adequacy is formulated with respect to the norm constant linear type: a Λ_{L}-term CbN-normalizes if and only if it is typable with norm—which, needless to say, is different from $\mathbf{0}$.

Instead, terms in CbV have to be normalized before being erased. Thus, CbV-normalizing terms and erasable terms coincide. Since the multi type system is meant to characterize CbV-normalizing Λ_{L}-terms, a Λ_{L}-term is typable in the CbV system if and only if it is typable with $\mathbf{0}$, as we shall prove below. Thus, the empty type is not a degenerate type excluded for adequacy from the relevant types of a term, as in CbN: it rather is *the* type, characterizing (adequate) typability altogether. This is in fact the reason for the absence of constant norm; an (informal) way to see it is that, in CbV, $\mathsf{norm} = \mathbf{0}$.

Note that, in particular, in the CbV system, a type judgement like $\Gamma \vdash^{(m,e)} t : M$ may be such that the type context Γ assigns the empty type to a variable x occurring in t, as for instance in the axiom $x : \mathbf{0} \vdash^{(0,1)} x : \mathbf{0}$. Although this may seem very strange to people familiar with CbN multi types, we hope that, according to the provided intuition that $\mathbf{0}$ is the type of CbV-normalization, it would rather seem natural instead.

Definition 15 (Tight derivations for CbV)**.**
A derivation $\Phi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} t : M$ is *tight* if $M = \mathbf{0}$ and Γ is empty.

5.3.2 Comparative example - derivation in the CbV type system

Let us consider again the term $t \coloneqq ((\lambda x.\lambda y.xx)(\mathsf{I}\!\mathsf{I}))(\mathsf{I}\!\mathsf{I})$. A CbV reduction sequence for it was given in Sect. 4.3 (CbV: wise duplication + silly erasure), and putting it in perspective with the CbN and CbNeed reduction sequences, and it was typed in the CbN system with a tight derivation—on page 54. We now proceed to type it with a tight type derivation in the CbV system.

First, let us give the derivation Φ_1 for the subterm $\lambda x.\lambda y.xx$ of t:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x : [\mathbf{0} \multimap \mathbf{0}] \vdash^{(0,1)} x : [\mathbf{0} \multimap \mathbf{0}]}\,\mathsf{ax} \quad \dfrac{}{x : \mathbf{0} \vdash^{(0,1)} x : \mathbf{0}}\,\mathsf{ax}}{x : [\mathbf{0} \multimap \mathbf{0}] \vdash^{(1,2)} xx : \mathbf{0}}\,\mathsf{app}}{x : [\mathbf{0} \multimap \mathbf{0}] \vdash^{(1,2)} \lambda y.xx : \mathbf{0} \multimap \mathbf{0}}\,\mathsf{fun}}{x : [\mathbf{0} \multimap \mathbf{0}] \vdash^{(1,2)} \lambda y.xx : [\mathbf{0} \multimap \mathbf{0}]}\,\mathsf{many}}{\emptyset \vdash^{(1,2)} \lambda x.\lambda y.xx : [\mathbf{0} \multimap \mathbf{0}] \multimap [\mathbf{0} \multimap \mathbf{0}]}\,\mathsf{fun}}{\emptyset \vdash^{(1,2)} \lambda x.\lambda y.xx : [[\mathbf{0} \multimap \mathbf{0}] \multimap [\mathbf{0} \multimap \mathbf{0}]]}\,\mathsf{many}$$

Note that $[\mathbf{0} \multimap \mathbf{0}] \uplus \mathbf{0} = [\mathbf{0} \multimap \mathbf{0}]$, which explains the shape of the type context in the conclusion of the app rule. Next, we define the derivation Φ_2 as follows

$$\dfrac{\dfrac{\dfrac{\dfrac{}{z : [\mathbf{0} \multimap \mathbf{0}] \vdash^{(0,1)} z : [\mathbf{0} \multimap \mathbf{0}]}\,\mathsf{ax}}{\emptyset \vdash^{(0,1)} \lambda z.z : [\mathbf{0} \multimap \mathbf{0}] \multimap [\mathbf{0} \multimap \mathbf{0}]}\,\mathsf{fun}}{\emptyset \vdash^{(0,1)} \lambda z.z : [[\mathbf{0} \multimap \mathbf{0}] \multimap [\mathbf{0} \multimap \mathbf{0}]]}\,\mathsf{many} \quad \dfrac{\dfrac{\dfrac{}{\tilde{x} : \mathbf{0} \vdash^{(0,1)} \tilde{x} : \mathbf{0}}\,\mathsf{ax}}{\emptyset \vdash^{(0,1)} \lambda \tilde{x}.\tilde{x} : \mathbf{0} \multimap \mathbf{0}}\,\mathsf{fun}}{\emptyset \vdash^{(0,1)} \lambda \tilde{x}.\tilde{x} : [\mathbf{0} \multimap \mathbf{0}]}\,\mathsf{many}}{\emptyset \vdash^{(1,2)} \mathsf{I}\!\mathsf{I} : [\mathbf{0} \multimap \mathbf{0}]}\,\mathsf{app}$$

and the derivation Φ_3 as follows

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\tilde{y}:\mathbf{0}\vdash^{(0,1)}\tilde{y}:\mathbf{0}}\,\mathsf{ax}}{\emptyset\vdash^{(0,1)}\lambda\tilde{y}.\tilde{y}:\mathbf{0}\multimap\mathbf{0}}\,\mathsf{fun}}{\emptyset\vdash^{(0,1)}\lambda\tilde{y}.\tilde{y}:[\mathbf{0}\multimap\mathbf{0}]}\,\mathsf{many} \qquad \dfrac{}{\emptyset\vdash^{(0,0)}\mathsf{I}:\mathbf{0}}\,\mathsf{many}}{\emptyset\vdash^{(1,1)}\mathsf{II}:\mathbf{0}}\,\mathsf{app}$$

Finally, we put Φ_1, Φ_2 and Φ_3 together in the following derivation Φ for t

$$\dfrac{\dfrac{\dfrac{\vdots\,\Phi_1}{\emptyset\vdash^{(1,2)}\lambda x.\lambda y.xx:[[\mathbf{0}\multimap\mathbf{0}]\multimap[\mathbf{0}\multimap\mathbf{0}]]} \qquad \dfrac{\vdots\,\Phi_2}{\emptyset\vdash^{(1,2)}\mathsf{II}:[\mathbf{0}\multimap\mathbf{0}]}\,\mathsf{app}}{\emptyset\vdash^{(3,4)}(\lambda x.\lambda y.xx)(\mathsf{II}):[\mathbf{0}\multimap\mathbf{0}]} \qquad \dfrac{\vdots\,\Phi_3}{\emptyset\vdash^{(1,1)}\mathsf{II}:\mathbf{0}}}{\emptyset\vdash^{(5,5)}((\lambda x.\lambda y.xx)(\mathsf{II}))(\mathsf{II}):\mathbf{0}}\,\mathsf{app}$$

Note that Φ is a tight derivation and the indices $(5,5)$ correspond to the number of $\mathsf{m}_{\mathrm{CbV}}$-steps and $\mathsf{e}_{\mathrm{CbV}}$-steps, respectively, from t to its CbV-normal form, as shown in Sect. 4.3 (CbV: wise duplication + silly erasure). We shall prove that this is not by chance, as tight derivations for CbV are minimal and provide exact bounds to evaluation lengths in CbV.

Correctness—i.e., that typability in the CbV system implies CbV -normalizability—and *completeness—i.e.*, that CbV -normalizability implies typability in the CbV system—follow exactly the same pattern of the CbN case, *mutatis mutandis*. They include quantitative information about the CbV-normalizing reduction sequence of the typed Λ_{L}-terms.

5.3.3 CbV correctness

The technical development in this section follows the same structure as that of Subsect. 5.2.4 (CbN correctness) on page 55, *mutatis mutandis*. Let us see the main statements for the CbV multi type system:

Proposition 5.3.2 (Typing properties of CbV-normal forms)**.**
Let $t\in\Lambda_{\mathsf{L}}$ be such that $\mathsf{norm}_{\mathrm{CbV}}(t)$, and $\Phi\rhd_{\mathrm{CbV}}\Gamma\vdash^{(m,e)}t:\mathbf{0}$ be a type derivation. Then $\Gamma=\emptyset$ and $(m,e)=(0,0)$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{norm}_{\mathrm{CbV}}(t)$.

□

Lemma 5.3.3 (Linear Substitution for CbV)**.**
Let $\Phi\rhd_{\mathrm{CbV}}\Gamma;x:M\vdash^{(m,e)}V\langle\!\langle x\rangle\!\rangle:N$ be a type derivation and let $v\in\mathsf{Val}$. Then $e\geq 1$ and there exists a splitting $M=O\uplus P$ such that for every type derivation $\Psi\rhd_{\mathrm{CbV}}\Pi\vdash^{(m',e')}v:O$, there exists type derivation

$$\Theta\rhd_{\mathrm{CbV}}\left(\Gamma\biguplus\Pi\right);x:P\vdash^{(m+m',e+e'-1)}V\langle\!\langle v\rangle\!\rangle:N$$

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on V.

□

Proposition 5.3.4 (Quantitative Subject Reduction for CbV)**.**
Let $\Phi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} t : M$ *be a type derivation.*

1. Multiplicative*: If* $t \to_{\mathsf{m}_{\mathrm{CbV}}} u$*, then* $m \geq 1$ *and there exists a derivation*

$$\Psi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m-1,e)} u : M$$

2. Exponential*: If* $t \to_{\mathsf{e}_{\mathrm{CbV}}} u$*, then* $e \geq 1$ *and there exists a derivation*

$$\Psi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m,e-1)} u : M$$

Proof. (Click here to see the complete proof in the Technical Appendix)
The multiplicative case is proven by induction on the CbV evaluation context V such that $t = V\langle s\rangle \to_{\mathsf{m}_{\mathrm{CbV}}} V\langle s'\rangle = u$. The exponential case is proven by induction on the CbV evaluation context V such that $t = V\langle s\rangle \to_{\mathsf{e}_{\mathrm{CbV}}} V\langle s'\rangle = u$, using Lemma 5.3.3 (Linear Substitution for CbV) for the root-step.

□

Theorem 5.3.5 (Tight correctness for CbV)**.**
Let $t \in \Lambda_{\mathsf{L}}$ *be closed and* $\Phi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} t : M$ *be a type derivation. Then there exists* $u \in \Lambda_{\mathsf{L}}$ *such that*

1. $\mathsf{norm}_{\mathrm{CbV}}(u)$*,*
2. *there exists a reduction sequence* $d : t \to^*_{\mathrm{CbV}} u$*, and*
3. $|d|_{\mathsf{m}} \leq m$ *and* $|d|_{\mathsf{e}} \leq e$*.*

Moreover, if Φ *is tight then* $(m, e) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}})$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the sum $m + e$, and proceeding by case analysis on whether $t \to_{\mathrm{CbV}}$-reduces or not:

- If t is in $\to_{\mathrm{CbV}}$-normal form, then we only need to prove the case where Φ is *tight*, which follows by Proposition 5.3.2 (Typing properties of CbV -normal forms).
- If $t \to_{\mathrm{CbV}} u$, then the statement follows by application of Proposition 5.3.4 (Quantitative Subject Reduction for CbV), distinguishing whether $t \to_{\mathrm{CbV}} u$ is a multiplicative or exponential step, and then by application of the *i.h.* on the type derivation for u giving by Proposition 5.3.4. Note that the "Moreover" part follows by the fact that Proposition 5.3.4 preserves the type context and the right-hand side type.

□

Note that Theorem 5.3.5 implicitly states that tight type derivations have *minimal* indices among type derivations.

5.3.4 CbV completeness

Proposition 5.3.6 (Tight typability of CbV -normal forms)**.**
Let $t \in \Lambda_{\mathsf{L}}$ *be such that* $\mathsf{norm}_{\mathrm{CbV}}(t)$*. Then there exists tight type derivation* $\Phi \rhd_{\mathrm{CbV}} \emptyset \vdash^{(0,0)} t : \mathbf{0}$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{norm}_{\mathrm{CbV}}(t)$.

□

Lemma 5.3.7 (Linear Removal for CbV)**.**

Let $\Phi \rhd_{\mathrm{CbV}} \Gamma; x : M \vdash^{(m,e)} V\langle\langle v \rangle\rangle : N$ *be a type derivation, where* $v \in \mathsf{Val}$ *and* $x \notin \mathsf{fv}(v)$*. Then there exist type derivations*

$$\begin{array}{lll} \Psi & \rhd_{\mathrm{CbV}} & \Pi \vdash^{(m',e')} v : O \\ \Theta & \rhd_{\mathrm{CbV}} & \Delta; x : (M \uplus O) \vdash^{(m'',e'')} V\langle\langle x \rangle\rangle : N \end{array}$$

such that $\Gamma = \Pi \biguplus \Delta$ *and* $(m, e) = (m' + m'', e' + e'' - 1)$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)

By structural induction on V.

□

Proposition 5.3.8 (Quantitative Subject Expansion for CbV)**.**

Let $\Phi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} u : M$ *be a type derivation.*

1. Multiplicative*: If* $t \to_{\mathsf{m}_{\mathrm{CbV}}} u$*, then there exists a derivation*

$$\Psi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m+1,e)} t : M$$

2. Exponential*: If* $t \to_{\mathsf{e}_{\mathrm{CbV}}} u$*, then there exists a derivation*

$$\Psi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m,e+1)} t : M$$

Proof. (Click here to see the complete proof in the Technical Appendix)

The multiplicative case is proven by induction on the CbV evaluation context V such that $t = V\langle s \rangle \to_{\mathsf{m}_{\mathrm{CbV}}} V\langle s' \rangle = u$. The exponential case is proven by induction on the CbV evaluation context V such that $t = V\langle s \rangle \to_{\mathsf{e}_{\mathrm{CbV}}} V\langle s' \rangle = u$, using Lemma 5.3.7 (Linear Removal for CbV) for the root-step.

□

Theorem 5.3.9 (Tight Completeness for CbV)**.**

Let $t \in \Lambda_{\mathsf{L}}$ *be closed. If there exists* $d : t \to^*_{\mathrm{CbV}} u$ *for some* u *in* $\to_{\mathrm{CbV}}$*-normal form, then there exists a type derivation* $\Phi \rhd_{\mathrm{CbV}} \emptyset \vdash^{(|d|_{\mathsf{m}},|d|_{\mathsf{e}})} t : \mathbf{0}$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)

By induction on the length of reduction sequences starting in t:

- *Base case*: By Proposition 4.6.1.1 (Syntactic characterization of closed normal forms - CbV), we have that $\mathsf{norm}_{\mathrm{CbV}}(t)$. We then get Φ by application of Proposition 5.3.6 (Tight typability of CbV -normal forms) on t.
- *Inductive case*: Let $t \to_{\mathsf{m}_{\mathrm{CbV}}} u$. By *i.h.*, we get the expected type derivation for u. We may then obtain the one for t by an application of Proposition 5.3.8 (Quantitative Subject Expansion for CbV), taking into account the kind of reduction step in $t \to_{\mathsf{m}_{\mathrm{CbV}}} u$—*i.e.*, whether it is exponential or multiplicative.

□

5.3.5 CbV semantics

The interpretation of Λ_{L}-terms with respect to the CbV system is quite similar—*mutatis mutandis*—to the interpretation of Λ_{L}-terms with respect to the CbN system:

Given a suitable list of variables $\vec{x} = (x_1, \dots, x_n)$ for $t \in \Lambda_{\mathsf{L}}$, the *semantics of t for $\vec{x}$* with respect to the CbV system is

$$[\![t]\!]^{\mathrm{CbV}}_{\vec{x}} \coloneqq \{((M_1, \dots, M_n), N) \mid \exists\, \Phi \triangleright_{\mathrm{CbV}} x_1 : M_1; \dots; x_n : M_n \vdash^{(m,e)} t : N\}$$

Note that the interpretation of t is given only for type derivations in the CbV system that assign it a multi type N. Hence, type derivations for t whose final typing rule is fun are discarded; this is harmless, as we can always extend them via a many rule without modifying none of the (quantitative or qualitative) information in it.

In addition, the *invariance* and the *adequacy* (and the *compositionality*) of the interpretation $[\![t]\!]^{\mathrm{CbV}}_{\vec{x}}$ with respect to CbV evaluation are obtained exactly as for the CbN case.

5.4 Multi type system for CbNeed

5.4.1 CbNeed and denotational semantics

Since CbN and CbNeed are termination-equivalent—see Subsect. 2.1.4 (Different flavors of evaluation strategies)—then every denotational model of CbN is also a model of CbNeed, and vice versa. Moreover, adequacy—*i.e.*, the fact that the denotation of a Λ_{L}-term is not degenerated if and only if CbNeed-normalizes—transfers from CbN to CbNeed. Analogously, every adequate denotational model of CbNeed is also an adequate model of CbN.

Given that denotational semantics are invariant by evaluation, then they are insensitive to evaluation lengths by definition. It could then seem that denotational semantics cannot distinguish between CbN and CbNeed. Remarkably enough, and somewhat counter-intuitively, we are able to separate CbN and CbNeed semantically.

We do this by designing a multi type system finely tuned for CbNeed evaluation, such that for every given $t \in \Lambda_{\mathsf{L}}$, all the type derivations for t in the system provide upper bounds for CbNeed evaluation. Some of these type derivations, called *tight* type derivations, even provide *exact* bounds. Thus, and since CbN evaluation is, generally speaking, less-efficient than CbNeed, while tight type derivations for a Λ_{L}-term in the multi type system for CbNeed provide exact upper bounds for CbNeed, they do not necessarily provide *upper bounds* with respect to CbN evaluation.

Unsurprisingly, the design of the type system requires a delicate mix of erasure and duplication and builds on the Linear Logic understanding of CbN and CbV.

5.4.2 CbNeed as a blend of CbN and CbV

The multi type system for CbNeed is obtained by carefully blending ingredients from the CbN and CbV ones:

- *Wise erasures from CbN.* In CbN, wise erasures are induced by the fact that the empty multi type $\mathbf{0}$—*i.e.*, the type of erasable terms—and the linear type norm—*i.e.*, the type of CbN-normalizable terms—are distinct. Every Λ_{L}-term is typable with $\mathbf{0}$—by using the many rule

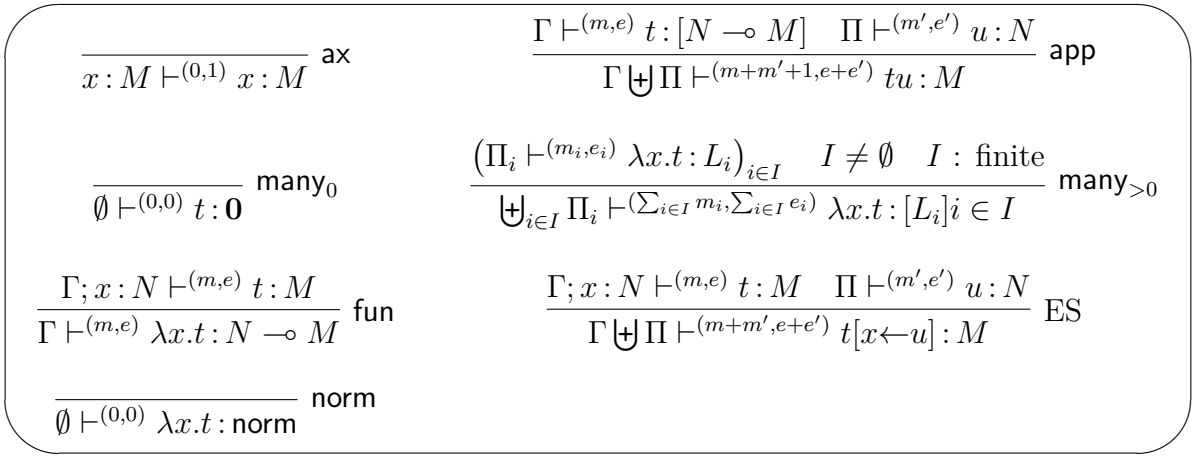

Figure 5.3: Naïve type system for CbNeed evaluation

with 0 premises—while only the CbN-normalizable ones are typable with norm[1]. Adequacy is then formulated with respect to (non-empty) linear types.

- *Wise duplications from CbV.* In CbV, wise duplications are due to two aspects. First, only λ-abstractions can have their linear types collected in multi types by rule many. This fact accounts for the evaluation of arguments to CbV-normal form before being substituted: even if CbV-normal forms are answers—*i.e.*, λ-abstractions in substitution contexts; see Sect. 4.6 (Characterizing closed normal forms)—it is only λ-abstractions that are substituted in CbV. Second, Λ_{L}-terms are typed with multi types instead of linear types. Roughly, this second fact allows the first one to actually work, because the function argument is reduced once for a whole multi set of types, and not once for each element of the multi set, as in CbN.

It seems then that a type system for CbNeed can easily be obtained by basically adopting the CbV system plus

- Separating $\mathbf{0}$ and norm, that is, adding norm to the system.
- Modifying the many rule by distinguishing two cases: if there are no premises, it can assign $\mathbf{0}$ to whatever term, as in the CbN system; otherwise, it is forced to work on λ-abstractions, as in CbV.
- Restricting adequacy to non-empty right multi types, to reflect CbN adequacy into the CbV style of types.

These are the reasons why the grammar of CbNeed linear and multi types, as previously introduced, is:

$$\begin{array}{rl} \textsc{CbNeed linear types} & L, L' ::= \mathsf{norm} \mid M \multimap N \\ \textsc{CbNeed multi types} & M, N ::= [L_i]_{i\in I} \ \text{ (for any finite set } I) \end{array}$$

5.4.3 The Naïve CbNeed type system

In view of the foregoing, we would, as a first attempt, propose the type system in Fig. 5.3 to characterize CbNeed-normalization. Unfortunately, the naïve system does not work, and needs to be re-adjusted. The problem is that tight derivations in this system—defined as expected: empty

[1]This is given by the contrapositive of Theorem 5.2.9 (Tight Completeness for CbN): if $t \in \Lambda_{\mathsf{L}}$ is not typable with norm, then t does not CbN-normalize.

type context and the term typed with [norm]—do not provide exact bounds. This is due to the fact that the naïve blend of ingredients in the system in Fig. 5.3 allows for derivations of $\mathbf{0}$ with strictly positive indices m, and e. Instead, derivations of $\mathbf{0}$ should always have 0 in both indices—as is the case when they are derived with a many_0 rule with 0 premises, as they correspond to terms to be erased, which are not evaluated in CbNeed. For any $t \in \Lambda_{\mathsf{L}}$, indeed, one can for instance derive the following type derivation Φ:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\emptyset \vdash^{(0,0)} x : \mathbf{0}}\,\mathsf{many}_0}{\emptyset \vdash^{(0,0)} \lambda x.x : \mathbf{0} \multimap \mathbf{0}}\,\mathsf{fun}}{\emptyset \vdash^{(0,0)} \lambda x.x : [\mathbf{0} \multimap \mathbf{0}]}\,\mathsf{many}_{>0} \qquad \dfrac{}{\emptyset \vdash^{(0,0)} t : \mathbf{0}}\,\mathsf{many}_0}{\emptyset \vdash^{(1,0)} (\lambda x.x)t : \mathbf{0}}\,\mathsf{app}$$

noting that Φ does not have the expected indices, namely $(0, 0)$. Additionally, introducing $\emptyset \vdash^{(0,1)} x : \mathbf{0}$ with rule ax rather than via many_0 would give a type derivation with final judgement $\emptyset \vdash^{(1,1)} (\lambda x.x)t : \mathbf{0}$. That is, the system can give inexact information in both the multiplicative and the exponential indices, if the inferred type is $\mathbf{0}$.

These cases are not a problem *per se*, because in CbNeed one expects correctness and completeness to hold only for derivations of non-empty right multi types. However, they also affect the quantitative information in some type derivations whose right type is not $\mathbf{0}$, in particular when they appear *inside* tight derivations—that is, as sub-derivations of sub-terms to be erased. Consider for instance:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\emptyset \vdash^{(0,0)} \mathsf{I} : \mathsf{norm}}\,\mathsf{norm}}{\emptyset \vdash^{(0,0)} \mathsf{I} : [\mathsf{norm}]}\,\mathsf{many}_{>0}}{\emptyset \vdash^{(0,0)} \lambda y.\mathsf{I} : \mathbf{0} \multimap [\mathsf{norm}]}\,\mathsf{fun}}{\emptyset \vdash^{(0,0)} \lambda y.\mathsf{I} : [\mathbf{0} \multimap [\mathsf{norm}]]}\,\mathsf{many}_{>0} \qquad \begin{array}{c}\vdots\,\Phi\\ \emptyset \vdash^{(1,0)} (\lambda x.x)t : \mathbf{0}\end{array}}{\emptyset \vdash^{(2,0)} (\lambda y.\mathsf{I})((\lambda x.x)t) : [\mathsf{norm}]}\,\mathsf{app}$$

Despite it being a tight type derivation, note that the term $\rightarrow_{\mathrm{CbNeed}}$-normalizes in just 1 $\mathsf{m}_{\mathrm{CbNeed}}$-step to $I[y{\leftarrow}(\lambda x.x)t]$ while the multiplicative index of the derivation is 2. The mismatch is due to an undesired derivation of $\mathbf{0}$ used as right-hand side premise of an application of the app rule. Similarly, the induced typing of $I[y{\leftarrow}(\lambda x.x)t]$ is an example of an undesired derivation used as right premise of an application of the ES rule:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\emptyset \vdash^{(0,0)} \mathsf{I} : \mathsf{norm}}\,\mathsf{norm}}{\emptyset \vdash^{(0,0)} \mathsf{I} : [\mathsf{norm}]}\,\mathsf{many}_{>0} \qquad \begin{array}{c}\vdots\,\Phi\\ \emptyset \vdash^{(1,0)} (\lambda x.x)t : \mathbf{0}\end{array}}{\emptyset \vdash^{(1,0)} \mathsf{I}[y{\leftarrow}(\lambda x.x)t] : [\mathsf{norm}]}\,\mathsf{ES}$$

5.4.4 The (actual) type system for CbNeed

Our solution to the issue in the type system in Fig. 5.3 is to modify it as to avoid altogether derivations of $\mathbf{0}$ to appear as right-hand side premises of rules app and ES. For this purpose, we follow the schema of the typing rules for counting erasing steps sketched in Sect. 4.5 (Erasing steps).

We therefore add two dedicated rules, called $\mathsf{app}_{\mathrm{gc}}$ and $\mathrm{ES}_{\mathrm{gc}}$, while constraining rules app and ES to be applicable only when the right-hand side type derivation premise has a non-empty right

multi type. This system, whose typing rules appear in Fig. 5.4, is based on the same grammar of linear and multi types of the naïve system. Note that rules many and ax can still introduce $\mathbf{0}$; this, however, does no longer have the undesired effects on the indices of tight derivations, as we are going to show.

$$\frac{}{x : M \vdash^{(0,1)} x : M}\ \mathsf{ax} \qquad \frac{}{\emptyset \vdash^{(0,0)} \lambda x.t : \mathsf{norm}}\ \mathsf{norm}$$

$$\frac{\Gamma; x : N \vdash^{(m,e)} t : M}{\Gamma \vdash^{(m,e)} \lambda x.t : N \multimap M}\ \mathsf{fun} \qquad \frac{\left(\Gamma_i \vdash^{(m_i,e_i)} \lambda x.t : L_i\right)_{i\in I}}{\biguplus_{i\in I} \Gamma_i \vdash^{(\sum_{i\in I} m_i, \sum_{i\in I} e_i)} \lambda x.t : [L_i]_{i\in I}}\ \mathsf{many}$$

$$\frac{\Gamma \vdash^{(m,e)} t : [\mathbf{0} \multimap M]}{\Gamma \vdash^{(m+1,e)} tu : M}\ \mathsf{app}_{\mathrm{gc}} \qquad \frac{\Gamma \vdash^{(m,e)} t : [N \multimap M] \quad \Pi \vdash^{(m',e')} u : N \quad N \neq \mathbf{0}}{\Gamma \uplus \Pi \vdash^{(m+m'+1,e+e')} tu : M}\ \mathsf{app}$$

$$\frac{\Gamma \vdash^{(m,e)} t : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e)} t[x \leftarrow u] : M}\ \mathrm{ES}_{\mathrm{gc}} \qquad \frac{\Gamma; x : N \vdash^{(m,e)} t : M \quad \Pi \vdash^{(m',e')} u : N \quad N \neq \mathbf{0}}{\Gamma \uplus \Pi \vdash^{(m+m',e+e')} t[x \leftarrow u] : M}\ \mathrm{ES}$$

Figure 5.4: Type system for CbNeed evaluation

Note that the indices m and e are incremented and summed exactly as in the CbN and CbV type systems.

Let us stress the fact that our approach for designing the CbNeed multi type system having dedicated typing rules to treat the case of the empty multi type is not an *ad hoc* one, since many multi type systems in the literature implement similar solutions. For instance, the multi type system in [Gar94] has two typing rules for applications, namely $APP1$ and $APP2$, where the second one morally covers the case where the type of the function of the β-redex is $\mathbf{0} \multimap M$.

Just like in the CbN and CbV cases, the CbNeed multi type system satisfies the following:

Lemma 5.4.1 (Relevance of the CbNeed type system)**.**
Let $t \in \Lambda_{\mathsf{L}}$ and let $\Phi \triangleright_{CbNeed} \Gamma \vdash^{(m,e)} t : L$ be a type derivation. If $x \notin \mathsf{fv}(t)$ then $x \notin \mathsf{dom}(\Gamma)$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on $\Phi \triangleright_{\mathrm{CbNeed}} \Gamma \vdash^{(m,e)} t : M$ and proceeding by case analysis on the last derivation rule of Φ.

□

The notion of minimal type derivations for the CbNeed type system is a combination of the ones for the CbN and CbV systems, where the type is a multi type—like in the CbV case—whose content is the norm constant—this corresponds to the CbN case.

Definition 16 (Tight derivations for CbNeed)**.**
A derivation $\Phi \triangleright_{\mathrm{CbNeed}} \Gamma \vdash^{(m,e)} t : M$ is *tight* if $M = [\mathsf{norm}]$ and Γ is empty.

5.4.5 Comparative example - derivation in the CbNeed type system

We return to the term $t := ((\lambda x.\lambda y.xx)(\mathsf{II}))(\mathsf{II})$. Let us give a tight type derivation for it in the CbNeed type system to provide the final, missing part of our running example, putting in contrast CbN, CbV and CbNeed from both operational and type-theoretic points of view.

For the sake of graphical conciseness, let us shorten norm to n and $[\mathsf{n}] \multimap [\mathsf{n}]$ to $[\mathsf{n}]^{[\mathsf{n}]}$. We define Ψ as

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x : [[\mathsf{n}]^{[\mathsf{n}]}] \vdash^{(0,1)} x : [[\mathsf{n}]^{[\mathsf{n}]}]}\,\mathsf{ax} \quad \dfrac{}{x : [\mathsf{n}] \vdash^{(0,1)} x : [\mathsf{n}]}\,\mathsf{ax}}{x : [\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}] \vdash^{(1,2)} xx : [\mathsf{n}]}\,\mathsf{app}}{x : [\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}] \vdash^{(1,2)} \lambda y.xx : \mathbf{0} \multimap [\mathsf{n}]}\,\mathsf{fun}}{x : [\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}] \vdash^{(1,2)} \lambda y.xx : [\mathbf{0} \multimap [\mathsf{n}]]}\,\mathsf{many}}{\emptyset \vdash^{(1,2)} \lambda x.\lambda y.xx : [\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}] \multimap [\mathbf{0} \multimap [\mathsf{n}]]}\,\mathsf{fun}}{\emptyset \vdash^{(1,2)} \lambda x.\lambda y.xx : [[\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}] \multimap [\mathbf{0} \multimap [\mathsf{n}]]]}\,\mathsf{many}$$

and Θ as

$$\dfrac{\dfrac{\dfrac{\dfrac{}{z : [\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}] \vdash^{(0,1)} z : [\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}]}\,\mathsf{ax}}{\emptyset \vdash^{(0,1)} \lambda z.z : [\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}] \multimap [\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}]}\,\mathsf{fun}}{\emptyset \vdash^{(0,1)} \lambda z.z : [[\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}] \multimap [\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}]]}\,\mathsf{many} \quad \dfrac{\dfrac{}{\emptyset \vdash^{(0,0)} \lambda\tilde{x}.\tilde{x} : \mathsf{n}}\,\mathsf{norm} \quad \dfrac{\dfrac{}{\tilde{x} : [\mathsf{n}] \vdash^{(0,1)} \tilde{x} : [\mathsf{n}]}\,\mathsf{ax}}{\emptyset \vdash^{(0,1)} \lambda\tilde{x}.\tilde{x} : [\mathsf{n}]^{[\mathsf{n}]}]}\,\mathsf{fun}}{\emptyset \vdash^{(0,1)} \lambda\tilde{x}.\tilde{x} : [\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}]}\,\mathsf{many}}{\emptyset \vdash^{(1,2)} \mathsf{II} : [\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}]}\,\mathsf{app}$$

Finally, we put Ψ and Θ together in the following derivation Φ for t

$$\dfrac{\dfrac{\dfrac{\vdots\,\Psi \qquad \vdots\,\Theta}{\emptyset \vdash^{(1,2)} \lambda x.\lambda y.xx : [[\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}] \multimap [\mathbf{0} \multimap [\mathsf{n}]]] \quad \emptyset \vdash^{(1,2)} \mathsf{II} : [\mathsf{n}, [\mathsf{n}]^{[\mathsf{n}]}]}\,\mathsf{app}}{\emptyset \vdash^{(3,4)} (\lambda x.\lambda y.xx)(\mathsf{II}) : [\mathbf{0} \multimap [\mathsf{n}]]}\,\mathsf{app}_{\mathrm{gc}}}{\emptyset \vdash^{(4,4)} ((\lambda x.\lambda y.xx)(\mathsf{II}))(\mathsf{II}) : [\mathsf{n}]}\,\mathsf{app}_{\mathrm{gc}}$$

Note that the indices $(4, 4)$ correspond exactly to the number of $\mathsf{m}_{\mathrm{CbNeed}}$-steps and $\mathsf{e}_{\mathrm{CbNeed}}$-steps, respectively, from t to its CbNeed-normal form—as shown in Sect. 4.4 (CbNeed: wise duplication + wise erasure)—and that Φ is a *tight* derivation. We shall see that this is not by chance, as tight type derivations in the CbNeed multi type system are supposed to provide *minimal* indices.

5.4.6 CbNeed correctness

Remarkably, the technical development to prove *correctness* and *completeness* of the CbNeed type system with respect to CbNeed evaluation follows smoothly along the same lines of the two other systems, *mutatis mutandis*:

Proposition 5.4.2 (Typing properties of CbNeed-normal forms)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be such that $\mathsf{norm}(t)$, and $\Phi \triangleright_{CbNeed} \Gamma \vdash^{(m,e)} t : [\mathsf{norm}]$ be a type derivation. Then $\Gamma = \emptyset$ and $(m, e) = (0, 0)$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{norm}(t)$.

□

Lemma 5.4.3 (Linear Substitution for CbNeed)**.**
Let $\Phi \triangleright_{CbNeed} \Gamma; x : M \vdash^{(m,e)} E\langle\langle x \rangle\rangle : N$ be a type derivation and let $v \in \mathsf{Val}$. Then $e \geq 1$ and there exists a splitting $M = O \uplus P$ such that for every derivation $\Psi \triangleright_{\mathrm{CbV}} \Pi \vdash^{(m',e')} v : O$ there is a derivation

$$\Theta \triangleright_{\mathrm{CbV}} \left(\Gamma \biguplus \Pi\right); x : P \vdash^{(m+m', e+e'-1)} V\langle\langle v \rangle\rangle : N$$

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on the CbNeed evaluation context E.

□

Proposition 5.4.4 (Quantitative Subject Reduction for CbNeed)**.**
Let $\Phi \triangleright_{CbNeed} \Gamma \vdash^{(m,e)} t : M$ be a type derivation such that $M \neq \mathbf{0}$.
1. Multiplicative*: If $t \to_{\mathsf{m}_{\mathrm{CbNeed}}} u$, then $m \geq 1$ and there exists a derivation*

$$\Psi \triangleright_{CbNeed} \Gamma \vdash^{(m-1,e)} u : M$$

2. Exponential*: If $t \to_{\mathsf{e}_{\mathrm{CbNeed}}} u$, then $e \geq 1$ and there exists a derivation*

$$\Psi \triangleright_{CbNeed} \Gamma \vdash^{(m,e-1)} u : M$$

Proof. (Click here to see the complete proof in the Technical Appendix)
The multiplicative case is proven by induction on the CbNeed evaluation context E such that $t = E\langle s\rangle \to_{\mathsf{m}_{\mathrm{CbN}}} E\langle s'\rangle = u$. The exponential case is proven by induction on the CbNeed evaluation context E such that $t = E\langle s\rangle \to_{\mathsf{e}_{\mathrm{CbN}}} E\langle s'\rangle = u$, using Lemma 5.4.3 (Linear Substitution for CbNeed) for the root-step.

□

Note the condition $M \neq \mathbf{0}$ in the statement of Proposition 5.4.4. This comes from the fact that erasable and normalizable terms *do not* coincide in CbNeed evaluation—akin to the CbN system and contrary to the CbV one. It is due to the way multi types are used as arguments, via rules $\mathrm{ES}_{\mathrm{gc}}$ and $\mathsf{app}_{\mathrm{gc}}$. This restriction is necessary; consider for instance the following type derivation in the CbNeed type system

$$\dfrac{\dfrac{}{x : \mathbf{0} \vdash^{(0,1)} x : \mathbf{0}}\ \mathsf{ax} \qquad \Gamma(x) = \mathbf{0}}{\emptyset \vdash^{(0,1)} x\,[x \leftarrow \delta\,\delta] : \mathbf{0}}\ \mathrm{ES}$$

where $x[x \leftarrow \delta\delta]$ is not CbNeed-normalizing. This is an expected feature of the system, as it amounts to the fact that adequacy holds only with respect to non-empty right multi types—as for CbN—which was stressed when introducing the CbNeed type system. The same restriction appears in Theorem 5.4.5, Proposition 5.4.8 and Theorem 5.4.9 below, all for the same reason.

Finally,

Theorem 5.4.5 (Tight Correctness for CbNeed)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be closed and $\Phi \triangleright_{CbNeed} \Gamma \vdash^{(m,e)} t : M$ be a type derivation such that $M \neq \mathbf{0}$. Then there exists $u \in \Lambda_{\mathsf{L}}$ such that

1. $\mathsf{norm}(u)$*,*
2. there exists a reduction sequence $d : t \to^{*}_{\mathrm{CbNeed}} u$, and
3. $|d|_{\mathsf{m}} \leq m$ and $|d|_{\mathsf{e}} \leq e$.

Moreover, if Φ is tight then $(m, e) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}})$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the sum $m + e$, and proceeding by case analysis on whether $t \to_{\mathrm{CbNeed}}$-reduces or not:

- If t is in $\to_{\mathrm{CbNeed}}$-normal form, then we only need to prove the case where Φ is *tight*, which follows by Proposition 5.4.2 (Typing properties of CbNeed-normal forms).

- If $t \to_{\mathrm{CbNeed}} u$, then the statement follows by application of Proposition 5.4.4 (Quantitative Subject Reduction for CbNeed), distinguishing whether $t \to_{\mathrm{CbNeed}} u$ is a multiplicative or exponential step, and then by application of the *i.h.* on the type derivation for u giving by Proposition 5.4.4. Note that the "Moreover" part follows by the fact that Proposition 5.4.4 preserves the type context and the right-hand side type.

□

As previously announced, Theorem 5.4.5 is stated with respect to non-empty right multi types. Note that, as for all the other cases, Theorem 5.2.5 implicitly states that tight type derivations have *minimal* indices among derivations.

5.4.7 CbNeed completeness

Proposition 5.4.6 (Tight typability of CbNeed-normal forms)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be such that $\mathsf{norm}(t)$. Then there exists tight type derivation

$$\Phi \rhd_{CbNeed} \emptyset \vdash^{(m,e)} t : [\mathsf{norm}]$$

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{norm}(t)$.

□

Lemma 5.4.7 (Linear Removal for CbNeed)**.**
Let $\Phi \rhd_{CbNeed} \Gamma; x : M \vdash^{(m,e)} E\langle\!\langle v \rangle\!\rangle : N$ be a type derivation, where $v \in \mathsf{Val}$ and $x \notin \mathsf{fv}(v)$. Then there exist type derivations

$$\begin{array}{lll} \Psi & \rhd_{CbNeed} & \Pi \vdash^{(m',e')} v : O \\ \Theta & \rhd_{CbNeed} & \Delta; x : (M \uplus O) \vdash^{(m'',e'')} E\langle\!\langle x \rangle\!\rangle : N \end{array}$$

such that $\Gamma = \Pi \biguplus \Delta$ and $(m, e) = (m' + m'', e' + e'' - 1)$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on the CbNeed evaluation context E.

□

Proposition 5.4.8 (Quantitative Subject Expansion for CbNeed)**.**
Let $\Phi \rhd_{CbNeed} \Gamma \vdash^{(m,e)} u : L$ be a type derivation such that $M \neq \mathbf{0}$.
1. Multiplicative*: If $t \to_{\mathsf{m}_{\mathrm{CbNeed}}} u$, then there exists a derivation*

$$\Psi \rhd_{CbNeed} \Gamma \vdash^{(m+1,e)} t : M$$

2. Exponential*: If $t \to_{\mathsf{e}_{\mathrm{CbNeed}}} u$, then there exists a derivation*

$$\Psi \rhd_{CbNeed} \Gamma \vdash^{(m,e+1)} t : M$$

Proof. (Click here to see the complete proof in the Technical Appendix)
The multiplicative case is proven by induction on the CbNeed evaluation context E such that $t = E\langle s \rangle \to_{\mathsf{m}_{\mathrm{CbNeed}}} E\langle s' \rangle = u$. The exponential case is proven by induction on the CbNeed evaluation context E such that $t = E\langle s \rangle \to_{\mathsf{e}_{\mathrm{CbNeed}}} E\langle s' \rangle = u$, using Lemma 5.4.7 (Linear Removal for CbNeed) for the root-step.

□

Once again, Proposition 5.4.8 is valid only for type derivations of non-empty right multi types, like what happened in Proposition 5.4.4 (Quantitative Subject Reduction for CbNeed).

Theorem 5.4.9 (Tight Completeness for CbNeed)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be closed. If there exists $d : t \to^*_{\mathrm{CbNeed}} u$ for some u in $\to_{\mathrm{CbNeed}}$-normal form, then there exists a type derivation $\Phi \rhd_{CbNeed} \emptyset \vdash^{(|d|_{\mathsf{m}},|d|_{\mathsf{e}})} t : [\mathsf{norm}]$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the length of reduction sequences starting in t:

- *Base case*: By Proposition 4.6.1.1 (Syntactic characterization of closed normal forms - CbNeed), we have that $\mathsf{norm}(t)$. We then get Φ by application of Proposition 5.4.6 (Tight typability of CbNeed-normal forms) on t.
- *Inductive case*: Let $t \to_{\mathsf{m}_{\mathrm{CbNeed}}} u$. By *i.h.*, we get the expected type derivation for u. We may then obtain the one for t by an application of Proposition 5.4.8 (Quantitative Subject Expansion for CbNeed), taking into account the kind of reduction step in $t \to_{\mathsf{m}_{\mathrm{CbNeed}}} u$—*i.e.*, whether it is exponential or multiplicative.

□

5.4.8 CbNeed semantics

The interpretation of Λ_{L}-terms with respect to the CbNeed system is quite similar to the interpretations of Λ_{L}-terms with respect to the CbN system and with respect to the CbV system:

Given a suitable list of variables $\vec{x} = (x_1, \dots, x_n)$ for a $t \in \Lambda_{\mathsf{L}}$, the *semantics of t for $\vec{x}$* with respect to the CbNeed system is

$$[\![t]\!]^{\mathrm{CbNeed}}_{\vec{x}} := \{((M_1, \dots, M_n), N) \mid \exists\, \Phi \rhd_{\mathrm{CbNeed}} x_1 : M_1; \dots; x_n : M_n \vdash^{(m,e)} t : N \text{ and } N \neq \mathbf{0}\}$$

Note that each of the type derivations in the interpretation of t is required to have a non-empty right multi type, in correspondence with the explanations above.

In addition, the *invariance* and *adequacy* (and *compositionality*) of $[\![t]\!]^{\mathrm{CbNeed}}_{\vec{x}}$ with respect to CbNeed evaluation are obtained exactly as for the CbN and CbV cases.

5.5 CbN and CbNeed are termination-equivalent

In the literature, *the* theorem about CbNeed is the fact that it is operationally equivalent to CbN; that is, on a fixed term, CbNeed evaluation terminates if and only if CbN evaluation terminates. Moreover, they essentially produce the same result—up to some technical details that are irrelevant here. This result was first proven independently by two groups, Maraist, Odersky, and Wadler [MOW98], and Ariola and Felleisen [AF97], in the 1990s, using intricate rewriting techniques.

Recently, Kesner gave a much simpler proof via CbN multi types [Kes16]. She uses multi types to first show termination-equivalence of CbN and CbNeed, from which she then infers operational equivalence. Termination-equivalence means that a given term terminates in CbN if and only if terminates in CbNeed, and it is a consequence of our slogan that *CbN and CbNeed both erase wisely*.

With our terminology and notations, Kesner's result takes the following form.

Theorem 5.5.1 (Kesner [Kes16])**.**
Let $t \in \Lambda_{\mathsf{L}}$ be closed.

1. Correctness*: If $\Phi \triangleright_{CbN} \Gamma \vdash^{(m,e)} t : L$, then there exists $u \in \Lambda_{\mathsf{L}}$ such that*
   (a) $\mathsf{norm}(u)$*,*
   (b) there exists a reduction sequence $d\colon t \to^{*}_{\mathrm{CbNeed}} u$, and
   (c) $|d|_{\mathsf{e}} \leq e$ and $|d|_{\mathsf{m}} \leq m$.
2. Completeness*: If $d\colon t \to^{*}_{\mathrm{CbNeed}} u$ and $\mathsf{norm}(u)$, then there exists $\Phi \triangleright_{CbN} \emptyset \vdash^{(m,e)} t : \mathsf{norm}$ for some indices (m, e).*

Note that, unlike the other similar Theorems we gave above, the result does not cover tight type derivations nor does it provide exact bounds. In fact, the CbN system *cannot* provide exact bounds for CbNeed—as explained in Sect. 5.4 (Multi type system for CbNeed)—because it does provide them for CbN evaluation, that in general is less efficient than CbNeed.

For instance, consider the tight CbN type derivation of term $t = ((\lambda x.\lambda y.xx)(II))(II)$ in Subsect. 5.2.3 (Comparative example - derivation in the CbN type system): the derivation provides indices $(5, 5)$ for t—and so t CbN-normalizes in 10 steps—while it CbNeed-normalizes in 8 steps—as shown in Sect. 4.4 (CbNeed = wise duplication + wise erasure). Closing such a gap is precisely the main motivation behind Sect. 5.4 (Multi type system for CbNeed).

5.6 CbNeed is as efficient as CbV

The fact that CbN and CbNeed are termination-equivalent—as just showed in Sect. 5.5 (CbN and CbNeed are termination-equivalent)—is due to the fact that both strategies avoid divergent sequences that reduce function arguments which are only going to be erased later on in the reduction sequence, such as reducing Ω in $(\lambda x.\lambda y.y)\Omega$.

In this sense, termination-equivalence is an abstract theorem stating that CbNeed erases as wisely as CbN. Curiously, in the literature there are no abstract theorems reflecting the dual fact: that CbNeed duplicates as wisely as CbV. One of the reasons for this lack is that it is a theorem that does not admit a simple formulation such as operational or termination-equivalence, given that CbNeed and CbV are not in such relationships. Morally, this is subsumed by the logical interpretation according to which CbNeed corresponds to an affine variant of the Linear Logic representation of CbV. Yet, it would be nice to have a precise, formal statement establishing that *CbNeed duplicates as wisely as CbV*—we provide it here.

Our result is relies on the CbV multi type system being correct with respect to CbNeed evaluation. In particular, the indices (m, e) provided by a CbV type derivation provide upper bounds for the length of a CbNeed-normalizing reduction sequence. Before proceeding with the formal statement, let us give two important remarks:

- *Bounds are not exact.* The indices of a CbV derivation do not generally provide exacts bounds for CbNeed, not even in the case of tight derivations. The reason is that CbNeed does not evaluate unneeded function arguments—which morally correspond to those typed with $\mathbf{0}$—while CbV does. In this sense, CbNeed is even more efficient than CbV. Consider again term $t = ((\lambda x.\lambda y.xx)(\mathsf{II}))(\mathsf{II})$, which CbNeed-normalizes in 8 steps—as shown in Sect. 4.4 (CbNeed: wise duplication + wise erasure)—but whose tight type derivation in the CbV system has indices $(5, 5)$—and so t CbV-normalizes in 10 steps; see Sect. 4.3 (CbV: wise duplication + silly erasure).

- *Completeness cannot hold.* We prove correctness but not completeness simply because the CbV system is not complete with respect to CbNeed evaluation. That is, there are CbNeed-normalizing Λ_{L}-terms that cannot be typed in the CbV system, as typability in that system implies CbV-normalizability by Theorem 5.3.5 (Tight Correctness for CbV). For instance, consider $(\lambda x.\mathsf{I})\Omega$, which cannot be typed in the CbV system and yet is CbNeed-normalizable.

Corollary 5.6.1 (Correctness for CbNeed in the CbV type system)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be closed and let $\Phi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} t : M$ be a type derivation. Then there exists $u \in \Lambda_{\mathsf{L}}$ such that

1. $\mathsf{norm}(u)$,
2. *there exists a reduction sequence $d : t \to^*_{\mathrm{CbNeed}} u$, and*
3. *$|d|_{\mathsf{m}} \leq m$ and $|d|_{\mathsf{e}} \leq e$.*

Proof. (Click here to see the complete proof in the Technical Appendix)
Morally, the proof follows the same schema as the proof of Theorem 5.3.5 (Tight Correctness for CbV), as $\to_{\mathrm{CbNeed}}$ can be easily seen to be a sub-relation of $\to_{\mathrm{CbV}}$.

□

It is pleasant to notice that our presentations of CbV and CbNeed make the proof of Corollary 5.6.1 straightforward. It is enough to observe that, since we do not consider garbage collection and we adopt a non-deterministic formulation of CbV, CbNeed is a subsystem of CbV. Formally, if $t \to_{\mathrm{CbNeed}} u$ then $t \to_{\mathrm{CbV}} u$, which is in fact easily seen from the definitions: CbNeed reduces only *some* subterms of applications and ESs, while CbV allows for reduction on *all* subterms. Corollary 5.6.1 is thus a corollary of the general part in Theorem 5.3.5 (Tight Correctness for CbV).

Finally, since the CbNeed system provides exact bounds—see Theorem 5.4.5 (Tight Correctness for CbNeed)—we get that CbNeed duplicates as wisely as CbV. Note that this comparison can only be considered when it makes sense; that is, only on CbV-normalizable Λ_{L}-terms:

Corollary 5.6.2 (CbNeed duplicates as wisely as CbV)**.**
Let $t, s \in \Lambda_{\mathsf{L}}$ be such that $d\colon t \to^*_{\mathrm{CbV}} s$ and $\mathsf{norm}_{\mathrm{CbV}}(s)$. Then there exist $u \in \Lambda_{\mathsf{L}}$ such that $\mathsf{norm}(u)$, and reduction sequence $d'\colon t \to^*_{\mathrm{CbNeed}} u$ such that $|d'|_{\mathsf{m}} \leq |d|_{\mathsf{m}}$ and $|d'|_{\mathsf{e}} \leq |d|_{\mathsf{e}}$.

Proof. (Click here to see the proof in the Technical Appendix)
By Theorem 5.3.9 (Tight Completeness for CbV), there exists type derivation

$$\Phi \rhd_{\mathrm{CbV}} \emptyset \vdash^{(m,e)} t : \mathbf{0}$$

such that $m = |d|_{\mathsf{m}}$ and $e = |d|_{\mathsf{e}}$. Then Corollary 5.6.1 (Correctness for CbNeed in the CbV type system) yields reduction sequence $d' : t \to^*_{\mathrm{CbNeed}} u$ such that $\mathsf{norm}(u)$, $|d'|_{\mathsf{m}} \leq m = |d|_{\mathsf{m}}$ and $|d'|_{\mathsf{e}} \leq e = |d|_{\mathsf{e}}$.

□

Chapter 6

Open CbNeed

Introduction

Weak reduction on closed terms is the standard model of computation in virtually every functional programming language, like OCaml or Haskell. But other settings sometimes require going further, reducing inside the bodies of λ-abstractions. This is mostly the case in the type checking (resp. proof checking) routines in type systems (resp. logics) based on dependent types. Examples of such systems are the Agda programming language and the Coq proof assistant. As their name hints, dependent types *depend* on terms, in such a way that comparing/establishing the equality of dependent types involves comparing the underlying terms up to *full* β-equivalence. This means β-reducing them to a strong β-normal form and then checking that they are equal, up to α-equivalence.

Evidently, if β-equivalence is to be checked efficiently, then it should at least be implemented by means of a reasonable model of computation—revisit Subsect. 2.3.1 (Reasonable cost models of the λ-calculus) on page 21. This is because models of computation which have not yet been proven reasonable are *unsuited* to efficiency comparisons, as the quantitative relation between their cost model and those of other models of computation has not yet been established.

Furthermore, we are interested in implementing an *efficient* model of computation, among all possible reasonable ones. Recall, for instance, Corollary 5.6.2 (CbNeed duplicates as wisely as CbV) in Sect. 5.6 (CbNeed is as efficient as CbV), which quantitatively compares the two evaluation strategies and concludes that the former is *at least as efficient as* the latter. Similarly, we may be interested in replacing the call-by-name nature of a reduction relation to a call-by-need one, where function arguments are reduced only once; this is the usual *lazy* tweak most implementations of the λ-calculus go through to speed up execution. The idea would thus be to propose a "wise-erasing" version for a reduction relation implementing call-by-name, following the terminology in Chapter 4 (CbN, CbV and CbNeed).

Generally speaking, and regarding the foundation of a resource and *efficiency*-aware theory of β-reduction, an important goal would be to attain a complete study of a call-by-need calculus, performing strong reduction and suitable for defining a useful evaluation strategy in. This would consist in providing denotational and operational semantics, as well as providing design principles for an abstract machine implementing it. Ideally, we would even provide an actual implementation in the form of a compiler, an interpreter, or a β-equivalence checker.

It is in this respect that we now turn to study a less restricted version of the weak and closed call-by-need strategy. We propose the Open CbNeed variant, whose features are:

- *Open terms*: Unlike the weak and closed setting assumed for CbNeed, reduction in Open CbNeed

is also defined for open terms—while remaining conservative with respect to CbNeed on closed terms.

- *Reducing arguments*: Arguments are also evaluated in Open CbNeed. For example, $x\,(\mathsf{I}\mathsf{I})$ is *not* in Open CbNeed-normal form, as reduction should continue on multiplicative redex $\mathsf{I}\mathsf{I}$.
- *Two kinds of normal forms*: Since Open CbNeed operates on open terms, its normal forms are split in two disjoint categories, namely the one of *values* and the one of *inert* terms. For example, $(x\,\mathsf{I})$ is *inert*, as it is normal but it is *not a value*. Note that this situation cannot not happen in CbNeed because normal forms are *closed*.
- *Partially useful*: Open CbNeed does not substitute *any* kind of normal forms, but only values. This is enough to prove the reasonability of Open CbNeed[1].
- *Weakness*: The Open CbNeed is *not a strong* evaluation strategy.

Let us further develop the last point, namely that Open CbNeed is a weak evaluation strategy, as one may wonder how exactly is Open CbNeed an intermediate step towards a strong evaluation strategy. The starting insight is that going from a weak and closed to a strong reduction relation acting on open terms requires two extensions, namely: going under λs, and reducing on arguments of open terms. Of course, these extensions may be applied in any order.

Historically, going under λs is usually considered first, motivated by the semantic relevance of head reduction. That is, one starts from weak head reduction, extends it to strong head reduction, and then to full strong reduction by iterating strong head reduction on arguments. One of the main problems with this approach is that going from weak head to strong head reduction is quite trivial, while most of the difficulties are only dealt with when going from strong head to full strong reduction.

In addition, this issue has historically made it difficult to analyze the complexity of full strong implementations, since going from strong head to full strong reduction then requires simultaneously switching to the open setting, reducing on arguments, avoiding the substitution of inert terms, and avoiding the substitution of values which do not contribute to the creation of new multiplicative redexes. We claim that this may not be the best factorization of the difficulties.

Conversely, we claim that, from an implementative point of view, it is wiser to first cover the reduction-on-arguments extension by iterating weak head reduction on arguments, and only then considering going under λs. This also makes it easier to implement the two useful optimizations required in the strong setting[2]:

- We first extend the weak head reduction by reducing on arguments. In this setting, it is natural to avoid the substitution of *inert* terms—which constitutes the first useful optimization.
- We may then avoid substituting values which do not contribute to the creation of multiplicative redexes—this is the second useful optimization. While it is not needed in the weak and open setting, this optimization is better studied in this scenario as it may then be smoothly extended to the strong case.
- The full strong *and reasonable* variant of the reduction relation may then be obtained by iterating the one in the previous item under λs. This also comprises many subtleties, but is radically simpler than going from strong head to full strong reduction.

Finally, let us mention that the evaluation strategy introduced in this section considerably

[1]The strong setting, on the other hand, requires a further constraint on the substitution process, which is studied in depth in Chapter 8 (Useful Open CbNeed).

[2]This was explained in Sect. 3.5 (Case study: Useful Open CbNeed) on page 34.

builds on the strong call-by-need evaluation strategy defined in Balabonski, Barenbaum, Bonelli and Kesner's "Foundation of Strong Call by Need" [Bal+17]. Nevertheless, our results here are valuable in that we specifically target a type-theoretical interpretation of the consumption of resources, while simultaneously setting the foundation for the useful variant presented in Chapter 8 (Useful Open CbNeed) below.

6.1 The Open CbNeed evaluation strategy

As a starting point to the operational study of the Open CbNeed evaluation strategy, the term syntax of the underlying calculus needs adjusting. This new syntax, which is used to define both the Open CbNeed and the Useful Open CbNeed evaluation strategies, is called the *split LSC* and is a split presentation of the LSC syntax, where (plain) Λ-terms are separated from its ESs.

Instead of being directly based on the LSC, the split LSC takes the λ-calculus as basis, building a slightly more complex structure from it. More concretely, we shall define our evaluation strategies in terms of *programs*, which are pairs of the form $(t, [x_1 \leftarrow t_1] \dots [x_n \leftarrow t_n])$, with $n \geq 0$ and such that $t, t_1, \dots t_n$ are all Λ-terms[3]:

Definition 17 (Terms, environments and programs)**.**

We recall some of the basic categories of Λ-terms, and present the ones of *inert*, *non-variable inert* and *normal* terms:

$$\begin{array}{rl} \text{Terms } (\Lambda) & t, u, s, m ::= \mathsf{Var} \mid \lambda x.t \mid t\, u \\ \text{Values } (\mathsf{Val}) & v, w, v' ::= \lambda x.t \\ \text{Inert terms} & i, j, k, i' ::= \mathsf{Var} \mid i\, n \\ \text{Non-variable inert terms} & i^+, j^+, k^+, i'^+ ::= i\, n \\ \text{Normal terms} & n, m, n' ::= \mathsf{Val} \mid i \end{array}$$

With them, we can define the notion of *environment* as follows:

$$\begin{array}{rl} \text{Environments} & E, E' ::= \epsilon \mid E[x \leftarrow t] \end{array}$$

and finally obtain the grammar on which the reduction relations are defined:

$$\begin{array}{rl} \text{Programs } (\mathcal{PR}) & p, q, r, p' ::= (t, E) \\ \text{Expressions} & \mathsf{e}, \mathsf{e}' := \Lambda \ \cup \ \mathcal{PR} \end{array}$$

Needless to say, evaluation contexts should also be adapted to a split formulation, allowing the (only) context hole to be placed on either one of the components of the pair. To differentiate these contexts from the λ-calculus/LSC evaluation contexts presented in Chapter 2 (Preliminaries), we shall call them "*program contexts*".

Definition 18 (Λ-contexts, term contexts and program contexts)**.**

Just like $\mathcal{PR}$ is based on Λ-terms, program contexts are based on Λ-contexts as given by

$$\begin{array}{rl} \Lambda\text{-contexts} & C, C' ::= \langle\cdot\rangle \mid C\, t \mid t\, C \\ \text{Program contexts} & P, Q, R, S ::= (C, E) \mid (t, E\, [x \leftarrow C]\, E') \end{array}$$

[3]Note that these pairs more closely resemble the structure of environment-based abstract machines than standard LSC terms.

The Open CbNeed reduction relation is defined by means of a proper subclass of Λ-contexts, which we dub *term contexts* and are defined as follows:

$$\text{Term contexts} \qquad \mathcal{H}, \mathcal{J}, \mathcal{I}, \mathcal{H}' ::= \langle\cdot\rangle \mid i\,\mathcal{H} \mid \mathcal{H}\,t$$

The intuition behind term contexts is that subterms on the *left* of the context hole correspond to a certain (adapted) notion of normal, while the subterms on the right are unrestricted. Roughly speaking, this matches the left-to-right evaluation order followed by the Open CbNeed evaluation strategy.

Definition 19 (Domain of programs and program contexts)**.**

The domain of a program or a program context is simply the set of (distinct) variables in the ESs of the environment. Formally, for programs

$$\begin{aligned} \mathsf{dom}_{\mathcal{PR}}(t, \epsilon) &\coloneqq \emptyset \\ \mathsf{dom}_{\mathcal{PR}}(t, E[x{\leftarrow}t]) &\coloneqq \mathsf{dom}_{\mathcal{PR}}(t, E) \cup \{x\} \end{aligned}$$

and for program contexts

$$\begin{aligned} \mathsf{dom}(C, \epsilon) &\coloneqq \emptyset \\ \mathsf{dom}(C, E[x{\leftarrow}t]) &\coloneqq \mathsf{dom}(C, E) \cup \{x\} \\ \mathsf{dom}(t, E[y{\leftarrow}C]E'[x{\leftarrow}u]) &\coloneqq \mathsf{dom}(t, E[y{\leftarrow}C]E') \cup \{x\} \\ \mathsf{dom}(t, E[x{\leftarrow}C]) &\coloneqq \mathsf{dom}_{\mathcal{PR}}(t, E) \cup \{x\} \end{aligned}$$

Note that the definition for program contexts uses the one for programs once the ES containing the context hole has been removed.

The complementary notion to that of domain is the mapping into Λ-terms provided by the domain of an environment. Formally,

Definition 20 (Look ups)**.**

1. Let $p \in \mathcal{PR}$. The *look up* given by p is defined as the (partial) mapping from Var into Λ-terms given by:

$$p(x) \quad \coloneqq \quad \begin{cases} t & \text{, if } p = (u, E\,[x{\leftarrow}t]\,E') \\ \text{undefined} & \text{, otherwise} \end{cases}$$

2. Let P be a program context. The *look up* given by P is defined as the (partial) mapping from Var into Λ-terms given by:

$$P(x) \coloneqq \begin{cases} t & \text{, if } P = (C, E\,[x{\leftarrow}t]\,E') \\ & \text{or } P = (u, E_1\,[x{\leftarrow}t]\,E_2\,[y{\leftarrow}C]\,E') \\ & \text{or } P = (u, E\,[y{\leftarrow}C]\,E'_1\,[x{\leftarrow}t]\,E'_2) \\ \text{undefined} & \text{, otherwise} \end{cases}$$

Given that we are presenting a new grammar of terms, defining evaluation strategies as contextual closure—of some root-steps by a proper notion of evaluation contexts—requires defining how programs should be plugged into program contexts.

Definition 21 (Plugging of programs into program contexts)**.**

Given $(t, E) \in \mathcal{PR}$ and program context P, we define the *plugging* of (t, E) into P, noted $P\langle t, E\rangle$, as follows:

$$P\langle t, E\rangle \quad := \quad \begin{cases} (C\langle t\rangle, EE') & \text{if } P = (C, E') \\ (u, E'[x{\leftarrow}C\langle t\rangle]EE'') & \text{if } P = (u, E'[x{\leftarrow}C]E'') \end{cases}$$

where EE' is the environment resulting from concatenating E and E' as expected, and the same applies to the other concatenations in the definition. For simplicity, we write $P\langle t\rangle$ for $P\langle t, \epsilon\rangle$—note the empty environment.

Definition 22 (Appending explicit substitutions to program contexts)**.**

Given a program context $P := (t, E)$ and an explicit substitution $[x{\leftarrow}u]$, we write $P@[x{\leftarrow}u]$ to denote program context $(t, E[x{\leftarrow}u])$.

6.1.1 Needed variables

Our whole analysis of the Open CbNeed evaluation strategy, both from an operational and semantical point of view, is structured around a particular subset of the free variables of an expression that we call the *needed variables* of the expression. The definition is given first for Λ-terms, upon which the one for programs builds.

Definition 23 (Needed variables of terms and programs)**.**

Let us first define the *base* case for the needed variables of programs—namely, the needed variables of Λ-terms, which are morally the free variables that have occurrences out of all λ-abstractions:

$$\begin{array}{rcl} \mathsf{nv}(x) & := & \{x\} \\ \mathsf{nv}(\lambda x.t) & := & \emptyset \\ \mathsf{nv}(t\, u) & := & \mathsf{nv}(t) \cup \mathsf{nv}(u) \end{array}$$

The needed variables of programs are then defined as follows:

$$\begin{array}{rcl} \mathsf{nv}(t, \epsilon) & := & \emptyset \\ \mathsf{nv}(t, E\,[x{\leftarrow}u]) & := & \begin{cases} \mathsf{nv}(t, E) & \text{if } x \notin \mathsf{nv}(t, E) \\ (\mathsf{nv}(t, E) \setminus \{x\}) \cup \mathsf{nv}(u) & \text{if } x \in \mathsf{nv}(t, E) \end{cases} \end{array}$$

Thus, the needed variables of programs are an extrapolation of the notion of needed variables of Λ-terms: given $(t, E[x{\leftarrow}u])$, the principle is that the needed variables of u are included in those of (t, E) if and only if $x \in \mathsf{nv}(t, E)$. In other words, needed variables are obtained *hereditarily*, replacing needed variables of subexpressions bound by ESs by the needed variables of the term in said ES. For example, x is not considered to be a needed variable in $(x, [x{\leftarrow}yz])$, as it is bound by $[x{\leftarrow}yz]$ and is then replaced by y and z.

In order to define the program contexts at play in the Open CbNeed reduction relation, we need to extend needed variables to programs contexts, starting with Λ-contexts.

Definition 24 (Needed variables of Λ-contexts)**.**

The set of needed variables of Λ-contexts is given by the following equations:

$$\begin{array}{rcl} \mathsf{nv}(\langle\cdot\rangle) & := & \emptyset \\ \mathsf{nv}(C\, t) & := & \mathsf{nv}(C) \\ \mathsf{nv}(t\, C) & := & \mathsf{nv}(t) \cup \mathsf{nv}(C) \end{array}$$

6.1.2 Evaluation strategy

Before formally defining the Open CbNeed evaluation strategy, let us give some examples showing what it should look like, using symbol $\to_{\mathsf{ond}}$ for the reduction relation. We believe these intuitions may guide the reader in the definitions given below, since the latter are rather technical.

- *Reducing arguments*: This feature is implemented by term contexts, which allow us to switch focus to arguments when a head normal form has been reached. For instance, note that $x\,\langle\cdot\rangle$ is a term context, and so the following multiplicative step should apply:

  $$(x((\lambda z.z)\mathsf{I}), [y{\leftarrow}t]) = (x\langle\cdot\rangle, [y{\leftarrow}t])\langle(\lambda z.z)\mathsf{I}\rangle \to_{\mathsf{ond}} (x\langle\cdot\rangle, [y{\leftarrow}t])\langle z, [z{\leftarrow}\mathsf{I}]\rangle = (xz, [z{\leftarrow}\mathsf{I}][y{\leftarrow}t])$$

  Similarly, the following exponential step should apply:

  $$(xz, [z{\leftarrow}\mathsf{I}][y{\leftarrow}t]) = (x\langle\cdot\rangle, [z{\leftarrow}\mathsf{I}][y{\leftarrow}t])\langle z\rangle \to_{\mathsf{ond}} (x\langle\cdot\rangle, [z{\leftarrow}\mathsf{I}][y{\leftarrow}t])\langle\mathsf{I}^{\alpha}\rangle = (x\mathsf{I}^{\alpha}, [z{\leftarrow}\mathsf{I}][y{\leftarrow}t])$$

- *Hereditary reduction*: Similarly to the $E_1\langle\!\langle x\rangle\!\rangle[x{\leftarrow}E_2]$ production for CbNeed evaluation contexts defined on page 47, the typical hereditary reduction performed in call-by-need is implemented by means of term contexts. For instance, since $(\langle\cdot\rangle t)$ and $(y\langle\cdot\rangle)$ are both term contexts, then the following multiplicative step should apply:

  $$(xt, [x{\leftarrow}y((\lambda z.z)\mathsf{I})]) = (xt, [x{\leftarrow}y\langle\cdot\rangle])\langle(\lambda z.z)\mathsf{I}\rangle \to_{\mathsf{ond}} (xt, [x{\leftarrow}y\langle\cdot\rangle])\langle z, [z{\leftarrow}\mathsf{I}]\rangle = (xt, [x{\leftarrow}yz][z{\leftarrow}\mathsf{I}])$$

  and so the following exponential step should apply as well:

  $$(xt, [x{\leftarrow}yz][z{\leftarrow}\mathsf{I}]) = (xt, [x{\leftarrow}y\langle\cdot\rangle][z{\leftarrow}\mathsf{I}])\langle z\rangle \to_{\mathsf{ond}} (xt, [x{\leftarrow}y\langle\cdot\rangle][z{\leftarrow}\mathsf{I}])\langle\mathsf{I}^{\alpha}\rangle = (xt, [x{\leftarrow}y\mathsf{I}^{\alpha}][z{\leftarrow}\mathsf{I}])$$

- *Weakness*: Since reduction in Open CbNeed is weak, then $(\lambda x.y, [y{\leftarrow}\mathsf{II}])$ is in normal form.
- *Two kinds of normal forms*: On the one hand, note that for any given value v and environment E, (v, E) is in Open CbNeed-normal form. This class of normal forms shall be called *abstraction programs*.
  On the other hand, the class of normal forms that shall be called *inert programs* do not have such a simple structure. For instance, note that if t contains multiplicative redexes, then program (xt, ϵ) is *not* in normal form; conversely, if t has no multiplicative redexes, then (xt, ϵ) is indeed an inert program. However, verifying the first coordinate of a program does not suffice to establish the Open CbNeed-normality of a program, since some free variable in it may be associated in the environment to a reduction step.
  For instance, note that no inert term i has multiplicative redexes, and so (i, ϵ) is always an Open CbNeed-normal form. However, if $x \in \mathsf{nv}(i)$, then $p = (i, [x{\leftarrow}\mathsf{II}])$ is *not* in Open CbNeed-normal form. In fact, we shall be able to write $i = \mathcal{H}\langle x\rangle$, which lets us rewrite p as $p = (\mathcal{H}\langle x\rangle, [x{\leftarrow}\langle\cdot\rangle])\langle\mathsf{II}\rangle$.
- *Partially useful*: As explained in the Introduction, only values are substituted. Hence, $(x, [x{\leftarrow}y])$ is in Open CbNeed-normal form, even if x is associated in the environment to a term without multiplicative redexes—namely y, which is morally a "normal" term.

Now that we have explored some intuitions on what the evaluation strategy should look like, let us proceed to define the (proper) subclass of program contexts used in the contextual closure definition of the Open CbNeed evaluation strategy. These contexts are defined via a parameterization by sets of variables, noted $\mathcal{V}$, $\mathcal{W}$, Σ. In other words, for each set of variables $\mathcal{V}$ we define the subclass of program contexts $\mathcal{E}_{\mathcal{V}}$ parameterized by it.

$$\frac{}{(\mathcal{H},\epsilon)\in\mathcal{E}_{\mathsf{nv}(\mathcal{H})}}\ \mathsf{O}_{\mathsf{AX}} \qquad \frac{P\in\mathcal{E}_{\mathcal{V}}\quad x\in\mathcal{V}}{P@[x{\leftarrow}i]\in\mathcal{E}_{(\mathcal{V}\setminus\{x\})\cup\mathsf{nv}(i)}}\ \mathsf{O}_{\mathsf{I}}$$

$$\frac{P\in\mathcal{E}_{\mathcal{V}}\quad x\notin\mathcal{V}}{P@[x{\leftarrow}t]\in\mathcal{E}_{\mathcal{V}}}\ \mathsf{O}_{\mathsf{GC}} \qquad \frac{P\in\mathcal{E}_{\mathcal{V}}\quad x\notin\mathcal{V}}{P\langle x\rangle@[x{\leftarrow}\mathcal{H}]\in\mathcal{E}_{\mathcal{V}\cup\mathsf{nv}(\mathcal{H})}}\ \mathsf{O}_{\mathsf{HER}}$$

Figure 6.1: Derivation rules for open evaluation contexts

Morally, these sets of variables allow us to inductively *restrict* the Λ-terms contained in the ESs. This is achieved by extending the notion of needed variables of inert terms and term contexts. Before discussing alternatives to the use of sets of variables, let us first present the derivation rules.

Definition 25 (Open evaluation contexts)**.**
The subclass of program contexts that we call *open evaluation context* is given by the derivation rules in Fig. 6.1.

For the sake of preciseness, let us say that our approach does not implement parameterization of sets of variables in the *exact same* fashion as appears in "Foundations of Strong Call byNeed' '[Bal+17]. Beyond the fact that ours is a split calculus, the difference lies in that we implement a somewhat *tighter* parameterization. That is, while there may be different sets of variables Φ such that $C\in E_{\Phi}$—adopting the terminology in [Bal+17]—there is only one set of variables $\mathcal{V}$ for every open evaluation context P in our definition such that $P\in\mathcal{E}_{\mathcal{V}}$. In fact, given a derivation of $C\in\mathcal{E}_{\Phi}$ from [Bal+17], there are infinitely many sets of variables $\Psi\supset\Phi$ such that also $C\in E_{\Psi}$, because sets of variables are *not necessarily minimal*. On the contrary, our derivation rules are parameterized by *minimal* sets of variables, and are unique.

Lemma 6.1.1 (Unique derivation parameterization of open evaluation contexts)**.**
Let $P\in\mathcal{E}_{\mathcal{V}}$ and $P\in\mathcal{E}_{\mathcal{W}}$. Then $\mathcal{V}=\mathcal{W}$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $P\in\mathcal{E}_{\mathcal{V}}$, proceeding by case analysis on the last applied derivation rule.

□

The (unique) set of variables parameterizing a given open evaluation context is called its *needed variables*. Much like the needed variables of terms, programs or term contexts, the concept of needed variables of open evaluation contexts is of the utmost importance in the extraction of *exact* quantitative information from type derivations in the Open CbNeed multi type system—as we shall see in Chapter 7 (Multi types for Open CbNeed).

Definition 26 (Needed variables of open evaluation contexts)**.**
Given an open evaluation context P derived as $P\in\mathcal{E}_{\mathcal{V}}$, the set of variables $\mathcal{V}$ are called the *needed variables* of P.

We finally have the needed elements to define the evaluation strategy.

Definition 27 (Open CbNeed evaluation strategy)**.**
Consider the following reduction relations

$$\begin{array}{rcll} P\langle(\lambda x.t)u\rangle & \to_{\mathsf{om}} & P\langle t, [x{\leftarrow}u]\rangle & \\ P\langle x\rangle & \to_{\mathsf{oe}} & P\langle v^{\alpha}\rangle & \text{if } P(x) = v \end{array}$$

where $v \in \mathsf{Val}$ in the definition of $\to_{\mathsf{oe}}$.

The Open CbNeed evaluation strategy $\to_{\mathsf{ond}}$ is defined as the union of $\to_{\mathsf{om}}$ and $\to_{\mathsf{oe}}$, which are called its *multiplicative* and *exponential* sub-relations, respectively. That is, given $p, q \in \mathcal{PR}$, we say that p reduces in the Open CbNeed evaluation strategy to q, and note it $p \to_{\mathsf{ond}} q$, if $p \to_{\mathsf{om}} q$ or if $p \to_{\mathsf{oe}} q$.

6.2 Characterizing Open CbNeed-normal forms

In order to prove the operational and semantical properties of Open CbNeed, it is helpful to have a concise, sound and complete predicate characterizing the set of programs in Open CbNeed-normal form, preferably in such a way that analyzing the structure of a Open CbNeed-normal form can be done by analyzing the inductive structure of its characterization.

Like we did while analyzing the weak and close evaluation strategies in Chapter 4 (CbN, CbV and CbNeed), we present here a *syntactic* characterization of Open CbNeed-normal forms in the form of predicate onorm, which is nothing but the union of other two (disjoint) predicates, inert and abs.

Definition 28 (The inert, abs and onorm predicates)**.**
Predicates inert and abs on $\mathcal{PR}$ are derived by the rules in Fig. 6.2.

$$\frac{}{\mathsf{inert}(i,\epsilon)}\ \mathsf{I}_{\mathsf{AX}} \qquad \frac{\mathsf{inert}(p) \quad x \in \mathsf{nv}(p)}{\mathsf{inert}(p@[x{\leftarrow}i])}\ \mathsf{I}_{\mathsf{l}} \qquad \frac{\mathsf{inert}(p) \quad x \notin \mathsf{nv}(p)}{\mathsf{inert}(p@[x{\leftarrow}t])}\ \mathsf{I}_{\mathsf{GC}}$$

$$\frac{}{\mathsf{abs}(v,\epsilon)}\ \mathsf{A}_{\mathsf{AX}} \qquad \frac{\mathsf{abs}(p)}{\mathsf{abs}(p@[x{\leftarrow}t])}\ \mathsf{A}_{\mathsf{GC}}$$

Figure 6.2: Predicates meant to characterize Open CbNeed-normal forms

Finally, predicate onorm on $\mathcal{PR}$ is defined as the union of inert and abs. That is, $\mathsf{onorm}(p)$ if and only if $\mathsf{inert}(p)$ or $\mathsf{abs}(p)$.

Let us point out a few details:

- Rule $\mathsf{I}_{\mathsf{Lift}}$ (resp. $\mathsf{A}_{\mathsf{Lift}}$) takes an inert term (resp. a value) and simply lifts/promotes it to the category of programs.
- Rule I_{l} requires the appended Λ-term to be inert, hence the notation i instead of a generic t.
- Rules I_{GC} and A_{GC} do not impose any constraint on the appended $t \in \Lambda$.

The intention is that the onorm predicate should characterize programs in Open CbNeed-normal form. Proving this is, however, quite involved. Firstly, let us give a basic property used pervasively in the study of Open CbNeed:

Lemma 6.2.1 (Redex in non-normal terms)**.**
Let $t \in \Lambda$. Then, t is not a normal term if and only if there exist term context $\mathcal{H}$, and terms $\lambda x.u$ and s such that $t = \mathcal{H}\langle(\lambda x.u)s\rangle$.

Proof. (Click here to see the complete proof in the Technical Appendix)
The $\Rightarrow$ direction is proven by structural induction on t, and proceeding by case analysis on the shape of t.
The $\Leftarrow$ direction is proven by structural induction on $\mathcal{H}$.

□

We now give the characterization of Open CbNeed-normal forms by means of predicate $\mathsf{onorm}(.)$. Its development is extremely subtle and we avoid it here altogether. The interested reader may refer to Sect. 13.3 (Proofs of Open CbNeed) in the Technical Appendix for further details—see in particular Subsect. 13.3.1 (Characterizing Open CbNeed-normal forms), where the proof of Proposition 6.2.2 is presented in its complete form.

Proposition 6.2.2 (Syntactic characterization of Open CbNeed-normal forms)**.**
Let $p \in \mathcal{PR}$. Then p is in $\rightarrow_{\mathsf{ond}}$-normal form if and only if $\mathsf{onorm}(p)$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the length of the environment of p.

□

We would like to stress the fact that Proposition 6.2.2 is of great help in the type-theoretical studies of the Open CbNeed evaluation strategy. The main reason is that we can then analyze the typing properties of programs in Open CbNeed-normal form simply proceeding by induction on the derivation rules of $\mathsf{onorm}(p)$.

6.3 Determinism

To prove determinism of Open CbNeed, we need to prove that for every $p \in \mathcal{PR}$, if $p = P_1\langle t_1\rangle = P_2\langle t_2\rangle$ for some $P_1 \in \mathcal{E}_{\mathcal{V}_1}$, $P_2 \in \mathcal{E}_{\mathcal{V}_2}$, $t_1, t_2 \in \Lambda$, and $p \rightarrow_{\mathsf{ond}}$-reduces, then $P_1 = P_2$ and $t_1 = t_2$.

While multiplicative redexes in Open CbNeed are simply given by the β-redex in the Λ-term plugged into the open evaluation context, note that exponential redexes are instead defined in terms of the variable occurrence to be substituted *and* the ES in the environment of the open evaluation context that binds that variable. Take, for example, $(x, [x{\leftarrow}\mathsf{I}]) = ((\langle\cdot\rangle, \epsilon)\langle x\rangle)@[x{\leftarrow}\mathsf{I}]$, which contains an exponential redex, but whose sub-program $(x, \epsilon) = (\langle\cdot\rangle, \epsilon)\langle x\rangle$ does not.

Therefore, the first thing to do to prove determinism of Open CbNeed consists in generalizing the notion of multiplicative and exponential redexes, via what are called *reduction places*, thus devising the kind of induction required in the proof of determinism.

Definition 29 (Reduction places in Open CbNeed)**.**
Let $t \in \Lambda$, $\mathcal{H}$ be a term context, and let $S \supseteq \mathsf{nv}(\mathcal{H})$. We say that t is a *S-reduction place* of $\mathcal{H}\langle t\rangle$ if one of the following conditions hold:

- *Multiplicative redex*: $t = (\lambda x.u)s$;
- *New needed variable*: $t = x$, $x \notin S$.

Let $t \in \Lambda$, $P \in \mathcal{E}_{\mathcal{V}}$ be an open evaluation context, and let $S \supseteq \mathcal{V}$. We say that t is a *S-reduction place* of $P\langle t\rangle$ if one of the following conditions hold:

- *Multiplicative redex*: $t = (\lambda x.u)s$;
- *Exponential redex*: $t = x$, $x \in \mathsf{dom}(P)$ and $P(x) = v$;
- *New needed variable*: $t = x$, $x \notin S$ and $x \notin \mathsf{dom}(P)$.

The notion of reduction place is enough to prove that given $t \in \Lambda$, we can single out the "*first*" multiplicative redex—if there is any—relative to the Open CbNeed evaluation strategy. We believe that this vague notion of a *first* multiplicative redex in the strategy—meant to serve as a guiding *intuition* to understanding determinism of Open CbNeed—becomes clearer when considering Λ-terms with a total order in their multiplicative redexes. For instance, if we consider $(...(xt_1)...t_{n-1})t_n \in \Lambda$, then we note that Open CbNeed proceeds to reduce $((...(xt_1)...t_{n-1})t_n, \epsilon)$ by reducing every t_i in a left-to-right fashion, first reducing t_1 to a normal Λ-term (if necessary), then reducing t_2 to a normal Λ-term (again, only if necessary), and so on until finally reducing t_n to a normal Λ-term. Thus, we can say that the "first" multiplicative redex in $((...(xt_1)...t_{n-1})t_n, \epsilon)$ is the smallest i such that t_i is not a normal term.

In this sense, the following Lemma implies that for every term context $\mathcal{H}$ and $(\lambda x.t)u \in \Lambda$, the Open CbNeed evaluation strategy reduces $(\mathcal{H}\langle(\lambda x.t)u\rangle, \epsilon)$ by contracting $(\lambda x.t)u$.

Lemma 6.3.1 (Unique decomposition of Λ-terms)**.**
Let $\mathcal{H}_1\langle t_1\rangle = \mathcal{H}_2\langle t_2\rangle$, with $\mathcal{H}_1, \mathcal{H}_2$ term contexts, let $S \supseteq (\mathsf{nv}(\mathcal{H}_1) \cup \mathsf{nv}(\mathcal{H}_2))$, and let t_i be a S-reduction place of $\mathcal{H}_i\langle t_i\rangle$, for $i = 1, 2$.
Then $t_1 = t_2$ and $\mathcal{H}_1 = \mathcal{H}_2$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on any one of the term contexts.

□

With Lemma 6.3.1 serving as the base case, we can now prove that there exists a "*first*" multiplicative or exponential redex in a program (provided it $\rightarrow_{\mathsf{ond}}$-reduces), in the sense that it is the only reduction step defined in the Open CbNeed reduction relation:

Theorem 6.3.2 (Unique decomposition of programs)**.**
Let $P_1\langle t_1\rangle = P_2\langle t_2\rangle$, with $P_1 \in \mathcal{E}_{\mathcal{V}_1}$, $P_2 \in \mathcal{E}_{\mathcal{V}_2}$, $S \supseteq (\mathcal{V}_1 \cup \mathcal{V}_2)$, and t_i be a S-reduction place of $P_i\langle t_i\rangle$ for $i = 1, 2$.
Then $t_1 = t_2$ and $P_1 = P_2$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on any one of the open evaluation contexts.

□

Finally,

Corollary 6.3.3 (Determinism of Open CbNeed)**.**
Let $p, q, r \in \mathcal{PR}$. If $p \rightarrow_{\mathsf{ond}} q$ and $p \rightarrow_{\mathsf{ond}} r$, then $q = r$.

Proof. (Click here to see the complete proof in the Technical Appendix)
Straightforward application of Theorem 6.3.2.

□

Chapter 7

Multi types for Open CbNeed

Introduction

We now provide a multi type system where typability characterizes Open CbNeed-termination and whose type derivations provide exact bounds on the normalization process. In addition, and unlike the weak and closed strategies in Chapter 5 (Multi types for CbN, CbV and CbNeed), type derivations of this multi type system also provide exact quantitative information on the *size* of Open CbNeed-normal forms—which, by the way, are of a non-trivial nature as observed in Chapter 6 (Open CbNeed).

Our starting point consists in adapting the multi type system given for (the weak and closed) CbNeed in Chapter 5 to a setting where characterization is given for programs instead of terms. Of course, properly adjusting the type system to the new evaluation strategy also requires several many tweaks, in several many different aspects, all of which have an impact on typability. Let us first present the Open CbNeed type system in its final form, and then give a list of general guidelines to understanding the rationale behind it.

7.1 Multi type system for Open CbNeed

Specific notions of linear and multi types are required for the Open CbNeed type system—as for the weak and closed ones in Chapter 5 (Multi types for CbN, CbV and CbNeed):

Definition 30 (Open CbNeed linear and multi types)**.**

Open CbNeed linear and multi types are defined mutually recursively as follows

Open CbNeed linear types	$L, L' ::= \mathsf{abs} \mid \mathsf{inert} \mid M \multimap N$
Open CbNeed multi types	$M, N ::= [L_i]_{i \in I},\ \ I$ a finite set
Tight types	$\mathsf{tight} ::= \mathsf{abs} \mid \mathsf{inert}$

We shall define the *tight* type derivations—a class of minimal type derivations—for the

Open CbNeed system via the tight constants. Hence, the following categories of multi types shall be extensively used hereafter:

Definition 31 (Multi types of tight constants)**.**

We write Inert for the class of *non-empty* multisets of inert. Similarly, we write Abs for the class of *non-empty* multisets of abs and we write Tight for the class of *non-empty* multisets of inert and abs.

Definition 32 (The Open CbNeed multi type system)**.**

The typing rules of the Open CbNeed type system appear in Fig. 7.1.

We write $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} \mathsf{e} : M$ to express that type derivation Φ in the Open CbNeed multi type system ends in type judgement $\Gamma \vdash^{(m,e,r)} \mathsf{e} : M$.

Let us provide now some general guidelines for the Open CbNeed type system:

- *A new index.* Besides the multiplicative and exponential indices in the CbNeed type system, the Open CbNeed system includes a third one accounting for the *size* of the normal form.
- *Axioms count exponential steps.* Every call-by-need evaluation strategy in the literature only performs substitutions on λ-abstractions—or similar syntactic objects—and Open CbNeed is not the exception: Definition 27 (Open CbNeed evaluation strategy) defines substitutions to act exclusively on λ-abstractions. Moreover, note that the only linear types derivable for λ-abstractions are abs and arrow types of the form $M \multimap N$, verifiable by the syntax-driven nature of typing rules abs and fun.
  In other words, no λ-abstraction is Inert-typable. This is why introducing a variable occurrence via the ax_I rule means that the index counting the $\to_{\mathsf{ond}}$-exponential steps to normal form—the second one—should be 0, as that particular variable occurrence cannot take part in a substitution of a λ-abstraction for it. Complementarily, introducing a variable occurrence via an ax rule means that the second index should be 1.
- *Splitting the constant.* Recall that in the (weak and closed) CbNeed type system, we relied on the use of a constant linear type norm to isolate a class of minimal type derivations—in that they provide exact bounds on the normalization process, instead of only upper bounds. This was enough in the weak and closed case because every CbNeed-normal form is an *answer* and hence they all have the same structure—see the explanation below Proposition 4.6.1, on page 50.
  But this no longer holds in the open case, a sign of this being the fact that we have two disjoint predicates to characterize Open CbNeed-normality—namely inert and abs—unlike the single norm predicate that characterizes all of Open CbNeed-normal forms. Hence, a first tweak to be made is splitting the constant norm from Chapter 5 (Multi types for CbN, CbV and CbNeed) in two new constant linear types, noted inert and abs. The intention behind these constants is that when a program is typed with inert (resp. abs) then the Open CbNeed-normal form of said program satisfies the inert predicate (resp. the abs predicate).
- *Typing inert applications.* All the systems in this work are designed to characterize termination by means of typability. Particularly regarding the CbNeed type system, note that certain sub-derivations allow some of the sub-terms not to be typed—consider *e.g.* the body of a λ-abstraction typed with typing rule norm in Fig. 5.4 (Type system for CbNeed evaluation); rules many, $\mathsf{app}_{\mathrm{gc}}$ and $\mathrm{ES}_{\mathrm{gc}}$ do not impose typability on *all* of the terms involved either.
  In the Open CbNeed type system, the minimality of certain type derivations for Open CbNeed-normal forms is partly achieved by forcing *all* sub-terms of inert programs to be typed. In particular, this means typing both sub-terms involved in an application of an inert Λ-term: if we are to type an application of the form $i\,n$, with i an inert term and n a normal term, in such a way that the type derivation provides the exact size of $i\,n$, then we need to add a

$$\frac{M \in \mathsf{Inert}}{x : M \vdash^{(0,0,0)} x : M}\ \mathsf{ax}_I \qquad \frac{M \notin \mathsf{Inert} \quad M \neq \mathbf{0}}{x : M \vdash^{(0,1,0)} x : M}\ \mathsf{ax}$$

$$\frac{\Gamma; x :\ N \vdash^{(m,e,r)} t : M}{\Gamma \vdash^{(m,e,r)} \lambda x.t : N \multimap M}\ \mathsf{fun} \qquad \frac{}{\vdash^{(0,0,0)} \lambda x.t : \mathsf{abs}}\ \mathsf{abs}$$

$$\frac{(\Gamma_i \vdash^{(m_i,e_i,r_i)} \lambda x.t : L_i)_{i \in I}}{\biguplus_{i \in I} \Gamma_i \vdash^{(\sum_{i \in I} m_i, \sum_{i \in I} e_i, \sum_{i \in I} r_i)} \lambda x.t : [L_i]_{i \in I}}\ \mathsf{many}$$

$$\frac{\Gamma \vdash^{(m,e,r)} t : [N \multimap M] \quad \Pi \vdash^{(m',e',r')} u : N \quad N \neq \mathbf{0}}{\Gamma \uplus \Pi \vdash^{(m+m'+1, e+e', r+r')} tu : M}\ \mathsf{app}$$

$$\frac{\Gamma \vdash^{(m,e,r)} t : [\mathsf{inert}]_{j \in J} \quad \Pi \vdash^{(m',e',r')} u : [\mathsf{tight}] \quad J \neq \emptyset}{\Gamma \uplus \Pi \vdash^{(m+m', e+e', r+r'+1)} tu : [\mathsf{inert}]_{j \in J}}\ \mathsf{app}_i$$

$$\frac{\Gamma \vdash^{(m,e,r)} t : [\mathbf{0} \multimap M]}{\Gamma \vdash^{(m+1,e,r)} tu : M}\ \mathsf{app}_{\mathrm{gc}}$$

$$\frac{\Gamma \vdash^{(m,e,r)} t : M}{\Gamma \vdash^{(m,e,r)} (t, \epsilon) : M}\ \mathsf{Lift}$$

$$\frac{\Gamma; x :\ N \vdash^{(m,e,r)} (t, E) : M \quad \Pi \vdash^{(m',e',r')} u : N \quad N \neq \mathbf{0}}{\Gamma + \Pi \vdash^{(m+m', e+e', r+r')} (t, E[x \leftarrow u]) : M}\ \mathrm{ES}$$

$$\frac{\Gamma \vdash^{(m,e,r)} (t, E) : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} (t, E[x \leftarrow u]) : M}\ \mathrm{ES}_{\mathrm{gc}}$$

Figure 7.1: Type system for Open CbNeed evaluation

new typing rule whose premises are the typing of i with inert and the typing of n with inert or abs—see typing rule app_i in Fig. 7.1.

- *Lifting typability of Λ-terms to programs.* Since we no longer have a single-classed syntax on which the reduction relation is defined, we have specific rules for Λ-terms and specific rules for programs, as well as a Lift rule for lifting the former category to the latter. Unsurprisingly, typing rules ES and $\mathrm{ES}_{\mathrm{gc}}$ only have programs as subject of the judgement, because only programs have an environment to append ESs to.
- *Characterizing a class of minimal type derivations.* Unlike the weak and closed cases, isolating a class of minimal type derivations in the Open CbNeed case must involve some type derivations whose type contexts are non-empty. This is due to the fact that every applicant of an inert term must, as previously noted, be typed either with inert or abs, one of whose direct consequences is that the *head* variable of inert terms are *always* typed.
  For example, consider $((xy)z, \epsilon)$, which is in Open CbNeed-normal form and satisfies that $\mathsf{inert}((xy)z, \epsilon)$. Since both x and xy need to be typed with inert in order to be able to type xy and $(xy)z$, respectively, then a minimal type derivation of $((xy)z, \epsilon)$ must *at least* include x in the domain of its type context.
  We give a proper definition isolating a class of minimal type derivations in the Open CbNeed type system in due time, extending the intuitions just given to programs with a non-trivial environment. Nonetheless, we would like to refer the reader to [AGK20], in particular to the semantical studies of an evaluation strategy that the authors call *Linear Head evaluation*. Therein, every type derivation for a *neutral* Λ_{L}-term—a kind of term playing a role analogous to the one played by inert Λ-terms here—must include the head variable of the neutral term in the domain of the type context. This fact regarding Linear Head evaluation has, in fact, been our starting point to understanding what type contexts should be like in order to isolate a class of minimal type derivations for the Open CbNeed type system.

In order to relate the third index in type judgements of the Open CbNeed type system to the size of Open CbNeed-normal forms, we first need to provide a meaningful notion of size of programs. The following serves our purposes:

Definition 33 (Needed size of terms and programs)**.**

Let us first define the *base* case for the *needed size* of programs—namely, the needed size of Λ-terms:

$$\begin{aligned} |x|_{\mathsf{nd}} &:= 0 \\ |\lambda x.t|_{\mathsf{nd}} &:= 0 \\ |t\,u|_{\mathsf{nd}} &:= 1 + |t|_{\mathsf{nd}} + |u|_{\mathsf{nd}} \end{aligned}$$

The needed size of programs is then defined as follows:

$$|(t, \epsilon)|_{\mathsf{nd}} := |t|_{\mathsf{nd}}$$

$$|(t, E\,[x{\leftarrow}u])|_{\mathsf{nd}} := \begin{cases} |(t, E)|_{\mathsf{nd}} + |u|_{\mathsf{nd}} & \text{, if } x \in \mathsf{nv}(t, E) \\ |(t, E)|_{\mathsf{nd}} & \text{, otherwise} \end{cases}$$

Let us isolate a class of type derivations meant to provide exact quantitative information. Namely, the class of

Definition 34 (Tight derivations for Open CbNeed)**.**

A derivation $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} \mathsf{e} : M$ is *tight* if $M = [\mathsf{tight}]$ and $\Gamma(x) \in \mathsf{Inert}$ for every $x \in \mathsf{dom}(\Gamma)$.

We could, in principle, extend the definition of tight type derivations to additionally encompass those whose type contexts assign any kind of Tight types. This would seem to be a more natural definition, in particular because domains of type contexts may be thought of as the (types) interface between the typed expression and (potential) other computational components. That is, a definition of tight type derivations for the Open CbNeed system where type contexts can assign *any* kind of Tight types—in the form of unrestricted combinations of inert's and abs's—would imply a higher degree of compositionality between tight type derivations.

For example, in the derivation of Φ below, note that Ψ is not a tight type derivation in the Open CbNeed sense—in particular because the type context assigns y to $[\mathsf{abs}]$:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x : [\mathsf{inert}] \vdash^{(0,0,0)} x : [\mathsf{inert}]}\,\mathsf{ax}_I \quad \dfrac{}{y : [\mathsf{abs}] \vdash^{(0,1,0)} y : [\mathsf{abs}]}\,\mathsf{ax}_I}{x : [\mathsf{inert}]; y : [\mathsf{abs}] \vdash^{(0,1,1)} xy : [\mathsf{inert}]}\,\mathsf{app}_i}{\Psi \rhd\ x : [\mathsf{inert}]; y : [\mathsf{abs}] \vdash^{(0,1,1)} (xy, \epsilon) : [\mathsf{inert}]}\,\mathsf{Lift} \quad \dfrac{\dfrac{}{\emptyset \vdash^{(0,0,0)} v : \mathsf{abs}}\,\mathsf{abs}}{\Theta \rhd\ \emptyset \vdash^{(0,0,0)} v : [\mathsf{abs}]}\,\mathsf{many}}{\Phi \rhd\ x : [\mathsf{inert}]; y : [\mathsf{abs}] \vdash^{(0,1,1)} (xy, [y \leftarrow v]) : [\mathsf{inert}]}\,\mathsf{ES}$$

Note that if we extended the definition of tight type derivation as just proposed, then Ψ would be tight and Φ would be obtained by composing two tight type derivations, namely Ψ and Θ.

However convenient having such an extended notion of tight type derivations may seem, building the alternate theory requires intricate and delicate technical developments that are just not natural to this formulation of Open CbNeed. It is certainly possible to do so, but the outcome does not justify the considerable increase in technical effort.

In fact, what Open CbNeed is missing in the current formulation, to be suitable for an extended notion of tight type derivations, is the study of *applied* and *unapplied* needed variables of an expression. Roughly, the distinction provided by these categories of needed variables allows us to tell precisely which variables can be safely and unrestrictedly assigned by type contexts to a Tight type—while maintaining the minimality of the type derivation—and which variables cannot.

We study these issues in depth in Chapter 9 (Multi types for Useful Open CbNeed), where the concepts of useful substitutions and the distinction between applied and unapplied needed variables are naturally linked.

Having explained why tight type derivations in the Open CbNeed case are limited in the way they are, we now turn to a relaxation on the definition of tight type derivations that proves to be immensely useful for many inductions to go through in the technical development:

Definition 35 (Inert restriction of type contexts)**.**

Given a type context Γ, we say that the restriction of Γ to some set of variables $S \subseteq \mathsf{Var}$ is inert, and note it $\mathsf{inert}_\Gamma(S)$, when every variable in $S \cap \mathsf{dom}(\Gamma)$ is assigned by Γ to some Inert type.

That is, instead of checking if a *whole* type context assigns nothing but Inert types to the variables in its domain, it sometimes is enough to check if it does so with respect to a particular subset of its domain. The reason why this works is that assigning some of the variables of an expression to Inert types forces the rest of the type context to be minimal—*i.e.*, to be tight. This technique is pervasive in our study of Open CbNeed semantics, although the particular subset of variables that we check type contexts against is determined by the different technical tools we require in the development of the theory.

7.1.1 Open CbNeed correctness

The study of Correctness for Open CbNeed is structured just like for the previous systems. Let us overview it, pointing out the intrinsic particularities that distinguish this case from the others.

First, a property common to all our systems

Lemma 7.1.1 (Relevance of the Open CbNeed type system)**.**
Let e *be an expression and* $\Phi \triangleright_O \Gamma \vdash^{(m,e,r)} \mathsf{e} : M$ *be a type derivation. If* $x \notin \mathsf{fv}(\mathsf{e})$ *then* $x \notin \mathsf{dom}(\Gamma)$*.*

Next, we show that *tight* type derivations—which are meant to be simultaneously minimal in each of the indices—are as expected. For this purpose, we have extracted the following base case

Lemma 7.1.2 (Typing properties of normal terms)**.**
1. Values*: Let* $\Phi \triangleright_O \Gamma \vdash^{(m,e,r)} v : M$*. If* $M = [\mathsf{tight}]$*, then* $\mathsf{dom}(\Gamma) = \mathsf{nv}(v)$ *and* $(m, e, r) = (0, 0, |t|_{\mathsf{nd}})$*.*
2. Inert terms*: Let* $\Phi \triangleright_O \Gamma \vdash^{(m,e,r)} i : M$*, with* $\mathsf{inert}_\Gamma(\mathsf{nv}(i))$*. Then* $M = [\mathsf{inert}]_{i \in I}$ *with* $I \neq \emptyset$*,* $\mathsf{dom}(\Gamma) = \mathsf{nv}(i)$ *and* $(m, e, r) = (0, 0, |i|_{\mathsf{nd}})$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)
Point (1) is trivial—given the trivial structure of values in this weak setting—while Point (2) is proven by structural induction on i. □

Note that Lemma 7.1.2.2 makes reference to the domain of the type context, stating that it matches exactly the set of needed variables of inert term i. Remarkably, this is obtained by *only* making the restricted assumption that $\mathsf{inert}_\Gamma(\mathsf{nv}(i))$, and may be extended to programs, as shown below.

Proposition 7.1.3 (Typing properties of Open CbNeed-normal forms)**.**
Let $p \in \mathcal{PR}$ *be such that* $\mathsf{onorm}(p)$*, and let* $\Phi \triangleright_O \Gamma \vdash^{(m,e,r)} p : M$ *be a tight type derivation for it. Then* $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$ *and* $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)
The proof is split into $\mathsf{abs}(p)$ and $\mathsf{inert}(p)$, proving the following statement—which together imply the original one:

1. If $\mathsf{abs}(p)$ and $M = [\mathsf{tight}]$, then $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$ and $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$.
2. If $\mathsf{inert}(p)$ and $\mathsf{inert}_\Gamma(\mathsf{nv}(p))$, then $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$ and $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$.

Both items are proven by induction on the derivation of the corresponding predicate.

□

The usual substitution lemma requires to be proven in two separate parts, as we first need to prove it for term contexts before being able to prove it for open evaluation contexts—the latter building on the former.

Moreover, this lemma requires a key extra condition to hold, namely that the restriction of the type context to a certain set of variables is inert. The set of variables to check the type context against depends on the kind of contexts we are proving the lemma for: in the case of a term context we require the restriction of the type context to the needed variables of said term context to be inert, whereas in the case of an open evaluation context we require the restriction of the type context to

the set of variables parameterizing the derivation of said open evaluation context to be inert—note that this is independent of the particular derivation of the open evaluation context, by Lemma 6.1.1 (Unique derivation parameterization of open evaluation contexts).

Lemma 7.1.4 (Linear Substitution for Open CbNeed)**.**
Let $x \in \mathsf{Var}$ and $v \in \mathsf{Val}$ such that $x \notin \mathsf{fv}(v)$.

1. *Let $\mathcal{H}$ be such that $x \notin \mathsf{nv}(\mathcal{H})$, and let*

$$\Phi_{\mathcal{H}\langle x\rangle} \rhd_O \Gamma; x : M \vdash^{(m,e,r)} \mathcal{H}\langle x\rangle : N$$

be such that $M, N \neq \mathbf{0}$ and $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$.
Then there exists splitting $M = M_1 \uplus M_2$, with $M_1 \neq \mathbf{0}$, such that for every

$$\Psi \rhd_O \Pi \vdash^{(m',e',r')} v : M_1$$

there exists

$$\Phi_{\mathcal{H}\langle v\rangle} \rhd_O \left(\Gamma \biguplus \Pi\right); x : M_2 \vdash^{(m+m',e+e'-1,r+r')} \mathcal{H}\langle v\rangle : N$$

2. *Let $P \in \mathcal{E}_\mathcal{V}$ be such that $x \notin \mathsf{dom}(P)$ and $x \notin \mathcal{V}$, and let*

$$\Phi_{P\langle x\rangle} \rhd_O \Gamma; x : M \vdash^{(m,e,r)} P\langle x\rangle : N$$

be such that $M, N \neq \mathbf{0}$ and $\mathsf{inert}_\Gamma(\mathcal{V})$.
Then there exists splitting $M = M_1 \uplus M_2$, with $M_1 \neq \mathbf{0}$, such that for every type derivation

$$\Psi \rhd_O \Pi \vdash^{(m',e',r')} v : M_1$$

there exists

$$\Phi_{P\langle v\rangle} \rhd_O \left(\Gamma \biguplus \Pi\right); x : M_2 \vdash^{(m+m',e+e'-1,r+r')} P\langle v\rangle : N$$

Proof. (Click here to see the complete proof in the Technical Appendix)

1. By structural induction on $\mathcal{H}$.
2. By structural induction on P.

□

In turn, the usual Subject Reduction property also needs to be split in two: one proof for term contexts and another one for open evaluation contexts—the latter building on the former. This is due to the fact that multiplicative redexes always take place, at the base level, inside a term context—as one can infer from a single inspection of the derivation rules for open evaluation contexts.

As what happened in the term context case in Lemma 7.1.4 (Linear Substitution for Open CbNeed), we require the restriction of the type context to the needed variables of the term context in question to be inert.

Lemma 7.1.5 (Quantitative Subject Reduction for $\to_{\mathsf{om}}$ in term contexts)**.**
Let $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} \mathcal{H}\langle(\lambda x.u)s\rangle : M$, with $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$. Then $m \geq 1$ and there exists $\Phi' \rhd_O \Gamma \vdash^{(m-1,e,r)} (\mathcal{H}\langle u\rangle, [x \leftarrow s]) : M$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on $\mathcal{H}$.

□

Proposition 7.1.6 (Quantitative Subject Reduction for Open CbNeed)**.**
Let $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} p : M$ *be a* tight *type derivation.*

1. Multiplicative*: If* $p \to_{\mathsf{om}} p'$*, then* $m \geq 1$ *and there exists a type derivation* $\Phi' \rhd_O \Gamma \vdash^{(m-1,e,r)} p' : M$*.*
2. Exponential*: If* $p \to_{\mathsf{oe}} p'$*, then* $e \geq 1$ *and there exists a type derivation* $\Phi' \rhd_O \Gamma \vdash^{(m,e-1,r)} p' : M$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)

The multiplicative case is proven by induction on $P \in \mathcal{E}_{\mathcal{V}}$ such that $p = P\langle s\rangle \to_{\mathsf{om}} P\langle s'\rangle = p'$, using Lemma 7.1.5 (Quantitative Subject reduction for $\to_{\mathsf{om}}$ in term contexts) for the base case.

The exponential case is proven by induction on $P \in \mathcal{E}_{\mathcal{V}}$ such that $p = P\langle x\rangle \to_{\mathsf{oe}} P\langle v^{\alpha}\rangle = p'$ and $p(x) = v$, using Lemma 7.1.4 (Linear Substitution for Open CbNeed) for the base case.

□

Conditions for the Subject Reduction property. In the complete proof of Proposition 7.1.6 given in the Technical Appendix on page 239, the reader may notice that the statement is relaxed by only assuming that the restriction of the type context to a certain set of variables is inert, while completely disregarding the right-hand side type. This relaxed statement allows the inductive hypothesis to smoothly go through, and coincides with the way previous properties are stated.

We would like to stress the fact that, rather than being a minor and inconsequential technical detail, this is an important result obtained in this work, as explained in Subsect. 3.7.1 (Axes-based analysis of the multi type systems) when discussing how the reduction depth of the evaluation strategies impacts the multi type system and its properties.

Indeed, in the Open CbNeed case, if the type context is not tight, then the type derivation provided by Subject Reduction may not see its multiplicative (resp. exponential) index decrease as expected after a multiplicative step (resp. after an exponential step). Nevertheless, we claim that a non-quantitative—or *qualitative*—version of Proposition 7.1.6 holds, although we do not provide a proof here.

Let us give a concrete example in which a non-quantitative version of the Subject Reduction property is satisfied, while absolutely neglecting the quantitative information appearing in the indices. Let $p = (x((\lambda y.y)(\lambda z.z)), \epsilon) \in \mathcal{PR}$. Note that p reduces to its Open CbNeed-normal form as follows

$$p = (x((\lambda y.y)(\lambda z.z)), \epsilon) \to_{\mathsf{om}} (xy, [y{\leftarrow}\lambda z.z]) \to_{\mathsf{oe}} (x(\lambda z'.z'), [y{\leftarrow}\lambda z.z])$$

Hence, let us define $q \coloneqq (xy, [y{\leftarrow}\lambda z.z])$ and $r \coloneqq (x(\lambda z'.z'), [y{\leftarrow}\lambda z.z])$, and give type derivations for all three programs, in such a way that the type context and right-hand side type are preserved but whose indices are not modified by reduction. First, let Φ_p be a type derivation for p of the following form:

$$\dfrac{\dfrac{\dfrac{}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(0,1,0)} x : [\mathbf{0} \multimap \mathsf{inert}]}\,\mathsf{ax}}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(1,1,0)} x((\lambda y.y)(\lambda z.z)) : [\mathsf{inert}]}\,\mathsf{app}_{\mathrm{gc}}}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(1,1,0)} (x((\lambda y.y)(\lambda z.z)), \epsilon) : [\mathsf{inert}]}\,\mathsf{Lift}$$

Next, let Φ_q be a type derivation for q of the following form:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(0,1,0)} x : [\mathbf{0} \multimap \mathsf{inert}]}\,\mathsf{ax}}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(1,1,0)} xy : [\mathsf{inert}]}\,\mathsf{app}_{\mathrm{gc}}}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(1,1,0)} (xy, \epsilon) : [\mathsf{inert}]}\,\mathsf{Lift}}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(1,1,0)} (xy, [y \leftarrow \lambda z.z]) : [\mathsf{inert}]}\,\mathrm{ES}_{\mathrm{gc}}$$

Finally, let Φ_r be a type derivation for r of the following form:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(0,1,0)} x : [\mathbf{0} \multimap \mathsf{inert}]}\,\mathsf{ax}}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(1,1,0)} x(\lambda z'.z') : [\mathsf{inert}]}\,\mathsf{app}_{\mathrm{gc}}}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(1,1,0)} (x(\lambda z'.z'), \epsilon) : [\mathsf{inert}]}\,\mathsf{Lift}}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(1,1,0)} (x(\lambda z'.z'), [y \leftarrow \lambda z.z]) : [\mathsf{inert}]}\,\mathrm{ES}_{\mathrm{gc}}$$

As we can appreciate, not only do the multiplicative and exponential indices not get updated as expected upon reduction of the subject, but also the size index does not even provide an upper bound to the size of the Open CbNeed-normal form: note that $|(x(\lambda z'.z'), [y \leftarrow \lambda z.z])|_{\mathsf{nd}} = 1$.

Even more problematic, note that $(\lambda y.y)(\lambda z.z)$ is not typed in the type derivation Φ for p, because x is applied to $(\lambda y.y)(\lambda z.z)$ via an application of the $\mathsf{app}_{\mathrm{gc}}$ rule. This means that we could obtain a similar result for any given term applied to x, even the diverging term Ω, as shown in:

$$\dfrac{\dfrac{\dfrac{}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(0,1,0)} x : [\mathbf{0} \multimap \mathsf{inert}]}\,\mathsf{ax}}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(1,1,0)} x\,\Omega : [\mathsf{inert}]}\,\mathsf{app}_{\mathrm{gc}}}{x : [\mathbf{0} \multimap [\mathsf{inert}]] \vdash^{(1,1,0)} (x\,\Omega, \epsilon) : [\mathsf{inert}]}\,\mathsf{Lift}$$

noting that $(x\,\Omega, \epsilon)$ does not normalize in Open CbNeed. Needless to say, this is completely wrong from a Correctness point of view.

Therefore, the Subject Reduction property in the Open CbNeed case *only* holds for tight type derivations, thus making correctness also *only* hold for tight type derivations.

Finally,

Theorem 7.1.7 (Tight Correctness for Open CbNeed)**.**

Let $p \in \mathcal{PR}$ and $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} p : M$ be a tight type derivation. Then there exists $q \in \mathcal{PR}$ such that

1. *q is in $\to_{\mathsf{ond}}$-normal form,*
2. *there exists a reduction sequence $d : p \to^*_{\mathsf{ond}} q$,*
3. *$(m, e, r) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})$, and*
4. *$\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$.*

Proof. (Click here to see the complete proof in the Technical Appendix)

By induction on the sum $m + e$, and proceeding by case analysis on whether $p \to_{\mathsf{ond}}$-reduces or not:

- If p is in $\to_{\mathsf{ond}}$-normal form, then we only need to prove the case where Φ is *tight*, which follows by Proposition 7.1.3 (Typing properties of Open CbNeed-normal forms).

- If $p \to_{\mathsf{ond}} q$, then the statement follows by first applying Proposition 7.1.6 (Quantitative Subject Reduction for Open CbNeed) on Φ—distinguishing whether $p \to_{\mathsf{ond}} q$ is a multiplicative or exponential step—thus obtaining a type derivation Ψ for q. Note that, by what is given in Proposition 7.1.6, the sum of multiplicative and exponential indices in Ψ are strictly smaller than $m + e$. Thus, we may finally apply the *i.h.* on Ψ to prove the statement.

□

7.1.2 Open CbNeed completeness

Particular technicalities aside, most features of the completeness section for Open CbNeed are either already present in Sect. 5.4 (Multi type system for Open CbNeed) or introduced and explained in the correctness section above. We therefore simply sketch out the general proof schema of completeness for Open CbNeed and deal with the technical details in the Appendix.

Somewhat similar to what happens for Correctness, proving that programs in Open CbNeed-normal form are (tightly) typable requires first to prove the following base case regarding (normal) Λ-terms. As in the correctness section, note that the domain of the type context matches exactly the set of needed variables of the term.

Lemma 7.1.8 (Tight typability of normal terms)**.**

1. Values*: For every* $v \in \mathsf{Val}$ *there exists a type derivation* $\Phi \rhd_O \Gamma \vdash^{(0,0,|v|_{\mathsf{nd}})} v : [\mathsf{abs}]$ *such that* $\mathsf{dom}(\Gamma) = \mathsf{nv}(v)$.
2. Inert terms*: For every inert* Λ*-term* i *and* $J \neq \emptyset$*, there exists a tight type derivation* $\Phi \rhd_O \Gamma \vdash^{(0,0,|i|_{\mathsf{nd}})} i : [\mathsf{inert}]_{j \in J}$ *such that* $\mathsf{dom}(\Gamma) = \mathsf{nv}(i)$.

Proof. (Click here to see the complete proof in the Technical Appendix)

1. Trivial.
2. By structural induction on i.

□

We can now use Lemma 7.1.8 as base case for the following

Proposition 7.1.9 (Tight typability of Open CbNeed-normal forms)**.**

Let $p \in \mathcal{PR}$ *be such that* $\mathsf{onorm}(p)$*. Then there exists a tight type derivation* $\Phi \rhd_O \Gamma \vdash^{(0,0,|p|_{\mathsf{nd}})} p : M$ *such that* $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.

Proof. (Click here to see the complete proof in the Technical Appendix)

Note that either $\mathsf{abs}(p)$ or $\mathsf{inert}(p)$. The statement is proven by induction on the derivation of said predicates, proceeding by case analysis on the last derivation rule in each case.

□

The removal Lemma takes the expected form, somewhat allowing us to revert substitution in the exact way required within the proof of the Subject Expansion property, taking into consideration every technicality appearing therein.

Lemma 7.1.10 (Linear Removal for Open CbNeed)**.**

Let $x \in \mathsf{Var}$ *and* $v \in \mathsf{Val}$ *such that* $x \notin \mathsf{fv}(v)$.

1. *Let* $\mathcal{H}$ *be such that* $x \notin \mathsf{nv}(\mathcal{H})$*, and let*

$$\Phi \rhd_O \Gamma; x : M \vdash^{(m,e,r)} \mathcal{H}\langle v \rangle : N$$

be such that $N \neq \mathbf{0}$*, and* $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$*.*
Then there exist type derivations

$$\Psi \rhd_O \Pi \vdash^{(m',e',r')} v : O$$
$$\Theta \rhd_O \Delta; x : (M \uplus O) \vdash^{(m'',e'',r'')} \mathcal{H}\langle x \rangle : N$$

such that
   - $\Gamma = \Pi \biguplus \Delta$*, and*
   - $(m' + m'', e' + e'', r' + r'') = (m, e + 1, r)$*.*

2. *Let* $P \in \mathcal{E}_\mathcal{V}$ *be such that* $x \notin (\mathcal{V} \cup \mathsf{dom}(P))$ *and that* $\mathsf{fv}(v) \cap \mathsf{dom}(P) = \emptyset$*. Let, moreover,*

$$\Phi \rhd_O \Gamma; x : M \vdash^{(m,e,r)} P\langle v \rangle : N$$

be such that $N \neq \mathbf{0}$*, and* $\mathsf{inert}_\Gamma(\mathcal{V})$*.*
Then there exist
   - *multi type* O*,*
   - *type derivation* $\Psi \rhd_O \Pi \vdash^{(m',e',r')} v : O$*, and*
   - *type derivation* $\Theta \rhd_O \Delta; x : (M \uplus O) \vdash^{(m'',e'',r'')} P\langle x \rangle : N$

such that
   - $\Gamma = \Pi \biguplus \Delta$*, and*
   - $(m' + m'', e' + e'', r' + r'') = (m, e + 1, r)$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)
1. By structural induction on the term context $\mathcal{H}$.
2. By structural induction on the open evaluation context $P \in \mathcal{E}_\mathcal{V}$.

□

The Subject Expansion property requires no extra explanation with respect to its correctness counterpart. Even the conditions assumed for Proposition 7.1.6 (Quantitative Subject Reduction for Open CbNeed)—analyzed on page 93—need to hold for the Subject Expansion property as well, as we can appreciate:

Lemma 7.1.11 (Quantitative Subject Expansion for $\to_{\mathsf{om}}$ in term contexts)**.**
Let $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} (\mathcal{H}\langle u \rangle, [x \leftarrow s]) : M$ *such that* $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$*. Then there exists* $\Phi' \rhd_O \Gamma \vdash^{(m+1,e,r)} \mathcal{H}\langle (\lambda x.u)s \rangle : M$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on $\mathcal{H}$.

□

Proposition 7.1.12 (Quantitative Subject Expansion for Open CbNeed)**.**
Let $\Phi' \rhd_O \Gamma \vdash^{(m,e,r)} p' : M$ *be a tight type derivation.*
1. Multiplicative*: If* $p \to_{\mathsf{om}} p'$*, then there exists a type derivation* $\Phi \rhd_O \Gamma \vdash^{(m+1,e,r)} p : M$*.*
2. Exponential*: If* $p \to_{\mathsf{oe}} p'$*, then there exists a type derivation* $\Phi \rhd_O \Gamma \vdash^{(m,e+1,r)} p : M$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)

The multiplicative case is proven by induction on $P \in \mathcal{E}_{\mathcal{V}}$ such that $p = P\langle s\rangle \to_{\mathsf{om}} P\langle s'\rangle = p'$, using Lemma 7.1.11 (Quantitative Subject Expansion for $\to_{\mathsf{om}}$ in term contexts) for the base case.

The exponential case is proven by induction on $P \in \mathcal{E}_{\mathcal{V}}$ such that $p = P\langle x\rangle \to_{\mathsf{oe}} P\langle v^{\alpha}\rangle = p'$ and $p(x) = v$, using Lemma 7.1.10 (Linear Removal for Open CbNeed) for the base case.

□

Finally,

Theorem 7.1.13 (Tight Completeness for Open CbNeed)**.**

Let $p \in \mathcal{PR}$. If there exists $d : p \to^{*}_{\mathsf{ond}} q$ for some $q \in \mathcal{PR}$ in $\to_{\mathsf{ond}}$-normal form, then there exists a tight type derivation $\Phi \rhd_O \Gamma \vdash^{(|d|_{\mathsf{m}},|d|_{\mathsf{e}},|q|_{\mathsf{nd}})} p : M$ such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$.

Proof. (Click here to see the complete proof in the Technical Appendix)

By induction on the length of the $\to_{\mathsf{ond}}$-normalizing sequence starting in p:

- *Base case*: By Proposition 6.2.2 (Syntactic characterization of closed normal forms - CbNeed), we have that $\mathsf{onorm}(p)$. We then get Φ by application of Proposition 7.1.9 (Tight typability of Open CbNeed-normal forms) on p.
- *Inductive case*: Let $p \to_{\mathsf{ond}} r$. By *i.h.*, we get the expected type derivation for r. We may then obtain the one for p by an application of Proposition 7.1.12 (Quantitative Subject Expansion for Open CbNeed), taking into account the kind of reduction step in $p \to_{\mathsf{ond}} r$—*i.e.*, whether it is exponential or multiplicative.

□

7.1.3 Open CbNeed semantics

First, let us adapt the notion of a suitable list of variables to the split LSC. Let $p \in \mathcal{PR}$ and $x_1, \ldots, x_n$ (with $n \geq 0$) be pairwise distinct variables. If $\mathsf{fv}(p) \subseteq \{x_1, \ldots, x_n\}$, we say that the list $\vec{x} = (x_1, \ldots, x_n)$ is *suitable for* p.

Now, following the analyses for the weak and closed evaluation strategies, we might be tempted to define the semantics of p for $\vec{x}$ with respect to the Open CbNeed type system as

$$[\![t]\!]^{\text{Open CbNeed}}_{\vec{x}} := \{((M_1, \ldots, M_n), M) \mid \exists\, \Phi \rhd_O x_1 : M_n; \ldots; x_n : M_n \vdash^{(m,e,r)} p : M\} \tag{7.1}$$

However, this semantics is not adequate. In particular, note that $(x\,\Omega, \epsilon)$ is not $\to_{\mathsf{ond}}$-normalizable but it is typable in the Open CbNeed system, as proven by the following type derivation:

$$\dfrac{\dfrac{\dfrac{}{x : [\mathbf{0} \multimap M] \vdash^{(0,1,0)} x : [\mathbf{0} \multimap M]}\,\mathsf{ax}}{x : [\mathbf{0} \multimap M] \vdash^{(0,1,1)} x\,\Omega : M}\,\mathsf{app}_{\mathrm{gc}}}{x : [\mathbf{0} \multimap M] \vdash^{(0,1,1)} (x\,\Omega, \epsilon) : M}\,\mathsf{Lift}$$

On the bright side, the definition in 7.1 is *invariant*: it suffices to prove a non-quantitative version of the Subject Reduction and Subject Expansion properties. We do not prove this here, as we are mostly interested in defining a semantics that is both invariant and adequate.

Therefore, let us redefine the *semantics of p for $\vec{x}$* with respect to the Open CbNeed type system as

$$[\![t]\!]^{\text{Open CbNeed}}_{\vec{x}} := \{((M_1, \ldots, M_n), M) \mid \exists\, \Phi \rhd_O x_1 : M_n; \ldots; x_n : M_n \vdash^{(m,e,r)} p : M, \text{ with } \Phi \textit{ tight}\}$$

This restricted semantics is indeed adequate, by Theorem 7.1.7 (Tight Correctness for Open CbNeed) and Theorem 7.1.13 (Tight Completeness for Open CbNeed). However, it comes at the price of breaking compositionality, as explained in Sect. 3.2 (Properties of multi type systems, page 28).

7.2 Counting techniques for exponential steps

Let us provide justifications for the choice we have made regarding axioms in the Open CbNeed system, namely splitting them between ax and ax_I, which stems from a combination of specific operational features of the Open CbNeed evaluation strategy.

Originally, our main sources of inspiration were Accattoli, Graham-Lengrand and Kesner's [AGK20] and Accattoli, Guerrieri and Leberle's [AGL19], where the latter was itself inspired by the former. None of them worked, and in the end we had to devise a new counting technique altogether, the one that made its way into this thesis. Nevertheless, both publications had to be analyzed in order to understand the situation in the Open CbNeed setting.

On the one hand, [AGL19] presents the CbNeed type system we thoroughly discussed in Chapter 5 (Multi types for CbN, CbV and CbNeed), one of the building blocks for the type system for Open CbNeed given in this Section.

On the other hand, [AGK20] presents an evaluation strategy dubbed "Linear head evaluation", formalized in the LSC, and a multi type system characterizing its termination and providing exact bounds is given for it. Roughly put, Linear head evaluation—see [MP94] and [DR04]—performs reduction following the well-known head reduction schema and order, while performing substitution of terms for variables in a linear way, one at a time. To the best of our knowledge, [AGK20] is the first-ever presentation of a multi type system *targeting an open LSC evaluation strategy* in which a class of derivations providing exact bounds on the normalization process is isolated.

Hence, the counting technique for exponential steps presented in [AGK20] had to be considered for the Open CbNeed system. In the end, it did not apply to our Open CbNeed multi type system, thus forcing us to resort to a completely different and novel technique.

Let us briefly explain the technique used for the Open CbNeed multi type system. The basic intuition is that

- In the weak and closed case of CbNeed, every typed variable occurrence takes part in an exponential step, which explains why indices in axioms are always $(0, 1)$.
- Conversely, not every typed variable occurrence in Open CbNeed takes part in an exponential step, and so we need to discriminate between these two categories of variable occurrences to properly assign indices in Open CbNeed axioms. This is done precisely by having two axiom rules, namely ax and ax_I.

Let us see this with an example. Consider program $p := (x\,y, [y{\leftarrow}v])$. Note that any type derivation Φ for p must be such that x belongs to the domain of the type context in the final type judgement. Moreover, if Φ is *tight*, then it is easy to see that Φ must be of the following form

$$\dfrac{\dfrac{\dfrac{\dfrac{M \in \mathsf{Inert}}{x : M \vdash^{(0,0,0)} x : M}\,\mathsf{ax}_I \qquad \dfrac{}{y : [\mathsf{abs}] \vdash^{(0,1,0)} y : [\mathsf{abs}]}\,\mathsf{ax}_I}{x : M; y : [\mathsf{abs}] \vdash^{(0,1,1)} x\,y : [\mathsf{inert}]}\,\mathsf{app}_i}{x : M; y : [\mathsf{abs}] \vdash^{(0,1,1)} (x\,y, \epsilon) : [\mathsf{inert}]}\,\mathsf{Lift} \qquad \Psi \rhd_O \emptyset \vdash^{(0,0,0)} v : [\mathsf{abs}]}{x : M \vdash^{(0,1,1)} (x\,y, [y{\leftarrow}v]) : M}\,\mathsf{ES}$$

where Lemma 7.1.2.1 (Typing properties of normal terms - Values) justifies the facts that y is typed with [abs]—instead of the alternative [inert]—and that Ψ has indices $(0, 0, 0)$.

Therefore, we notice that

- Since x appears in the Open CbNeed-normal form $(x\, v^{\alpha}, [y \leftarrow v])$ of p, which means that it does not take part in any exponential step and so it is correct to type it with indices $(0, 0, 0)$.
- Since y takes part in an exponential step in the reduction sequence, then it is correct to type it with indices $(0, 1, 0)$.

Chapter 8

Useful Open CbNeed

Introduction

In Chapter 6 we presented the Open CbNeed evaluation strategy, formalized in a setting derived from the LSC, that we called the *split* LSC. As we explained at the end of the Introduction to Chapter 6, this non-standard framework serves the purpose of paving the way for designing a derivative of the Open CbNeed evaluation strategy, one that only performs *useful* substitutions. That is, only substitutions that lead to the creation of multiplicative redexes are performed, while the ones that do not are left undone.

We present here such a strategy, which we call the *Useful Open CbNeed* evaluation strategy. Just like Open CbNeed, Useful Open CbNeed is formalized in the split LSC; as a matter of fact, both these strategies were designed as being the useful and non-useful counterpart of each other. Useful Open CbNeed refines the key concept of needed variables in Open CbNeed in such a way that each program has both *applied* and *unapplied* sets of variables, both of them covering exactly the set of its needed variables. In other words, the sets of applied and unapplied variables of a program are not a partition of its set of needed variables, but rather separate refinements of the needed variables of the program. As what happened in the Open CbNeed case, applied and unapplied sets of variables are defined first for Λ-terms and then extended to cover programs.

Interestingly enough, and much like the fact that needed variables are crucial to the operational and type-theoretical analyses of Open CbNeed, Useful Open CbNeed relies on the sets of applied and unapplied variables of expressions both for its operational and type-theoretical analyses. It should be remarked as well that the technical complexity of this part of the work is remarkably bigger than the one of Open CbNeed; we believe this to be unavoidable, as Open CbNeed constitutes the very ground upon which Useful Open CbNeed was developed.

8.1 The Useful Open CbNeed evaluation strategy

Applied and Unapplied Variables. As we just mentioned, the definition of the system is based on two subsets of the set $\mathsf{nv}(\cdot)$ of needed variables of Λ-terms and programs, namely the sets of applied and unapplied variables $\mathsf{a}(\cdot)$ and $\mathsf{u}(\cdot)$.

We prove that the two sets cover $\mathsf{nv}(t)$ exactly—that is, $\mathsf{nv}(t) = \mathsf{a}(t) \cup \mathsf{u}(t)$—and similarly for programs. Applied and unapplied variables, however, are *not* a partition of needed variables; that is, in general $\mathsf{a}(t) \cap \mathsf{u}(t) \neq \emptyset$ because a variable can have both applied and unapplied needed *occurrences*, as does x in xx—the same holds for programs.

The notion of applied variables for programs is subtle, the key, potentially confusing point being the *non-local applicative constraint*: the occurrence of a variable x may not be itself applied but, if it is meant to replace an applied variable y, then x has to be considered itself applied—this is the case, for instance, of x in $(yz, [y{\leftarrow}x])$. The non-local applicative constraint shall raise a number of technicalities in the technical development of the operational study of Useful Open CbNeed, but is transparent when dealing with its type-theoretical interpretation. Similarly, the fact that a variable occurrence is unapplied also has a non-local quality to it, and we take care of it in the technical development.

We believe that this issue is an unavoidable ingredient of usefulness in a call-by-need scenario, and not an ad-hoc point of our study. Indeed, if instead we considered x to be unapplied in $(yz, [y{\leftarrow}x])$, we would be giving rise to a notion of evaluation that does not seem to be measurable via the multi type system, and would even (incorrectly) lead us to consider programs like $(yz, [y{\leftarrow}x][x{\leftarrow}\lambda\tilde{x}.u])$ to be in Useful Open CbNeed-normal form.

Let us start with applied variables:

Definition 36 (Applied variables)**.**
The set of applied variables for Λ-terms is given by the following equations:

$$\begin{array}{rcl} \mathsf{a}(\lambda x.t) & := & \emptyset \\ \mathsf{a}(x) & := & \emptyset \\ \mathsf{a}(tu) & := & \begin{cases} \{x\} \cup \mathsf{a}(u), & \text{if } t = x \in \mathsf{Var} \\ \mathsf{a}(t) \cup \mathsf{a}(u), & \text{if } t \notin \mathsf{Var} \end{cases} \end{array}$$

The set of applied variables for $\mathcal{PR}$ is based on the one for Λ-terms as follows:

$$\begin{array}{rcl} \mathsf{a}(t, \epsilon) & := & \mathsf{a}(t) \\ \mathsf{a}(t, E[x{\leftarrow}u]) & := & \begin{cases} \mathsf{a}(t, E), & \text{if } x \notin \mathsf{nv}(t, E) \\ (\mathsf{a}(t, E) \setminus \{x\}) \cup \mathsf{a}(u), & \text{if } x \in \mathsf{nv}(t, E) \wedge (u \notin \mathsf{Var} \vee x \notin \mathsf{a}(t, E)) \\ (\mathsf{a}(t, E) \setminus \{x\}) \cup \{y\}, & \text{if } x \in \mathsf{nv}(t, E) \wedge u = y \in \mathsf{Var} \wedge x \in \mathsf{a}(t, E) \end{cases} \end{array}$$

At first sight, the definition of applied variables of programs may seem a bit cryptic to the unacquainted reader. For instance, and having in mind that $\mathsf{a}(p) \subseteq \mathsf{nv}(p)$, condition $x \in \mathsf{nv}(t, E) \wedge u = y \in \mathsf{Var} \wedge x \in \mathsf{a}(t, E)$ in the definition of $\mathsf{a}(t, E[x{\leftarrow}u])$ would simply be $x \in \mathsf{a}(t, E) \wedge u = y \in \mathsf{Var}$. However, we have not proved yet that $\mathsf{a}(p) \subseteq \mathsf{nv}(p)$, which is why the definition is given in this more general form.

Let us give a few examples explaining these three possibilities in the definition of applied variables for programs:

- *Being applied is hereditary*: Let $p = (t, E)$. Given that the applied variables of p should be contained in its needed variables, we follow the approach used for Open CbNeed and discard applied variables in E that are not (hereditarily) linked to a needed variable in t.
  For example, z is not an applied variable of $(x, [y{\leftarrow}z])$. Thus, the apparent exponential step in $q := (x, [y{\leftarrow}z][z{\leftarrow}v])$ should not be triggered, as q is in Useful Open CbNeed-normal form.
- *Adding the applied variables linked to a needed variable*: Contrary to the previous point, given a program $p := (t, E[x{\leftarrow}u])$ such that $x \in \mathsf{nv}(t)$, then the applied variables of u are included in those of p *even if* x *is not applied in* (t, E). Morally, this feature corresponds to the fact

that the *unfolding*[1] of p would only unfold ESs (hereditarily) linked to the needed variables of t, and the result of the unfolding should not contain any β-redexes.
For example, y is an applied variable of $p \coloneqq (x_1\, x_2, [x_2 \leftarrow y\, z])$, even if x_2 is not an applied variable of $(x_1\, x_2, \epsilon)$. Thus, the Useful Open CbNeed evaluation strategy shall be defined to include exponential steps like $(x_1\, x_2, [x_2 \leftarrow y\, z][z \leftarrow v]) \to_{\mathsf{und}} (x_1\, x_2, [x_2 \leftarrow v^\alpha\, z][y \leftarrow v])$.
- *Adding the variables that are non-locally applied*: Let $p \coloneqq (t, E[x \leftarrow y])$, noting that y is not applied in itself. Nonetheless, if $y \in \mathsf{a}(t, E)$, then $z \in \mathsf{a}(p)$. This corresponds to the non-local applicative constraint explained above.
For instance, if $p \coloneqq (xt, [x \leftarrow y])$, then $y \in \mathsf{a}(p)$. With $\to_{\mathsf{und}}$ being the symbol representing the Useful Open CbNeed evaluation strategy, note that the fact that $y \in \mathsf{a}(p)$ is correct, in particular considering that the following two exponential steps should apply:

$$(xt, [x \leftarrow y][y \leftarrow v]) \to_{\mathsf{und}} (xt, [x \leftarrow v^\alpha][y \leftarrow v]) \to_{\mathsf{und}} (v^\alpha t, [x \leftarrow v^\alpha][y \leftarrow v])$$

Note that these two exponential steps should be followed by a multiplicative step, morally contracting the $v^\alpha t$ β-redex.

Definition 37 (Unapplied variables)**.**
The set of unapplied variables for Λ-terms is given by the following equations:

$$\begin{array}{rcl} \mathsf{u}(\lambda x.t) & \coloneqq & \emptyset \\ \mathsf{u}(x) & \coloneqq & \{x\} \\ \mathsf{u}(tu) & \coloneqq & \begin{cases} \mathsf{u}(u) & t \in \mathsf{Var} \\ \mathsf{u}(t) \cup \mathsf{u}(u) & t \notin \mathsf{Var} \end{cases} \end{array}$$

The set of unapplied variables for programs is based on the one for terms as follows:

$$\begin{array}{rcl} \mathsf{u}(t, \epsilon) & \coloneqq & \mathsf{u}(t) \\ \mathsf{u}(t, E[x \leftarrow u]) & \coloneqq & \begin{cases} \mathsf{u}(t, E) & x \notin \mathsf{u}(t, E) \wedge (x \notin \mathsf{nv}(t, E) \vee u = y \in \mathsf{Var}) \\ (\mathsf{u}(t, E) \setminus \{x\}) \cup \mathsf{u}(u) & x \in \mathsf{u}(t, E) \vee (x \in \mathsf{nv}(t, E) \wedge u \notin \mathsf{Var}) \end{cases} \end{array}$$

Two remarks. First, if we assume the property given in Lemma 8.1.1.2 below—namely that $\mathsf{nv}(p) = \mathsf{a}(p) \cup \mathsf{u}(p)$—then note that in the definition of $\mathsf{u}(t, E[x \leftarrow u])$ the side condition $x \in \mathsf{u}(t, E) \vee (x \in \mathsf{nv}(t, E) \wedge u \notin \mathsf{Var})$ may be rewritten as $x \in \mathsf{u}(t, E) \vee (x \in \mathsf{a}(t, E) \wedge u \notin \mathsf{Var})$, which is arguably a more intuitive formulation. The same may be done for the alternative condition.

Second, and perhaps counter-intuitively, note that $y \in \mathsf{a}(p)$ and $y \in \mathsf{u}(p)$ with respect to $p \coloneqq (xx, [x \leftarrow y])$. That is, y is both applied and unapplied in p. This proves to be in accordance with the multi types view: y has only one syntactic occurrence in p but it is needed twice, and any tight type derivation of p types y twice, as we shall see below.

Let us explain the intuitions guiding the recursive definition of unapplied variables for programs by showing some examples:
- *Being unapplied is hereditary*: Similar to applied ones, unapplied variables are needed variables. Hence, if $x \notin \mathsf{nv}(t, E)$, then the unapplied variables of u are not added to those of

[1] Unfoldings are not studied in this thesis because they do not play any role in our theory of Useful Open CbNeed. However, they may be simply defined as a mapping from Λ_{L}-terms to Λ-terms which removes ESs by applying meta-level substitutions.

(t, E) in $(t, E[x{\leftarrow}u])$. This case is contemplated in the first line in the definition of unapplied variables for programs given above. For example, z is not an unapplied variable of $(x, [y{\leftarrow}z])$.
- *Adding unapplied variables linked to* any *needed variable*: We shall prove that $\mathsf{nv}(p) = \mathsf{u}(p) \cup \mathsf{a}(p)$. Therefore, z is an unapplied variable both in $(x_1\, x_2, [x_2{\leftarrow}y\, z])$—since it is linked to an unapplied variable—as well as in $(x_1\, x_2, [x_1{\leftarrow}y\, z])$—since it is linked to an *applied* variable.

As mentioned above, the following holds:

Lemma 8.1.1 (Unapplied, applied, and needed variables)**.**
1. Terms*:* $\mathsf{nv}(t) = \mathsf{u}(t) \cup \mathsf{a}(t)$ *for every* $t \in \Lambda$.
2. Programs*:* $\mathsf{nv}(p) = \mathsf{u}(p) \cup \mathsf{a}(p)$ *for every* $p \in \mathcal{PR}$.

Proof. (Click here to see the complete proof in the Technical Appendix)
The statement for terms is proven by structural induction on t, while in the one for programs we proceed by induction on the length of the environment of p.

□

In an attempt to simplify the technical development while providing a separate definition for one of the key concepts in Useful Open CbNeed, let us give the following

Definition 38 (Useless variables)**.** Given $t \in \Lambda$, we define the set of its *useless* variables as $\mathsf{ul}\,(t) \coloneqq \mathsf{u}(t) \setminus \mathsf{a}(t)$.
Similarly, the set of *useless* variables of $p \in \mathcal{PR}$ is given by $\mathsf{ul}\,(p) \coloneqq \mathsf{u}(p) \setminus \mathsf{a}(p)$.

The notion of useless variables is intuitively very simple but technically complex. Let us give some examples:
- Note that $\mathsf{ul}\,(x\, x, \epsilon) = \emptyset$.
- The previous example may be extended to a hereditary setting, noting that $\mathsf{ul}\,(y, [y{\leftarrow}x\, x]) = \emptyset$.
- However, all these reasonings only apply under the assumption of a hereditary link to the first coordinate of the program. For example, note that $x \in \mathsf{ul}\,(z\, x, [y{\leftarrow}x\, x])$, as ES $[y{\leftarrow}x\, x]$ is not linked to $z\, x$.

The notion of *useless* variables plays a decisive role in differentiating Open CbNeed from Useful Open CbNeed, considering that the set of *useless* variables of a program is comprised exactly by the needed variables of the program that *may* be bound by explicit substitutions containing values—that is, λ-abstractions—*without* creating an exponential redex in the process. In other words, we should be able to verify that if p is in Useful Open CbNeed-normal form and $x \in \mathsf{ul}\,(p)$, then $p@[x{\leftarrow}v]$ is also in Useful Open CbNeed-normal form.

Applied and unapplied variables of term contexts. Much like what we did for Open CbNeed when defining needed variables, we can extend the definition of applied and unapplied sets of variables for Λ-terms to term contexts. To avoid complicating things more than necessary, we define them directly for term contexts—instead of defining them for Λ-contexts first, like we did in the Open CbNeed case.

Definition 39 (Applied and unapplied variables of term contexts)**.**

The set of applied variables for term contexts is given by the following equations:

$$\begin{array}{rcl} \mathsf{a}(\langle\cdot\rangle) & \coloneqq & \emptyset \\ \mathsf{a}(\mathcal{H}t) & \coloneqq & \mathsf{a}(\mathcal{H}) \\ \mathsf{a}(i\mathcal{H}) & \coloneqq & \begin{cases} \{x\} \cup \mathsf{a}(\mathcal{H}) & i = x \in \mathsf{Var} \\ \mathsf{a}(i) \cup \mathsf{a}(\mathcal{H}) & i \notin \mathsf{Var} \end{cases} \end{array}$$

The set of unapplied variables for term contexts is given by the following equations:

$$\begin{array}{rcl} \mathsf{u}(\langle\cdot\rangle) & \coloneqq & \emptyset \\ \mathsf{u}(\mathcal{H}t) & \coloneqq & \mathsf{u}(\mathcal{H}) \\ \mathsf{u}(i\mathcal{H}) & \coloneqq & \begin{cases} \mathsf{u}(\mathcal{H}) & t \in \mathsf{Var} \\ \mathsf{u}(i) \cup \mathsf{u}(\mathcal{H}) & t \notin \mathsf{Var} \end{cases} \end{array}$$

Note that these definitions are a relatively simple extension of the ones given above for Λ-terms. The difference lies in that the sets are computed in a syntactic left-to-right fashion *and up to the context hole*; that is, note that $\mathsf{a}(\mathcal{H}t) = \mathsf{a}(\mathcal{H})$ and similarly for $\mathsf{u}(\mathcal{H}t)$.

The properties that hold for Λ-terms and programs also hold for term contexts, as expected:

Lemma 8.1.2 (Term contexts: Unapplied, applied and needed variables)**.**
Let $\mathcal{H}$ be a term context. Then $\mathsf{nv}(\mathcal{H}) = \mathsf{u}(\mathcal{H}) \cup \mathsf{a}(\mathcal{H})$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on $\mathcal{H}$.

□

Useful Open CbNeed evaluation contexts. The definition of evaluation contexts is particularly subtle in the useful case. We actually need to define two different notions of evaluation contexts, a more permissive one called *multiplicative evaluation contexts*, and a more restrictive one called *exponential evaluation contexts*. Multiplicative evaluation contexts are used to express the multiplicative redexes of the strategy, while the exponential evaluation contexts are used to express the exponential redexes, where the variable occurrence to be replaced has to be in an applicative context—that is, it has to be in an applied position, in accordance with the non-local applicative constraint explained above.

Both multiplicative and exponential evaluation contexts are each parameterized by a pair of sets of variables: the first one, noted $\mathcal{A}$, $\mathcal{B}$ or $\mathcal{C}$, is meant to extend the notion of applied variables of the underlying components of the evaluation context; the second one, noted $\mathcal{U}$, $\mathcal{V}$ or $\mathcal{W}$, is meant to extend the notion of unapplied variables. These "underlying components" we talk about are of varied nature, including Λ-terms or term contexts in explicit substitutions and the like.

Thus, given sets of variables $\mathcal{U}$ and $\mathcal{A}$, we write $\mathcal{E}_{\mathcal{U},\mathcal{A}}$ to represent the class of multiplicative evaluation contexts parameterized by $\mathcal{U}$ and $\mathcal{A}$, and we write $\mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ to represent the class of exponential evaluation contexts parameterized by $\mathcal{U}$ and $\mathcal{A}$, where the @ symbol is a mnemonic for "*applied*".

Like for Open CbNeed evaluation contexts, we do not provide an explicit definition of applied and unapplied variables of multiplicative and exponential evaluation contexts. In fact, $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ and $\mathcal{A}$ and $\mathcal{U}$ are defined mutually recursively, and it is only in that sense that we may call $\mathcal{A}$ and $\mathcal{U}$ the sets of *applied* and *unapplied* variables of P, respectively.

Definition 40 (Multiplicative evaluation contexts)**.**

We say that a program context P is a *multiplicative evaluation context* if it is derived using the following rules:

$$\frac{}{(\mathcal{H},\epsilon)\in\mathcal{E}_{\mathsf{u}(\mathcal{H}),\mathsf{a}(\mathcal{H})}}\ \mathsf{M_{AX}} \qquad \frac{P\in\mathcal{E}_{\mathcal{U},\mathcal{A}}\quad x\in(\mathcal{U}\cup\mathcal{A})}{P@[x{\leftarrow}y]\in\mathcal{E}_{\mathsf{upd}(\mathcal{U},x,y),\mathsf{upd}(\mathcal{A},x,y)}}\ \mathsf{M_{VAR}}$$

$$\frac{P\in\mathcal{E}_{\mathcal{U},\mathcal{A}}\quad x\in(\mathcal{U}\cup\mathcal{A})}{P@[x{\leftarrow}i^{+}]\in\mathcal{E}_{(\mathcal{U}\setminus\{x\})\cup\mathsf{u}(i^{+}),(\mathcal{A}\setminus\{x\})\cup\mathsf{a}(i^{+})}}\ \mathsf{M_{I}} \qquad \frac{P\in\mathcal{E}_{\mathcal{U},\mathcal{A}}\quad x\notin(\mathcal{U}\cup\mathcal{A})}{P@[x{\leftarrow}t]\in\mathcal{E}_{\mathcal{U},\mathcal{A}}}\ \mathsf{M_{GC}}$$

$$\frac{P\in\mathcal{E}_{\mathcal{U},\mathcal{A}}\quad x\in(\mathcal{U}\setminus\mathcal{A})}{P@[x{\leftarrow}v]\in\mathcal{E}_{\mathcal{U}\setminus\{x\},\mathcal{A}}}\ \mathsf{M_{U}} \qquad \frac{P\in\mathcal{E}_{\mathcal{U},\mathcal{A}}\quad x\notin(\mathcal{U}\cup\mathcal{A})}{P\langle x\rangle@[x{\leftarrow}\mathcal{H}]\in\mathcal{E}_{\mathcal{U}\cup\mathsf{u}(\mathcal{H}),\mathcal{A}\cup\mathsf{a}(\mathcal{H})}}\ \mathsf{M_{HER}}$$

Figure 8.1: Derivation rules for multiplicative evaluation contexts

Note that multiplicative evaluation contexts are, roughly speaking, defined similarly to Open CbNeed evaluation contexts *except for* rule

$$\frac{P\in\mathcal{E}_{\mathcal{V}}\quad x\in\mathcal{V}}{P@[x{\leftarrow}i]\in\mathcal{E}_{(\mathcal{V}\setminus\{x\})\cup\mathsf{nv}(i)}}\ \mathsf{O_{I}}$$

which is refined into 3 different rules for multiplicative evaluation contexts, depending on the kind of Λ-term contained in the explicit substitution. That is, given $P\in\mathcal{E}_{\mathcal{U},\mathcal{A}}$ and $x\in(\mathcal{U}\cup\mathcal{A})$, constraints to extend P with explicit substitution $[x{\leftarrow}t]$ are as follows:

- *Rule* $\mathsf{M_{I}}$: there are no constraints if t is an inert Λ-term. Note that $\mathsf{M_{I}}$ and $\mathsf{M_{GC}}$ together imply that we can *always* append explicit substitutions containing inert Λ-terms to multiplicative evaluation contexts, without altering the Useful Open CbNeed order of reduction.
- *Rule* $\mathsf{M_{VAR}}$: This rule covers the case where $t\in\mathsf{Var}$. It is used to handle the non-local applicative constraint for variables. The sets of variables in the conclusion of this rule are updated similarly to rule $\mathsf{M_{I}}$.
- *Rule* $\mathsf{M_{U}}$: It covers the case where $t\in\mathsf{Val}$, requiring that $x\notin\mathcal{A}$. In other words, we can derive that $P@[x{\leftarrow}t]$ is a multiplicative evaluation context *only* if $x\in(\mathcal{U}\setminus\mathcal{A})$, meaning that x is a "useless" variable of P, because otherwise extending P with $[x{\leftarrow}t]$ introduces an exponential redex.

Regarding the non-local applicative constraint, note that it impacts on multiplicative evaluation contexts only for the updating of sets $\mathcal{U}$ and $\mathcal{A}$. This is precisely what happens in rule $\mathsf{M_{VAR}}$, where the unapplied and applied set of variables are updated as $\mathsf{upd}(\mathcal{U},x,y)$ and as $\mathsf{upd}(\mathcal{A},x,y)$, respectively.

For exponential evaluation contexts, instead, it also impacts on the position of the hole, which must be located in such a way that it isolates an applied occurrence. This is implemented by introducing a notion of *applicative term context* and using it to further refine the notion of multiplicative evaluation context to take the applicativity of the context hole into consideration:

Definition 41 (Applicative term contexts)**.**

A term context $\mathcal{H}$ is called an *applicative term context* if it is derived using the following rules:

$$\mathcal{H}^{@}, \mathcal{J}^{@}, \mathcal{I}^{@} ::= \langle\cdot\rangle t \mid \mathcal{H}^{@} t \mid i\mathcal{H}^{@}$$

Note that every applicative term context is a (plain) term context, but the converse does not hold.

Let us first give the definition of exponential evaluation contexts, and discuss the role played by applicative term contexts right below:

Definition 42 (Exponential evaluation contexts)**.**
We say that a program context P is an *exponential evaluation context* if it is derived with the rules in Fig. 8.2, where upd is defined as follows

$$\mathsf{upd}(S, x, y) \coloneqq \begin{cases} S & x \notin S \\ (S \setminus \{x\}) \cup \{y\} & x \in S \end{cases}$$

$$\frac{}{(\mathcal{H}^{@}, \epsilon) \in \mathcal{E}^{@}_{\mathsf{u}(\mathcal{H}^{@}),\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_1} \qquad \frac{P \in \mathcal{E}_{\mathcal{U},\mathcal{A}} \quad x \notin (\mathcal{U} \cup \mathcal{A})}{P\langle x\rangle@[x{\leftarrow}\mathcal{H}^{@}] \in \mathcal{E}^{@}_{(\mathcal{U}\setminus\{x\})\cup\mathsf{u}(\mathcal{H}^{@}),\mathcal{A}\cup\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_2}$$

$$\frac{P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad x \in (\mathcal{U} \cup \mathcal{A})}{P@[x{\leftarrow}y] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{U},x,y),\mathsf{upd}(\mathcal{A},x,y)}}\ \mathsf{E}_{\mathsf{VAR}} \qquad \frac{P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad x \in (\mathcal{U} \cup \mathcal{A})}{P@[x{\leftarrow}i^{+}] \in \mathcal{E}^{@}_{(\mathcal{U}\setminus\{x\})\cup\mathsf{u}(i^{+}),(\mathcal{A}\setminus\{x\})\cup\mathsf{a}(i^{+})}}\ \mathsf{E}_{\mathsf{I}}$$

$$\frac{P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad x \notin (\mathcal{U} \cup \mathcal{A})}{P@[x{\leftarrow}t] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E}_{\mathsf{GC}} \qquad \frac{P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad x \in (\mathcal{U} \setminus \mathcal{A})}{P@[x{\leftarrow}v] \in \mathcal{E}^{@}_{\mathcal{U}\setminus\{x\},\mathcal{A}}}\ \mathsf{E}_{\mathsf{U}}$$

$$\frac{P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad x \notin \mathcal{A}}{P\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{U}\setminus\{x\},\mathcal{A}}}\ \mathsf{E}_{\mathsf{NL}}$$

Figure 8.2: Derivation rules for exponential evaluation contexts

Coming back to applicative term contexts, note that they play a role both in the base and hereditary cases of exponential evaluation contexts:

1. Rule $\mathsf{E}_{\mathsf{AX}_1}$ is akin to rule $\mathsf{M}_{\mathsf{HER}}$ except that it requires the term context to be *applicative*.
2. The plugging-based rule $\mathsf{M}_{\mathsf{HER}}$ for multiplicative evaluation contexts gets split in two: rule $\mathsf{E}_{\mathsf{AX}_2}$, which also serves as base case for exponential evaluation contexts and which simply plugs an applicative term context into a *multiplicative* evaluation context, and E_{NL}, which handles the special case of non-local applicative constraint.

Let us see the differences between rules $\mathsf{E}_{\mathsf{AX}_2}$ and E_{NL} with some examples:

- Consider program $p \coloneqq (x\, t, [x{\leftarrow}z])$, where z is in applied position due to the non-local applicative constraint, as it substitutes x which is applied to t. Hence, we may derive an exponential

evaluation context P that focuses on z in such a way that $P\langle z\rangle = p$ as follows:

$$\dfrac{\dfrac{}{(\langle\cdot\rangle\, t, \epsilon) \in \mathcal{E}^{@}_{\emptyset,\emptyset}}\ \mathsf{E}_{\mathsf{AX}_1} \quad x \notin \emptyset}{P \coloneqq ((\langle\cdot\rangle\, t, \epsilon)\langle x\rangle)\, @[x{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\emptyset,\emptyset}}\ \mathsf{E}_{\mathsf{NL}}$$

noting that $P = ((\langle\cdot\rangle\, t, \epsilon)\langle x\rangle)\, @[x{\leftarrow}\langle\cdot\rangle] = (x\, t, [x{\leftarrow}\langle\cdot\rangle])$, and so $p = P\langle z\rangle$ as expected. Hence, the fact that variable z is applied in p only because it is linked to x (which is itself applied) translates to exponential evaluation context P in that the premise of derivation rule E_{NL} requires the *sub-program context* to be *applied*.

- Conversely, the $\mathsf{E}_{\mathsf{AX}_2}$ derivation rule requires the *term context* to be *applicative*, while the sub-program context is in fact a multiplicative evaluation context. Since we shall prove that exponential evaluation contexts are also multiplicative evaluation contexts, this means that rule $\mathsf{E}_{\mathsf{AX}_2}$ imposes a relaxed condition on the sub-program context in the premise when compared to rule E_{NL}.
  For example, consider $p \coloneqq (x, [x{\leftarrow}z\, t])$, for which z is an applied variable because it is itself applied and is linked to needed variable x in the first coordinate of the program. Let us derive an exponential evaluation context P focusing on z in such a way that $P\langle z\rangle = p$ as follows:

$$\dfrac{\dfrac{}{(\langle\cdot\rangle, \epsilon) \in \mathcal{E}_{\emptyset,\emptyset}}\ \mathsf{O}_{\mathsf{AX}} \quad x \notin (\emptyset \cup \emptyset)}{P \coloneqq ((\langle\cdot\rangle, \epsilon))\, \langle x\rangle @[x{\leftarrow}\langle\cdot\rangle\, t] \in \mathcal{E}^{@}_{\emptyset,\emptyset}}\ \mathsf{E}_{\mathsf{AX}_2}$$

noting that $P = ((\langle\cdot\rangle, \epsilon))\, \langle x\rangle @[x{\leftarrow}\langle\cdot\rangle\, t] = (x, [x{\leftarrow}\langle\cdot\rangle\, t])$, and so $p = P\langle z\rangle$ as expected. Hence, the fact that z is applied in p translates to exponential evaluation context P in that term context $\langle\cdot\rangle\, t$ is *applicative*, while the *sub-program context* is only used to *focus* on variable x.

Let us finally define the evaluation strategy:

Definition 43 (Useful Open CbNeed evaluation strategy)**.**
The reduction rules for the Useful Open CbNeed evaluation strategy are defined as follows:

$$\begin{array}{rcll} P\langle(\lambda x.t)u\rangle & \to_{\mathsf{um}} & P\langle t, [x{\leftarrow}u]\rangle & \text{if } P \in \mathcal{E}_{\mathcal{U},\mathcal{A}} \\ P\langle x\rangle & \to_{\mathsf{ue}} & P\langle v^{\alpha}\rangle & \text{if } P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \text{ and } P(x) = v \end{array}$$

Note that $\to_{\mathsf{um}}$ is defined with respect to *multiplicative* evaluation contexts, while $\to_{\mathsf{ue}}$ is defined with respect to *exponential* evaluation contexts.

Moreover, we say that p reduces in the Useful Open CbNeed strategy to q, and write $p \to_{\mathsf{und}} q$, if $p \to_{\mathsf{um}} q$ or $p \to_{\mathsf{ue}} q$.

8.1.1 The usefulness criterion

The study of Useful Open CbNeed would be baseless if we did not prove that each exponential step invariably leads to a multiplicative step down the reduction sequence. That is, the following is of absolute importance:

Proposition 8.1.3 (Usefulness of exponential steps)**.**

Let $p = P\langle x\rangle \to_{\mathsf{ue}} P\langle v^{\alpha}\rangle = q$ with $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ and $P(x) = v$. Then there exists $r \in \mathcal{PR}$ and reduction sequence $d : q \to^{k}_{\mathsf{ue}} \to_{\mathsf{um}} r$ such that

1. *the evaluation context of each $\to_{\mathsf{ue}}$ steps in d is in $\mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$, and the one of $\to_{\mathsf{um}}$ is in $\mathcal{E}_{\mathcal{U},\mathcal{A}}$.*
2. *$k \geq 0$ is the number of $\mathsf{E_{NL}}$ rules in the derivation of $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$.*

Proof. (Click here to see the complete proof in the Technical Appendix)

By induction on the derivation of $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. The proof relies significantly on a characterization of exponential evaluation contexts that we explore in Chapter 13 (Technical appendix)—see Lemma 13.5.1 (Characterization of exponential evaluation contexts) on page 262.

□

8.2 Characterizing Useful Open CbNeed-normal forms

As in any of the other cases, the Useful Open CbNeed one also requires concise, syntax-directed gadgets to reason about normal forms. Unfortunately, restricting Open CbNeed to perform useful substitutions implies a much more complex shape of the normal forms at play. This can be appreciated in Fig. 8.3 below, where we define the unorm predicate which is meant to characterize Useful Open CbNeed-normal forms, as well as other three predicates that unorm is defined in terms of: genVar, uabs and uinert.

$$\frac{}{\mathsf{genVar}_x(x,\epsilon)}\ \mathsf{GV_{AX}} \qquad \frac{\mathsf{genVar}_x(p)}{\mathsf{genVar}_y(p@[x{\leftarrow}y])}\ \mathsf{GV_{HER}} \qquad \frac{\mathsf{genVar}_x(p) \quad z \neq x}{\mathsf{genVar}_x(p@[z{\leftarrow}t])}\ \mathsf{GV_{GC}}$$

$$\frac{}{\mathsf{uabs}(v,\epsilon)}\ \mathsf{A_{Lift}} \qquad \frac{\mathsf{genVar}_x(p)}{\mathsf{uabs}(p@[x{\leftarrow}v])}\ \mathsf{A_{GV}} \qquad \frac{\mathsf{uabs}(p)}{\mathsf{uabs}(p@[x{\leftarrow}t])}\ \mathsf{A_{GC}}$$

$$\frac{}{\mathsf{uinert}(i^{+},\epsilon)}\ \mathsf{I_{Lift}} \qquad \frac{\mathsf{genVar}_x(p)}{\mathsf{uinert}(p@[x{\leftarrow}i^{+}])}\ \mathsf{I_{GV}} \qquad \frac{\mathsf{uinert}(p) \quad x \in \mathsf{nv}(p)}{\mathsf{uinert}(p@[x{\leftarrow}i])}\ \mathsf{I_{I}}$$

$$\frac{\mathsf{uinert}(p) \quad x \in \mathsf{u}(p) \quad x \notin \mathsf{a}(p)}{\mathsf{uinert}(p@[x{\leftarrow}v])}\ \mathsf{I_{U}} \qquad \frac{\mathsf{uinert}(p) \quad x \notin \mathsf{nv}(p)}{\mathsf{uinert}(p@[x{\leftarrow}t])}\ \mathsf{I_{GC}}$$

$$\frac{\mathsf{uinert}(p) \ \vee\ \mathsf{uabs}(p) \ \vee\ \mathsf{genVar}_{\#}(p)}{\mathsf{unorm}(p)}\ \mathsf{unorm}_{P}$$

Figure 8.3: Predicates characterizing Useful Open CbNeed-normal forms

Given $\mathsf{uinert}(p)$ (resp. $\mathsf{uabs}(p)$), we say that p is a *useful inert program* (resp. a *useful abstraction program*). Additionally, we define *generalized variables* as follows

Definition 44 (Generalized variables)**.**

Given $p \in \mathcal{PR}$ such that $\mathsf{genVar}_x(p)$, we say that p is a *generalized variable* having x as its *(hereditary) head* variable. Moreover, we write $\mathsf{genVar}_\#(p)$ to state that there exists $x \in \mathsf{Var}$ such that $\mathsf{genVar}_x(p)$.

Note that useful inert programs and useful abstraction programs in Useful Open CbNeed may be seen as adaptations of inert programs and abstraction programs in Open CbNeed, respectively. But the isolation of the concept of generalized variables is a major, conceptual difference between the two evaluation strategies. Let us give some examples of generalized variables:

- The base cases for generalized variables are programs of the form (x, ϵ), for some $x \in \mathsf{Var}$.
- The fact of being a generalized variable may be extended by substituting the hereditary head variable with another variable. For example, since x is the head variable of (x, ϵ), then y is the head variable of $(x, [x{\leftarrow}y])$, and so z is the head variable of $(x, [x{\leftarrow}y][y{\leftarrow}z])$.
- We may change head variables of generalized variables, like we did in the previous item, while interspersing the program with ESs binding other variables (which should morally be garbage-collected). Thus, since x is the head variable of (x, ϵ), then y is the head variable of $(x, [\tilde{x}{\leftarrow}\mathsf{II}][x{\leftarrow}y])$, and so z is the head variable of $(x, [\tilde{x}{\leftarrow}\mathsf{II}][x{\leftarrow}y][\tilde{y}{\leftarrow}\Omega][y{\leftarrow}z])$. Note that the ESs which do not affect the head variable of the generalized variable may contain *any* term, even II and Ω, while the overall program remains a generalized variables.

Therefore, we may say that generalized variables are variables up to unfolding[2]. The interest in generalized variables is that they serve as a building block for both useful inert programs and useful abstraction programs.

To clarify this point, consider the two following derivations:

$$\frac{\mathsf{genVar}_z(z, [z{\leftarrow}x])}{\mathsf{uinert}(z, [z{\leftarrow}x][x{\leftarrow}i^+])} \qquad \frac{\mathsf{genVar}_z(z, [z{\leftarrow}x])}{\mathsf{uabs}(z, [z{\leftarrow}x][x{\leftarrow}v])}$$

each of them taking $p = (z, [z{\leftarrow}x])$ as base sub-program. Said differently, every generalized variable has the potential to become a useful inert program or a useful abstraction program, the difference lying in the kind of term we use to bound the hereditary head variable. We believe this is not an ad-hoc feature of our system, but rather something present and to be taken into consideration in the design of every presentation of usefulness.

Further below, we shall see that generalized variables have typing properties of their own, and that splitting the category of inert programs in Open CbNeed into the categories of useful inert programs and generalized variables in Useful Open CbNeed also makes sense at the level of types.

As the Lemma below shows, predicates genVar, uinert and uabs are meant to characterize disjoint categories of Useful Open CbNeed-normal forms. We shall see in the following chapter that having a disjoint partition of normal forms eases up considerably the types analysis of Useful Open CbNeed-normal forms.

Lemma 8.2.1 (Disjointness of generalized variables, useful abstraction programs and useful inert programs)**.**

For every $p \in \mathcal{PR}$, at most one *of the following holds:*

i) $\mathsf{genVar}_\#(p)$*,*

[2]Once again, we do not define an unfolding function in this thesis, so this claim should be merely taken as an intuition for the reader.

ii) $\mathsf{uabs}(p)$*, or*
iii) $\mathsf{uinert}(p)$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the length of the environment of p.

□

Let us begin proving the syntactic characterization of Useful Open CbNeed-normal forms by considering the syntactic properties of programs satisfying the genVar, uinert and uabs predicates. As expected, these syntactic properties are proven in an incremental way, extending the syntactic properties of Λ-terms into properties of programs. Before doing that, let us recall the following lemma—first presented in Chapter 6 (Open CbNeed) and used repeatedly throughout this chapter:

Lemma 8.2.2 (Redex in non-normal terms)**.**
Let $t \in \Lambda$. Then, t is not a normal term if and only if there exist term context $\mathcal{H}$, and terms $\lambda x.u$ and s such that $t = \mathcal{H}\langle(\lambda x.u)s\rangle$.

We may now proceed to present the properties satisfied by normal terms. Clearly, variables and values are too simple to state anything valuable about, so we directly move on to what we call *useful inert terms*, which are simply non-variable inert Λ-terms. That is, we prove that the size of useful inert terms is not null, and that their set of applied variables is non-empty.

Lemma 8.2.3 (Properties of useful inert terms)**.**
Let i^+ be a non-variable inert term. Then $|i^+|_{\mathsf{nd}} \geq 1$ and $\mathsf{a}(i^+) \neq \emptyset$.

(Click here to see the complete proof in the Technical Appendix)
Let us now give the properties regarding each category of programs. The properties of generalized variables form the base case:

Lemma 8.2.4 (Properties of generalized variables)**.**
Let $\mathsf{genVar}_x(p)$. Then $|p|_{\mathsf{nd}} = 0$, $\mathsf{nv}(p) = \mathsf{u}(p) = \{x\}$ and $\mathsf{a}(p) = \emptyset$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{genVar}_x(p)$.

□

The properties of useful abstraction programs rely on Lemma 8.2.4:

Lemma 8.2.5 (Properties of useful abstraction programs)**.**
Let $\mathsf{uabs}(p)$. Then $|p|_{\mathsf{nd}} = 0$ and $\mathsf{nv}(p) = \mathsf{u}(p) = \mathsf{a}(p) = \emptyset$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{uabs}(p)$.

□

The properties of useful inert programs also rely on Lemma 8.2.4:

Lemma 8.2.6 (Properties of useful inert programs)**.**
Let $\mathsf{uinert}(p)$. Then $|p|_{\mathsf{nd}} \geq 1$ and $\mathsf{a}(p) \neq \emptyset$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{uinert}(p)$.

□

This concludes the list of syntactic properties satisfied by Useful Open CbNeed-normal forms that we wanted to present in this chapter. However, Proposition 8.2.7 relies on a non-trivial series of other lemmas that we considered too technical to even be mentioned in this chapter. We refer the reader to Subsect. 13.5.2 in the Technical Appendix, starting on page 264, to see all that is required for Proposition 8.2.7.

Proposition 8.2.7 (Syntactic characterization of Useful Open CbNeed-normal forms)**.**
Let $p \in \mathcal{PR}$. Then p is in $\to_{\mathsf{und}}$-normal form if and only if $\mathsf{unorm}(p)$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the length of the environment of p.

□

8.3 Determinism

Another important property Useful Open CbNeed should satisfy is determinism. This is simply proven in a way analogous to that of Open CbNeed, following the same proof schema.

First, we define what a reduction place in the Useful Open CbNeed case is:

Definition 45 (Reduction places in Useful Open CbNeed)**.**
Let $t \in \Lambda$, $\mathcal{H}$ be a term context, and let $S \subseteq \mathsf{Var}$. We say that t is a *S-reduction place* of $\mathcal{H}\langle t\rangle$ if one of the following conditions holds:

- *Multiplicative redex*: $t = (\lambda x.u)s$;
- *New hereditary jump*: $t = x$ and $x \notin S \supseteq \mathsf{nv}(\mathcal{H})$.
- *New applied variable*: $t = x$, $\mathcal{H}$ is applicative, and $x \notin S \supseteq \mathsf{a}(\mathcal{H})$.

Let $t \in \Lambda$ and P be a program context, and let $S \subseteq \mathsf{Var}$. We say that t is a *S-reduction place* of $P\langle t\rangle$ if one of the following conditions hold:

- *Multiplicative redex*: $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ and $t = (\lambda x.u)s$;
- *Exponential redex*: $t = x$, $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$, $x \in \mathsf{dom}(P)$ and $P(x) = v$;
- *New hereditary jump*: $t = x$, $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$, $x \notin S \supseteq (\mathcal{U} \cup \mathcal{A})$, and $x \notin \mathsf{dom}(P)$.
- *New applied variable*: $t = x$, $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$, $x \notin S \supseteq \mathcal{A}$, and $x \notin \mathsf{dom}(P)$.

Note that the conditions in the *New applied variable* item for term contexts above are included in the conditions of *New hereditary jump* for term contexts. In spite of this, we state it as a separate category of reduction places, since the former serves as base case for *New applied variable*—for exponential evaluation contexts)—while the latter serves as base case for *New hereditary jump*—for multiplicative evaluation contexts.

The following is a direct consequence of Lemma 6.3.1 (Unique decomposition of Λ-terms) in the Determinism subsection of Open CbNeed—Sect. 6.3—and Lemma 8.1.2 (Term contexts: Unapplied, applied and needed variables).

Lemma 8.3.1 (Unique decomposition of Λ-terms)**.**
Let $\mathcal{H}_1\langle t_1\rangle = \mathcal{H}_2\langle t_2\rangle$*, with* $\mathcal{H}_1, \mathcal{H}_2$ *term contexts, let* $S \supseteq (\mathsf{a}(\mathcal{H}_1) \cup \mathsf{a}(\mathcal{H}_2))$*, and let* t_i *be an* S*-reduction place of* $\mathcal{H}_i\langle t_i\rangle$*, for* $i = 1, 2$*.*
Then $t_1 = t_2$ *and* $\mathcal{H}_1 = \mathcal{H}_2$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on any one of the term contexts.

□

Lemma 8.3.2 (Unique decomposition of programs)**.**
Let $P_1\langle t_1\rangle = P_2\langle t_2\rangle$*, with* $P_1 \in (\mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1} \cup \mathcal{E}^{@}_{\mathcal{U}_1,\mathcal{A}_1})$ *and* $P_2 \in (\mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2} \cup \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2})$ *such that* $S \supseteq (\mathcal{U}_1 \cup \mathcal{A}_1 \cup \mathcal{U}_2 \cup \mathcal{A}_2)$*, and* t_i *be an* S*-reduction place of* $P_i\langle t_i\rangle$ *for* $i = 1, 2$*.*
Then $t_1 = t_2$ *and* $P_1 = P_2$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on any one of the open evaluation contexts.

□

Finally,

Corollary 8.3.3 (Determinism of Useful Open CbNeed)**.**
If $p \to_{\mathsf{und}} q$ *and* $p \to_{\mathsf{und}} r$ *then* $q = r$*.*

Proof. (Click here to see the complete proof in the Technical Appendix)
Straightforward application of Lemma 8.3.2.

□

Chapter 9

Multi types for Useful Open CbNeed

Introduction

Technicalities aside, this chapter provides what other ones in this work provide for their corresponding evaluation strategies: a multi type system in which typability is equal to Useful Open CbNeed-normalization, together with a class of type derivations—in said multi type system—which contain quantitative information regarding the Useful Open CbNeed-normalization process.

That being said, it cannot be denied that the operational aspects of Useful Open CbNeed make the entire study harder and trickier to tackle, forcing the study of the multi type system to meet the new challenges arising from the operational side. Fortunately, the Open CbNeed case serves as a remarkably helpful base tool with which we can deal with Useful Open CbNeed, particularly in the type-theoretical aspects.

We shall see that a very minor change to the Open CbNeed type system is required to get a multi type system that gives exact bounds on the Useful Open CbNeed-normalization process. Although proving that this is the case shall show to require complex reasoning on the sets of applied and unapplied variables of expressions, the type system is designed in such a way that reasoning about these sets of variables comes out naturally and consistently with the other systems in this work.

Let us begin by deriving a multi type system for Useful Open CbNeed following a very simple reasoning about axioms/variable occurrences.

9.1 Multi type system for Useful Open CbNeed

9.1.1 Changing quantitative information in axioms.

Let us go back to the Open CbNeed type system for a moment. Recall that Open CbNeed is an evaluation strategy that performs *linear* substitutions, and so counting exponential steps in a Open CbNeed-normalizing sequence means counting the number of variable occurrences substituted by a value, along said normalizing sequence, until reaching the Open CbNeed-normal form. The Open CbNeed type system captures this by counting 1 for every typed variable occurrence whose type is not in Inert, and counting 0 for every variable occurrence which is not typed or which is typed with a multi type in Inert. *Typed* variable occurrences are thus split in:

$$\frac{M \notin \mathsf{Inert} \quad M \neq \mathbf{0}}{x : M \vdash^{(0,1,0)} x : M}\ \mathsf{ax} \qquad \frac{M \in \mathsf{Inert}}{x : M \vdash^{(0,0,0)} x : M}\ \mathsf{ax}_I$$

The set Inert of multi types was proven to be a correct type-theoretical discriminant between an exponential step and a variable occurrence that is not be substituted—in the Open CbNeed-normalizing sequence of a program containing said variable occurrence—in Chapter 7 (Multi types for Open CbNeed), when we show that exponential steps are correctly counted in tight type derivations. The intuition is simple: we only substitute values in Open CbNeed, and since values are not typable with the inert linear constants, then variables typed with a multi type in Inert cannot be substituted by values.

Then, a natural follow up question arises: since values are typable both with abs and types of the form $M \multimap N$, then what is the semantical difference between these linear types? To answer this, let us consider a subset of applications of the ax rule, namely those of the form $\Phi \rhd_O x : M \vdash^{(0,1,0)} x : M$ where $M \in \mathsf{Abs}$. Note that, among all three typing rules used to type application Λ-terms—namely app, app_i and $\mathsf{app}_{\mathrm{gc}}$—it is impossible to place Φ as left-hand side premise of any of these typing rules. That is, it seems that if Φ is used to type a larger Λ-term or program, then the particular occurrence of x in Φ can never end up in an applied position.

All this gives us the intuition that the variable occurrences which end up being substituted in a Useful Open CbNeed-normalizing sequence are those typed with a multi type containing at least one linear arrow type, while any other variable occurrence—be it typed with any multi type in Tight, or not typed at all—are not be substituted along the sequence.

The intuition above led us to derive a multi type system for Useful Open CbNeed simply by changing the way we count exponential steps in axioms:

Definition 46 (The Useful Open CbNeed type system)**.**

The typing rules of the Useful Open CbNeed type system appear in Fig. 9.1.

We write $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} \mathsf{e} : M$ to express that type derivation Φ in the Useful Open CbNeed multi type system ends in type judgement $\Gamma \vdash^{(m,e,r)} \mathsf{e} : M$.

Note that the axioms rules are the only difference between the Open CbNeed type system—see Fig. 7.1 in Chapter 7 (Multi types for Open CbNeed)—and the Useful Open CbNeed type system in Fig. 9.1, using Inert to discriminate between axiom cases in the former and using Tight to discriminate between axioms cases in the latter. All the other typing rules are the same in both systems.

As a first requirement of our Useful Open CbNeed type system, the following must hold

Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system)**.**

Let e *be an expression and* $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} \mathsf{e} : M$ *be a type derivation. If* $x \notin \mathsf{fv}(\mathsf{e})$ *then* $x \notin \mathsf{dom}(\Gamma)$.

Let us now present our notion of *tight* type derivations, constituting a class of type derivations in the Useful Open CbNeed system which we shall prove to provide minimal indices.

Definition 47 (Tight type derivations for Useful Open CbNeed)**.**

A derivation $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} p : M$ is said to be *tight* if $M = [\mathsf{tight}]$ and $\Gamma(x) = \mathsf{Tight}$ for every $x \in \mathsf{dom}(\Gamma)$.

Note that this definition covers the one given for Open CbNeed—namely, those whose type contexts only assign multi types in Inert. As explained right after Definition 34 (Tight derivations for Open CbNeed, page 90), the definition of tight type derivations we are presenting here is more natural than the one given for Open CbNeed. But it is only within the Useful Open CbNeed evaluation strategy that we have all the elements needed to make the notion of applied and unapplied positions explicit enough, thus allowing for this extended and more natural isolation of a class of minimal type derivations.

$$\frac{M \in \mathsf{Tight}}{x : M \vdash^{(0,0,0)} x : M}\ \mathsf{ax}_T \qquad \frac{M \notin \mathsf{Tight} \quad M \neq \mathbf{0}}{x : M \vdash^{(0,1,0)} x : M}\ \mathsf{ax}$$

$$\frac{\Gamma; x : N \vdash^{(m,e,r)} t : M}{\Gamma \vdash^{(m,e,r)} \lambda x.t : N \multimap M}\ \mathsf{fun} \qquad \frac{}{\vdash^{(0,0,0)} \lambda x.t : \mathsf{abs}}\ \mathsf{abs}$$

$$\frac{(\Gamma_i \vdash^{(m_i,e_i,r_i)} \lambda x.t : L_i)_{i \in I}}{\biguplus_{i \in I} \Gamma_i \vdash^{(\sum_{i \in I} m_i, \sum_{i \in I} e_i, \sum_{i \in I} r_i)} \lambda x.t : [L_i]_{i \in I}}\ \mathsf{many}$$

$$\frac{\Gamma \vdash^{(m,e,r)} t : [N \multimap M] \quad \Pi \vdash^{(m',e',r')} u : N \quad N \neq \mathbf{0}}{\Gamma \uplus \Pi \vdash^{(m+m'+1, e+e', r+r')} tu : M}\ \mathsf{app}$$

$$\frac{\Gamma \vdash^{(m,e,r)} t : [\mathsf{inert}]_{j \in J} \quad \Pi \vdash^{(m',e',r')} u : [\mathsf{tight}] \quad J \neq \emptyset}{\Gamma \uplus \Pi \vdash^{(m+m', e+e', r+r'+1)} tu : [\mathsf{inert}]_{j \in J}}\ \mathsf{app}_i$$

$$\frac{\Gamma \vdash^{(m,e,r)} t : [\mathbf{0} \multimap M]}{\Gamma \vdash^{(m+1,e,r)} tu : M}\ \mathsf{app}_{\mathrm{gc}}$$

$$\frac{\Gamma \vdash^{(m,e,r)} t : M}{\Gamma \vdash^{(m,e,r)} (t, \epsilon) : M}\ \mathsf{Lift}$$

$$\frac{\Gamma; x : N \vdash^{(m,e,r)} (t, E) : M \quad \Pi \vdash^{(m',e',r')} u : N \quad N \neq \mathbf{0}}{\Gamma + \Pi \vdash^{(m+m', e+e', r+r')} (t, E[x \leftarrow u]) : M}\ \mathrm{ES}$$

$$\frac{\Gamma \vdash^{(m,e,r)} (t, E) : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} (t, E[x \leftarrow u]) : M}\ \mathrm{ES}_{\mathrm{gc}}$$

Figure 9.1: Type system for Useful Open CbNeed evaluation

9.1.2 Useful Open CbNeed correctness

Refining the typing properties of inert expressions: spreading tightness within the type context. Here, we proceed to prove a Correctness theorem for Useful Open CbNeed: if $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} p : M$ is tight, then there exists a reduction sequence $d : p \to^*_{\mathsf{und}} q$ such that q is in $\to_{\mathsf{und}}$-normal form and $(m, e, r) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})$.

As a base case for this, we need to prove that if p is itself in $\to_{\mathsf{und}}$-normal form, then $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$. In order to do so, a *weaker* hypothesis on Φ is assumed, namely for the induction to go through, thus allowing us to prove the Correctness theorem in a modular—that is, inductive—way. The reason for this is subtle, and hopefully better explained with the following example:

Let $p \coloneqq (t, E'[x{\leftarrow}v])$ be a program in Useful Open CbNeed-normal form, with $v \in \mathsf{Val}$. By Proposition 8.2.7 (Syntactic characterization of Useful Open CbNeed-normal forms), we get that either $\mathsf{genVar}_{\#}(p)$ or $\mathsf{uabs}(p)$ or $\mathsf{uinert}(p)$. Say $\mathsf{uinert}(p)$, derived as follows

$$\frac{\mathsf{uinert}(t, E') \quad x \in \mathsf{ul}\,(t, E')}{\mathsf{uinert}(t, E'[x{\leftarrow}v])}\ \mathsf{I_U}$$

Note that $|(t, E'[x{\leftarrow}v])|_{\mathsf{nd}} = |(t, E')|_{\mathsf{nd}}$, since $|v|_{\mathsf{nd}} = 0$.

Moreover, let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} (t, E'[x{\leftarrow}v]) : M$ be a tight type derivation of the form

$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m',e',r')} (t, E') : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} v : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} (t, E'[x{\leftarrow}v]) : M}\ \mathrm{ES}$$

Since we want to prove that $(m' + m'', e' + e'', r' + r'') = (0, 0, |(t, E'[x{\leftarrow}v])|_{\mathsf{nd}})$, and given that $|(t, E')|_{\mathsf{nd}} = r'$, we should be able to prove in particular that $r'' = 0$. However, if $N \notin \mathsf{Tight}$ then we cannot ensure this. For example, let $v = \lambda y.yy$, $N = [[\mathsf{inert}, \mathsf{inert}] \multimap [\mathsf{inert}]]$ and $\Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} \lambda y.yy : N$ be derived as

$$\dfrac{\dfrac{\dfrac{\dfrac{}{y : [\mathsf{inert}] \vdash^{(0,0,0)} y : [\mathsf{inert}]}\ \mathsf{ax}_T \quad \dfrac{}{y : [\mathsf{inert}] \vdash^{(0,0,0)} y : [\mathsf{inert}]}\ \mathsf{ax}_T}{y : [\mathsf{inert}, \mathsf{inert}] \vdash^{(0,0,1)} yy : [\mathsf{inert}]}\ \mathsf{app}_i}{\emptyset \vdash^{(0,0,1)} \lambda y.yy : [\mathsf{inert}, \mathsf{inert}] \multimap [\mathsf{inert}]}\ \mathsf{fun}}{\emptyset \vdash^{(0,0,1)} \lambda y.yy : [[\mathsf{inert}, \mathsf{inert}] \multimap [\mathsf{inert}]]}\ \mathsf{many}$$

where $r'' = 1 \neq 0$.

Nevertheless, we shall show that this is *impossible* under the assumption that Φ is tight, which forces $N \in \mathsf{Tight}$. This is because we can infer the tightness of the useless variables in the type context from the tightness of the applied variables, while showing that no other variable can appear in the domain of the type context. In our running example, the tightness of the useless variable x (*i.e.*, that $N \in \mathsf{Tight}$) is given by the tightness of the variables appearing in the intersection of the domain of Π and the applied variables of (t, E') in $\Psi \rhd_U \Pi; x : N \vdash^{(m',e',r')} (t, E') : M$.

In a more general sense, we can say that given a tight type derivation for a Useful Open CbNeed-normal form, there exists a kind of *spreading of tightness* from the applied variables into the unapplied ones. Although this property could be proven to hold for Open CbNeed, it is only stated for Useful Open CbNeed because it requires having refined the definition of needed variables into that of applied and unapplied ones. These are not relevant concepts in Open CbNeed, and so we have skipped this analysis therein.

Moreover, the spreading of tightness studied in the open case—that is, the *inert spreading of tightness* from the type context to the right-hand type—happens to be relevant also in the useful

case, and are therefore included in the typing properties of useful inert expressions.

Like it happened in the Open CbNeed, we first need to provide typing properties of values before being able to prove the typing properties of useful inert terms:

Lemma 9.1.2 (Typing properties of values)**.**
Let $\Phi \triangleright_U \Gamma \vdash^{(m,e,r)} v : M$ *with* $M \in \mathsf{Tight}$.
Then $(m, e, r) = (0, 0, |v|_{\mathsf{nd}})$, $\mathsf{dom}(\Gamma) = \mathsf{nv}(v)$, *and* $M \in \mathsf{Abs}$.

(Click here to see the complete proof in the Technical Appendix)

Tight type derivations for useful inert terms and useful inert programs satisfy the *exact same* typing properties—namely, the one discussed above; we prove it first for useful inert terms given that useful inert programs build on them:

Lemma 9.1.3 (Typing properties of useful inert terms)**.**
Let $\Phi \triangleright_U \Gamma \vdash^{(m,e,r)} i^+ : M$ *such that* $\mathsf{tight}_\Gamma(\mathsf{a}(i^+))$.
Then $(m, e, r) = (0, 0, |i^+|_{\mathsf{nd}})$, $\mathsf{dom}(\Gamma) = \mathsf{nv}(i^+)$, $\mathsf{tight}(\Gamma)$, *and* $M \in \mathsf{Inert}$.

(Click here to see the complete proof in the Technical Appendix)

What do types say about generalized variables? In the study of the operational properties of Useful Open CbNeed, we defined generalized variables separately from useful inert expressions. Remarkably, this is coherent with our type-theoretical study of Useful Open CbNeed: we shift from having hypotheses on the type derivations defined in terms of the *needed* variables in the open case, to defining them in terms of the *applied* variables in the useful case. This shift in the assumptions of the type derivations can be successfully applied precisely because generalized variables have been separated from the characterization by the uinert predicate: on the one hand, Lemma 8.2.4 (Properties of generalized variables) shows that generalized variables have no applied variables; on the other hand, useful inert expressions always do, as their hereditary head variables are applied—by Lemma 8.2.3 (Properties of useful inert terms) and Lemma 8.2.6 (Properties of useful inert programs). Therefore, we study the typing properties of generalized variables separately:

Lemma 9.1.4 (Typing properties of generalized variables)**.**
Let $\Phi \triangleright_U \Gamma \vdash^{(m,e,r)} p : M$ *such that* $\mathsf{genVar}_x(p)$.
Then $\mathsf{dom}(\Gamma) = \mathsf{nv}(p) = \{x\}$ *and* $\Gamma(x) = M$. *Moreover, if* $M \in \mathsf{Tight}$ *then* $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{genVar}_x(p)$.

□

Typing properties of useful inert programs and of useful abstraction programs, when they are based on generalized variables. Let us explain the consequences of Lemma 9.1.4 with a few of examples:

First, let us take $p \coloneqq (x, [x{\leftarrow}y][y{\leftarrow}zz])$, which satisfies that $\mathsf{uinert}(p)$ by the following derivation:

$$\dfrac{\dfrac{}{\mathsf{genVar}_y(x, [x{\leftarrow}y])}\,\mathsf{GV_{AX}}}{\mathsf{uinert}(x, [x{\leftarrow}y][y{\leftarrow}zz])}\,\mathsf{I_{GV}}$$

Say, moreover, that $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} (x, [x{\leftarrow}y][y{\leftarrow}zz]) : M$ is a *tight* type derivation obtained as follows:

$$\dfrac{\Psi \rhd_U \Pi; y : O \vdash^{(m',e',r')} (x, [x{\leftarrow}y]) : M \quad \Theta \rhd_U z : N \vdash^{(m'',e'',r'')} zz : O}{\Pi; z : N \vdash^{(m'+m'',e'+e'',r'+r'')} (x, [x{\leftarrow}y][y{\leftarrow}zz]) : M}\,\mathrm{ES}$$

where $\Gamma = \Pi; z : N$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$.

As explained above, we are faced here with the challenge of proving that the indices in Φ are the expected ones—*i.e.*, that $(m, e, r) = (0, |(x, [x{\leftarrow}y][y{\leftarrow}zz])|_{\mathsf{nd}}, 0)$—under the assumption that Γ is tight. This requires proving that $(m', e', r') = (0, 0, |(x, [x{\leftarrow}y])|_{\mathsf{nd}})$, on the one hand, and $(m'', e'', r'') = (0, 0, |zz|_{\mathsf{nd}})$, on the other hand. How do we succeed in proving this?

First, by noting that Lemma 9.1.4 (Typing properties of generalized variables) on Ψ gives that $\Pi = \emptyset$, that $O = M \in \mathsf{Tight}$—by the assumption that Φ is tight—and that $(m', e', r') = (0, 0, |(x, [x{\leftarrow}y])|_{\mathsf{nd}})$. Thus, refining Φ with these new conclusions, we see that it is derived as follows:

$$\dfrac{\Psi \rhd_U y : M \vdash^{(0,0,|(x,[x{\leftarrow}y])|_{\mathsf{nd}})} (x, [x{\leftarrow}y]) : M \quad \Theta \rhd_U z : N \vdash^{(m'',e'',r'')} zz : M}{z : N \vdash^{(m'',e'',|(x,[x{\leftarrow}y])|_{\mathsf{nd}}+r'')} (x, [x{\leftarrow}y][y{\leftarrow}zz]) : M}\,\mathrm{ES}$$

Finally, we need to prove that Θ is minimal as well. For this, note that $z \in \mathsf{a}(x, [x{\leftarrow}y][y{\leftarrow}zz])$ and that zz is a useful inert term. Moreover, note that $\mathsf{tight}_\Gamma(\mathsf{a}(p))$—since $\mathsf{tight}(\Gamma)$ by assumption—and so $\mathsf{tight}_{z:N}(\mathsf{a}(zz))$, allowing us to apply Lemma 9.1.3 (Typing properties of useful inert terms) on Θ to conclude that $(m'', e'', r'') = (0, 0, |zz|_{\mathsf{nd}})$.

This example represents roughly how the *minimality* of a tight type derivation is proven for useful inert programs built from generalized variables. But, what happens when the program under consideration is instead a useful abstraction program, also built from a generalized variable?

Let us take now $q \coloneqq (x, [x{\leftarrow}y][y{\leftarrow}\lambda z.t])$, for which we can derive:

$$\dfrac{\dfrac{}{\mathsf{genVar}_y(x, [x{\leftarrow}y])}\,\mathsf{GV_{AX}}}{\mathsf{uinert}(x, [x{\leftarrow}y][y{\leftarrow}\lambda z.t])}\,\mathsf{I_{GV}}$$

Following a similar reason to the one used for p above—*i.e.*, applying Lemma 9.1.4 (Typing properties of generalized variables)—a type derivation for q must be of the following form:

$$\dfrac{\Psi \rhd_U y : M \vdash^{(0,0,|(x,[x{\leftarrow}y])|_{\mathsf{nd}})} (x, [x{\leftarrow}y]) : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} \lambda z.t : M}{\Delta \vdash^{(m'',e'',|(x,[x{\leftarrow}y])|_{\mathsf{nd}}+r'')} (x, [x{\leftarrow}y][y{\leftarrow}\lambda z.t]) : M}\,\mathrm{ES}$$

This time, inferring the minimality of Φ under the assumption that it is tight requires a different reasoning on sub-type derivation Θ: since we know that $M \in \mathsf{Tight}$, we can conclude that Θ is minimal by application of Lemma 9.1.2 (Typing properties of values), getting that $(m'', e'', r'') = (0, 0, |\lambda z.t|_{\mathsf{nd}})$.

Now that we have explained more concretely what the set of hypotheses in each case should be, given the technicalities at play in each of them, let us proceed to present the properties. Building on Lemma 9.1.4, we now give the typing properties of useful abstraction programs:

Lemma 9.1.5 (Typing properties of useful abstraction programs)**.**
Let $\Phi \triangleright_U \Gamma \vdash^{(m,e,r)} p : M$ *such that*
- $\mathsf{uabs}(p)$*, and*
- $M \in \mathsf{Tight}$.

Then $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$, $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$*, and* $M \in \mathsf{Abs}$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{genVar}_x(p)$.

□

Finally, the one for useful inert programs, which also builds on Lemma 9.1.4. Note that the typing properties of useful inert terms and useful inert programs are morally the same, where the one for programs is based on the one for terms:

Lemma 9.1.6 (Typing properties of useful inert programs)**.**
Let $\Phi \triangleright_U \Gamma \vdash^{(m,e,r)} p : M$ *such that*
- $\mathsf{uinert}(p)$,
- $\mathsf{tight}_\Gamma(\mathsf{a}(p))$.

Then $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$, $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$, $\mathsf{tight}(\Gamma)$*, and* $M \in \mathsf{Inert}$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{uinert}(p)$.

□

Having proven separate typing properties for each of the three predicates that characterize Useful Open CbNeed-normal forms—under the assumption that Proposition 8.2.7 (Syntactic characterization of Useful Open CbNeed-normal forms) holds—we can finally prove the following

Proposition 9.1.7 (Typing properties of Useful Open CbNeed-normal forms)**.**
Let $p \in \mathcal{PR}$ *be such that* $\mathsf{unorm}(p)$*, and let* $\Phi \triangleright_U \Gamma \vdash^{(m,e,r)} p : M$ *be a tight type derivation for it. Then* $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$ *and* $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By case analysis on whether $\mathsf{genVar}_{\#}(p)$, $\mathsf{uabs}(p)$ or $\mathsf{uinert}(p)$, according to what is given by Lemma 8.2.1 (Disjointness of generalized variables, useful abstraction programs and useful inert programs).

□

Linear Substitution. Let us consider the Substitution property in the Useful Open CbNeed. We only provide the statement here, and proceed to prove this entire part in the Appendix.

Once again, we first need to cover the term contexts cases before lifting the properties to program contexts. As expected, we only study linear substitution concerning applicative term contexts and exponential evaluation contexts, namely because it is inside those contexts that values are substituted for variables in the useful case.

The approach used here is consistent with the rest of this chapter: we assume that term contexts and exponential contexts satisfy certain conditions regarding only their sets of *applied* variables.

Lemma 9.1.8 (Linear Substitution for Useful Open CbNeed)**.**

1. *Let $\mathcal{H}^{@}$ be such that $x \notin \mathsf{a}(\mathcal{H}^{@})$, and let*

$$\Phi \rhd_U \Gamma; x : M \vdash^{(m,e,r)} \mathcal{H}^{@}\langle x \rangle : N$$

be such that $M, N \neq \mathbf{0}$, and $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}^{@}))$.
Then there exists a splitting $M = M_1 \uplus M_2$, with $M_1 \neq \mathbf{0}$, such that for every $\Psi \rhd_U \Pi \vdash^{(m',e',r')} v : M_1$ there exists

$$\Theta \rhd_U \left(\Gamma \biguplus \Pi\right); x : M_2 \vdash^{(m+m',e+e'-1,r+r')} \mathcal{H}^{@}\langle v \rangle : N$$

2. *Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ such that $x \notin \mathcal{A}$ and $x \notin \mathsf{dom}(P)$, and let*

$$\Phi \rhd_U \Gamma; x : M \vdash^{(m,e,r)} P\langle x \rangle : N$$

be such that $M, N \neq \mathbf{0}$, and $\mathsf{tight}_\Gamma(\mathcal{A})$.
Then there exists a splitting $M = M_1 \uplus M_2$, with $M_1 \neq \mathbf{0}$, such that for every type derivation $\Psi \rhd_U \Pi \vdash^{(m',e',r')} v : M_1$ there exists a type derivation

$$\Theta \rhd_U \left(\Gamma \biguplus \Pi\right); x : M_2 \vdash^{(m+m',e+e'-1,r+r')} P\langle v \rangle : N$$

Proof. (Click here to see the complete proof in the Technical Appendix)

1. By structural induction on $\mathcal{H}$.
2. By structural induction on P.

□

By Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system), it is safe to assume, in both items of Lemma 9.1.8 (Linear substitution for Useful Open CbNeed) above, that $x \notin \mathsf{dom}(\Pi)$ in $\Psi \rhd_U \Pi \vdash^{(m',e',r')} v : M_1$. The reason is that Lemma 9.1.8 is only used when proving Proposition 9.1.10.2 (Quantitative Subject Reduction for Useful Open CbNeed). That is, the context in which Lemma 9.1.8 is used is when we consider the substitution of a variable x bound by an explicit substitution of the form $[x \leftarrow v]$. Thus, α-conversion allows us to safely assume that $x \notin \mathsf{nv}(v)$, while Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system) gives us that $x \notin \mathsf{dom}(\Pi)$.

Next, we move on to prove the required Subject Reduction property. Once again, we first prove the properties for term contexts and then prove them for program contexts on top of the ones for term contexts.

Lemma 9.1.9 (Quantitative Subject Reduction for $\to_{\mathsf{um}}$ in term contexts)**.**
Let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} \mathcal{H}\langle(\lambda x.u)s\rangle : M$ such that $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}))$. Then $m \geq 1$ and there exists $\Phi' \rhd_U \Gamma \vdash^{(m-1,e,r)} (\mathcal{H}\langle u \rangle, [x \leftarrow s]) : M$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on $\mathcal{H}$.

□

Proposition 9.1.10 (Quantitative Subject Reduction for Useful Open CbNeed)**.**
Let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} p : M$ be a tight type derivation.

1. Multiplicative*: if* $p \to_{\mathsf{um}} p'$*, then* $m \geq 1$ *and there exists a type derivation* $\Phi' \rhd_U \Gamma \vdash^{(m-1,e,r)} p' : M$.
2. Exponential*: if* $p \to_{\mathsf{ue}} p'$*, then* $e \geq 1$ *and there exists a type derivation* $\Phi' \rhd_U \Gamma \vdash^{(m,e-1,r)} p' : M$.

Proof. (Click here to see the complete proof in the Technical Appendix)

The multiplicative case is proven by induction on $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ such that $p = P\langle s\rangle \to_{\mathsf{um}} P\langle s'\rangle = p'$, using Lemma 9.1.9 (Quantitative Subject reduction for $\to_{\mathsf{um}}$ in term contexts) for the base case.

The exponential case is proven by induction on $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ such that $p = P\langle x\rangle \to_{\mathsf{ue}} P\langle v^{\alpha}\rangle = p'$ and $p(x) = v$, using Lemma 9.1.8 (Linear Substitution for Useful Open CbNeed) for the base case. □

It is interesting to remark here that the proof of both items in Proposition 9.1.10 is obtained by weakening the assumptions on Φ, namely by only assuming that $\mathsf{tight}_\Gamma(\mathcal{A})$ instead of $\mathsf{tight}(\Gamma)$. This is similar to what we did in the proof of Proposition 7.1.6 (Quantitative Subject Reduction for Open CbNeed)—where we only assume that $\mathsf{inert}_\Gamma(\mathcal{V})$, with $\Gamma \in \mathcal{E}_{\mathcal{V}}$.

Finally,

Theorem 9.1.11 (Tight Correctness for Useful Open CbNeed)**.**
Given $p \in \mathcal{PR}$ *and a tight type derivation* $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} p : M$*, there exists* $q \in \mathcal{PR}$ *such that*
1. q *is in* $\to_{\mathsf{und}}$*-normal form,*
2. *there exists a reduction sequence* $d : p \to^*_{\mathsf{und}} q$*, and*
3. $(m, e, r) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})$.
4. $\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$.

Proof. (Click here to see the complete proof in the Technical Appendix)

By induction on the sum $m + e$, proceeding by case analysis on whether $p \to_{\mathsf{und}}$-reduces or not. If it is in $\to_{\mathsf{und}}$-normal form, then the statement follows by Proposition 9.1.7 (Typing properties of Useful Open CbNeed-normal forms). Instead, if $p \to_{\mathsf{und}}$-reduces, then the statement follows by Proposition 9.1.10 (Quantitative Subject Reduction for Useful Open CbNeed) and then by application of the *i.h.*.

□

9.1.3 Useful Open CbNeed completeness

Tight typability of programs in Useful Open CbNeed-normal form. As we noticed in Chapter 8 (Useful Open CbNeed), usefulness induces a very delicate and technical notion of normal forms. This gets especially reflected at the level of the type system in the completeness part, where many technicalities show up.

The goal for the completeness part is to show that, for every program which Useful Open CbNeed-normalizes there exists a *tight* type derivation. As a base case for this, we need to prove that every program in Useful Open CbNeed-normal form can be given a tight type derivation; this is proven separately for each category of Useful Open CbNeed-normal forms. But proving the tight typability of useful inert programs requires a delicate treatment on the variables in the domain of the type context.

To understand why, let us first recall that tight type derivations in the Open CbNeed multi type system are, in particular, defined to have a type context that only assigns variables to multi

types in the Inert class. For example, consider an *inert* program p, and note that the tight type derivation $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} p : M$ given for it in Proposition 7.1.9 (Tight typability of Open CbNeed-normal forms) is such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$ and Γ is *inert*. That is, if $\mathsf{nv}(p) = \{x_1, \ldots, x_n\}$ then $\Gamma = \{x_1 : M_1; \ldots; x_n : M_n\}$, where $M_1, .., M_n \in \mathsf{Inert}$. Hence, while the precise cardinality of each one of $M_1, ..., M_n$ may vary, the structure remains the same: they are all in Inert.

This shall not be the case for Useful Open CbNeed. Namely, if q is a useful inert program, then we shall give it a tight type derivation $\Psi \rhd_U \Pi \vdash^{(m',e',r')} q : N$ such that $\mathsf{dom}(\Pi) = \mathsf{nv}(q)$ and Π is *tight*. This means that, if $\mathsf{nv}(q) = \{y_1, \ldots, y_n\}$ then $\Pi = \{y_1 : N_1; \ldots; y_n : N_n\}$, where $N_1, .., N_n \in \mathsf{Tight}$. In fact, we shall see that type contexts are a bit more limited than this: while indeed $\Gamma(x) \in \mathsf{Tight}$ for every $x \in \mathsf{ul}(p)$, it happens that every $x \in \mathsf{a}(p)$ shall be proven to satisfy that $\Gamma(x) \in \mathsf{Inert}$[1].

In this sense, the useless variables enjoy a greater degree of freedom with respect to applied ones. And this shall be incarnated in the form of what we call *choice functions*, which are functions of the form $f : \mathcal{V} \to \{\mathsf{Inert}, \mathsf{Abs}\}$, where $\mathcal{V}$ is a set of variables. Thus, given a useful inert program p, we shall use choice functions to generalize the tight type derivation that we give for p. That is, for every $f : \mathsf{ul}(p) \to \{\mathsf{Inert}, \mathsf{Abs}\}$, we shall give a tight type derivation $\Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} p : O$ satisfying that $\Delta(x) \in \mathsf{Tight}$ for every $x \in \mathsf{ul}(p)$, and that $\Delta(x) \in \mathsf{Inert}$ for every $x \in \mathsf{a}(p)$.

To conclude, let us consider an example of the way choice functions are used when deriving tight type derivations for useful inert programs. First, let $p \coloneqq (x\, y, [y \leftarrow z\, z])$ and $q \coloneqq (x\, y, [y \leftarrow \lambda z.z])$. Note that y is bound in p to inert term $z\, z$, while it is bound in q to value $\lambda z.z$; these are two significantly distinct categories of normal terms. Next, note that a tight type derivation for p must be of the following form

$$\dfrac{\dfrac{\dfrac{\dfrac{M \in \mathsf{Inert}}{x : M \vdash^{(0,0,0)} x : M}\,\mathsf{ax}_T \quad \dfrac{}{y : [\mathsf{inert}] \vdash^{(0,0,0)} y : [\mathsf{inert}]}\,\mathsf{ax}_T}{x : M; y : [\mathsf{inert}] \vdash^{(0,0,1)} x\, y : M}\,\mathsf{app}_i}{x : M; y : [\mathsf{inert}] \vdash^{(0,0,1)} (x\, y, \epsilon) : M}\,\mathsf{Lift} \quad \dfrac{\dfrac{\dfrac{}{z : [\mathsf{inert}] \vdash^{(0,0,0)} z : [\mathsf{inert}]}\,\mathsf{ax}_T \quad \dfrac{O \in \{[\mathsf{inert}], [\mathsf{abs}]\}}{z : O \vdash^{(0,0,0)} z : O}\,\mathsf{ax}_T}{z : ([\mathsf{inert}] \uplus O) \vdash^{(0,0,1)} z\, z : [\mathsf{inert}]}\,\mathsf{app}_i}{x : M\,;\, z : ([\mathsf{inert}] \uplus O) \vdash^{(0,0,2)} (x\, y, [y \leftarrow z\, z]) : M}\,\mathrm{ES}$$

while a tight type derivation for q must be of the following form

$$\dfrac{\dfrac{\dfrac{\dfrac{M \in \mathsf{Inert}}{x : M \vdash^{(0,0,0)} x : M}\,\mathsf{ax}_I \quad \dfrac{}{y : [\mathsf{abs}] \vdash^{(0,0,0)} y : [\mathsf{abs}]}\,\mathsf{ax}_I}{x : M; y : [\mathsf{abs}] \vdash^{(0,0,1)} x\, y : M}\,\mathsf{app}_i}{x : M; y : [\mathsf{abs}] \vdash^{(0,0,1)} (x\, y, \epsilon) : M}\,\mathsf{Lift} \quad \dfrac{\dfrac{}{\emptyset \vdash^{(0,0,0)} \lambda z.z : \mathsf{abs}}\,\mathsf{abs}}{\emptyset \vdash^{(0,0,0)} \lambda z.z : [\mathsf{abs}]}\,\mathsf{many}}{x : M \vdash^{(0,0,2)} (x\, y, [y \leftarrow z\, z]) : M}\,\mathrm{ES}$$

It is particularly interesting to notice that both sub-type derivations for $(x\, y, \epsilon)$, namely those ending in type judgements $x : M; y : [\mathsf{inert}] \vdash^{(0,0,1)} (x\, y, \epsilon) : M$ and $x : M; y : [\mathsf{abs}] \vdash^{(0,0,1)} (x\, y, \epsilon) : M$, are *tight* type derivations, but that their type contexts are different. And most importantly, they are both needed: one to provide a tight type derivation for p, and the other one to provide a tight type derivation for q.

Let us now present the Lemmas, leaving further technicalities for the Appendix.

Lemma 9.1.12 (Tight typability of values)**.**

Let $v \in \mathsf{Val}$ and $M \in \mathsf{Abs}$. Then there exists tight type derivation $\Phi \rhd_U \Gamma \vdash^{(0,0,|v|_{\mathsf{nd}})} v : M$ such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(v)$.

[1] Recall that $\mathsf{nv}(p) = \mathsf{u}(p) \cup \mathsf{a}(p)$, by Lemma 8.1.1 (Unapplied, applied and needed variables), which may be rewritten as $\mathsf{nv}(p) = \mathsf{ul}(p) \cup \mathsf{a}(p)$.

(Click here to see the complete proof in the Technical Appendix)

Lemma 9.1.13 (Tight typability of useful inert terms)**.**
Let i^+ be a useful inert term, $M \in \mathsf{Inert}$, and $f : \mathsf{ul}(i^+) \to \{\mathsf{Inert}, \mathsf{Abs}\}$ be a choice function. Then there exists

- *type context Γ such that $\mathsf{dom}(\Gamma) = \mathsf{a}(i^+)$ and $\mathsf{inert}(\Gamma)$;*
- *type context Π such that $\mathsf{dom}(\Pi) = \mathsf{ul}(i^+)$ and $\Pi(z) \in f(z)$, for every $z \in \mathsf{dom}(\Pi)$;*
- *tight type derivation $\Phi \rhd_U \Gamma; \Pi \vdash^{(0,0,|i^+|_{\mathsf{nd}})} i^+ : M$.*

(Click here to see the complete proof in the Technical Appendix)

Lemma 9.1.14 (Tight typability of generalized variables)**.**
Let $p \in \mathcal{PR}$ be such that $\mathsf{genVar}_{\#}(p)$ and $M \in \mathsf{Tight}$. Then there exists tight type derivation $\Phi \rhd_U \Gamma \vdash^{(0,0,|p|_{\mathsf{nd}})} p : M$ such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{genVar}_{\#}(p)$.

□

Lemma 9.1.15 (Tight typability of useful abstraction programs)**.**
Let $p \in \mathcal{PR}$ be such that $\mathsf{uabs}(p)$. Then there exists tight type derivation $\Phi \rhd_U \Gamma \vdash^{(0,0,|p|_{\mathsf{nd}})} p : [\mathsf{abs}]$ such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{uabs}(p)$.

□

Lemma 9.1.16 (Tight typability of useful inert programs)**.**
Let $p \in \mathcal{PR}$ be such that $\mathsf{uinert}(p)$, and let $f : \mathsf{ul}(p) \to \{\mathsf{Inert}, \mathsf{Abs}\}$ be a choice function. Then there exists

- *type context Γ such that $\mathsf{dom}(\Gamma) = \mathsf{a}(p)$ and $\mathsf{inert}(\Gamma)$;*
- *type context Π such that $\mathsf{dom}(\Pi) = \mathsf{ul}(p)$ and $\Pi(z) = f(z)$, for every $z \in \mathsf{dom}(\Pi)$;*
- *tight type derivation $\Phi \rhd_U \Gamma; \Pi \vdash^{(0,0,|p|_{\mathsf{nd}})} p : [\mathsf{inert}]$.*

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the derivation of $\mathsf{uinert}(p)$.

□

Finally, we get the desired

Proposition 9.1.17 (Tight typability of Useful Open CbNeed-normal forms)**.**
Let $p \in \mathcal{PR}$ be such that $\mathsf{unorm}(p)$. Then there exists a tight type derivation $\Phi \rhd_U \Gamma \vdash^{(0,0,|p|_{\mathsf{nd}})} p : M$ such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.

Proof. (Click here to see the complete proof in the Technical Appendix)

By Lemma 8.2.1 (Disjointness of generalized variables, useful abstraction programs and useful inert programs), either $\mathsf{genVar}_{\#}(p)$, $\mathsf{uabs}(p)$ or $\mathsf{uinert}(p)$. The statement follows by induction on the derivation of each of these predicates, proceeding by case analysis on the last derivation rule.

□

The removal Lemma takes the expected shape:

Lemma 9.1.18 (Linear Removal for Useful Open CbNeed)**.**
Let $x \in \mathsf{Var}$ and $v \in \mathsf{Val}$ such that $x \notin \mathsf{fv}(v)$.

1. *Let $\mathcal{H}^{@}$ be such that $x \notin \mathsf{a}(\mathcal{H}^{@})$, and let*

$$\Phi \triangleright_U \Gamma; x : M \vdash^{(m,e,r)} \mathcal{H}^{@}\langle v \rangle : N$$

be such that $N \neq \mathbf{0}$, and $\mathsf{tight}_{\Gamma}(\mathsf{a}(\mathcal{H}^{@}))$.
Then there exist type derivations

$$\Psi \triangleright_U \Pi \vdash^{(m',e',r')} v : O$$
$$\Theta \triangleright_U \Delta; x : (M \uplus O) \vdash^{(m'',e'',r'')} \mathcal{H}^{@}\langle x \rangle : N$$

such that
   - $\Gamma = \Pi \biguplus \Delta$, *and*
   - $(m' + m'', e' + e'', r' + r'') = (m, e + 1, r)$.

2. *Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be such that $x \notin (\mathcal{A} \cup \mathsf{dom}(P))$ and $\mathsf{fv}(v) \cap \mathsf{dom}(P) = \emptyset$, and let*

$$\Phi \triangleright_U \Gamma; x : M \vdash^{(m,e,r)} P\langle v \rangle : N$$

be such that $N \neq \mathbf{0}$, and $\mathsf{tight}_{\Gamma}(\mathcal{A})$.
Then there exist multi type $O \neq \mathbf{0}$ and type derivations

$$\Psi \triangleright_U \Pi \vdash^{(m',e',r')} v : O$$
$$\Theta \triangleright_U \Delta; x : (M \uplus O) \vdash^{(m'',e'',r'')} P\langle x \rangle : N$$

such that
   - $\Gamma = \Pi \biguplus \Delta$, *and*
   - $(m' + m'', e' + e'', r' + r'') = (m, e + 1, r)$.

Proof. (Click here to see the complete proof in the Technical Appendix)

1. By structural induction on the applicative term context $\mathcal{H}^{@}$.
2. By structural induction on the exponential evaluation context $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$.

□

Unsurprisingly, it is first required to prove a subject expansion Lemma regarding term contexts before lifting the result to the general $\to_{\mathsf{und}}$ case. This is analogous to what happens in the Subject Expansion for Open CbNeed[2] and what happens in the Subject Reduction property for Useful Open CbNeed[3].

[2]See Proposition 7.1.12 (Quantitative Subject Expansion for Open CbNeed), which builds on Lemma 9.1.19 (Quantitative Subject Expansion for $\to_{\mathsf{om}}$ in term contexts).

[3]See Proposition 9.1.10 (Quantitative Subject Reduction for Useful Open CbNeed), which builds on Lemma 9.1.9 (Quantitative Subject Reduction for $\to_{\mathsf{um}}$ in term contexts).

Lemma 9.1.19 (Quantitative Subject Expansion for $\to_{\mathsf{um}}$ in term contexts)**.**
Let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} (\mathcal{H}\langle u\rangle, [x\leftarrow s]) : M$ such that $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}))$. Then there exists $\Phi' \rhd_U \Gamma \vdash^{(m+1,e,r)} \mathcal{H}\langle(\lambda x.u)s\rangle : M$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on $\mathcal{H}$.

□

Proposition 9.1.20 (Quantitative Subject Expansion for Useful Open CbNeed)**.**
Let $\Phi' \rhd_U \Gamma \vdash^{(m,e,r)} p' : M$ be a tight type derivation.
1. Multiplicative*: if $p \to_{\mathsf{um}} p'$, then there exists a type derivation $\Phi \rhd_U \Gamma \vdash^{(m+1,e,r)} p : M$.*
2. Exponential*: if $p \to_{\mathsf{ue}} p'$, then there exists a type derivation $\Phi \rhd_U \Gamma \vdash^{(m,e+1,r)} p : M$.*

Proof. (Click here to see the complete proof in the Technical Appendix)
The multiplicative case is proven by induction on $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ such that $p = P\langle s\rangle \to_{\mathsf{um}} P\langle s'\rangle = p'$, using Lemma 9.1.19 (Quantitative Subject Expansion for $\to_{\mathsf{um}}$ in term contexts) for the base case.
The exponential case is proven by induction on $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ such that $p = P\langle x\rangle \to_{\mathsf{ue}} P\langle v^\alpha\rangle = p'$ and $p(x) = v$, using Lemma 9.1.18 (Linear Removal for Useful Open CbNeed) for the base case.

□

Finally,

Theorem 9.1.21 (Tight Completeness for Useful Open CbNeed)**.**
Given $p \in \mathcal{PR}$ such that $d : p \to^*_{\mathsf{und}} q$ for some q in $\to_{\mathsf{und}}$-normal form, there exists a tight type derivation $\Phi \rhd_U \Gamma \vdash^{(|d|_{\mathsf{m}},|d|_{\mathsf{e}},|q|_{\mathsf{nd}})} p : M$ where $\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the length of the $\to_{\mathsf{und}}$-normalizing sequence starting in p. The base case is covered by Proposition 9.1.17 (Tight typability of Useful Open CbNeed-normal forms), while the inductive one is given by application of the *i.h.* followed by Proposition 9.1.20 (Quantitative Subject Expansion for Useful Open CbNeed).

□

9.1.4 Useful Open CbNeed semantics

Following the reasonings given in Subsect. 7.1.3 (Open CbNeed semantics, page 97), let us define the *semantics of p for $\vec{x}$* with respect to the Useful Open CbNeed type system as

$$[\![t]\!]^{\text{Useful Open CbNeed}}_{\vec{x}} := \{((M_1,\dots,M_n), M) \mid \exists\, \Phi \rhd_U x_1 : M_n; \dots; x_n : M_n \vdash^{(m,e,r)} p : M, \text{ with } \Phi \textit{ tight}\}$$

Note that this restricted version of the Useful Open CbNeed semantics is such that:
- It is *invariant*, by Proposition 9.1.10 (Quantitative Subject Reduction for Useful Open CbNeed) and Proposition 9.1.20 (Quantitative Subject Expansion for Useful Open CbNeed).
- It is also *adequate*, by Theorem 9.1.11 (Tight Correctness for Useful Open CbNeed) and Theorem 9.1.21 (Tight Completeness for Useful Open CbNeed).
- Unfortunately, it is not compositional, as explained in Sect. 3.2 (Properties of multi type systems, page 28).

Chapter 10

Strong CbV

Introduction

Plotkin's call-by-value λ-calculus [Plo75] is at the heart of programming languages such as OCaml and proof assistants such as Coq. In the study of (functional) programming languages, call-by-value evaluation is usually weak and closed. In previous chapters, we took both these constraints and presented an elegantly framed version of a weak and closed CbV evaluation strategy, which we simply called "CbV"—revisit Chapter 4 (CbN, CbV and CbNeed) for the operational account of CbV, and Sect. 5.3 (Multi type system for CbV) for its type-theoretical account. Other names have been given to our CbV, like "*Closed CbV*" in [AG16].

Despite the utility this seemingly ideal setting of CbV may have—in particular in the functional programming languages world—one often needs to go beyond it. For instance, we may extend the definition of the reduction relation by including open terms. This has been called many different names; we use "Open CbV" here, following again the terminology in [AG16] where the authors present four different proposals for a CbV evaluation strategy working on open terms, all of which are shown to be "slightly different incarnations of the same immaterial setting", namely Open CbV, given that they are all termination-equivalent and all their normalizing reduction sequences have the same number of steps. The strategies they relate are the following

1. Accattoli and Paolini's *Value Substitution Calculus* [AP12], which is a term syntax for the proof-nets representation of Λ-terms according to the CbV translation to Linear Logic.
2. Carraro and Guerrieri's *shuffling calculus* [CG14], used to provide the first semantical characterization of solvable terms in a relational model.
3. Paolini and Ronchi Della Rocca's (unnamed) reduction relation extending call-by-value to a setting where terms may have free variables [PR99]. It was later reintroduced by Accattoli and Sacerdoti Coen as the *fireball calculus* in their [AS15] and used to study cost models. Furthermore, Grégoire and Leroy rediscovered it when improving the implementation of the Coq proof assistant in [GL02].
4. The *value sequent calculus*, the intuitionistic and CbV sub-calculus of Curien and Herbelin's $\overline{\lambda}\mu\tilde{\mu}$-calculus [CH00].

It should be remarked here that Open CbV remains a *weak* strategy, as reduction does not take place under λ-abstractions. One may then go a step further and define the reduction relation on possibly open terms *and* under λ-abstractions—*i.e.*, strong reduction, as discussed in Subsect. 2.1.5 (Degrees of generality in β-reduction)—which we here present as "Strong CbV". It is a carefully designed extension of (weak and close) CbV to strong reduction acting on possibly open Λ_{L}-terms,

bearing an exact quantitative correspondence with its semantics—as we shall see in Chapter 11 (Multi types for Strong CbV).

The need for Strong CbV arises, most notably, when describing the implementation model of the Coq proof assistant—as done by Grégoire and Leroy in [GL02]—but also from other motivations, such as denotational semantics [AP12; CG14; PR99; RP04], monad and continuation-passing style translations [DL07; HZ09; Mog89; SF93; SW97], bisimulations [Las05], partial evaluation [JGS93], Linear Logic proof nets [AG17], or cost models [AS15].

Unfortunately, it is well known that Plotkin's call-by-value λ-calculus has issues when reduction can take place under λ-abstractions and terms may be open. In this chapter, we give an operational account of Strong CbV, which turns out to be a successful embodiment of such an extension of Plotkin's call-by-value λ-calculus to possibly open terms and reduction going under λ-abstractions; the semantical studies of Strong CbV are left for next chapter.

More concretely put, we present here a Strong CbV-normalizing evaluation strategy $\to_{\mathsf{s}}$, in the sense that $\to_{\mathsf{s}}$ reaches a Strong CbV-normal form whenever there exists one. Moreover, our presentation of Strong CbV happens to be a conservative extension of Open CbV—that is, Open CbV reduction is a subset of Strong CbV, and the properties proven for the latter hold also for the former.

Let us clarify a point before continuing. A (weak) value is a λ-abstraction. In CbV, β-redexes can be fired only when the argument is a value, and values are CbV-normal forms; close CbV can thus be seen as a *call-by-normal-form* strategy. A similar property can be recovered also in Open CbV, although doing so is considerably delicate; this is discussed in the following Sect. 10.1 (Variants of Open CbV). In Strong CbV, however, such an essence is lost. To see why, let Ω denote the well-known diverging Λ-term, and note that

- Not all λ-abstractions are Strong CbV-normal forms. For instance, $\lambda x.\Omega$ diverges in Strong CbV, as it keeps reducing Ω indefinitely.
- $(\lambda y.z)\,\Omega$ is Strong CbV-weakly normalizing but not Strong CbV-strongly normalizing: there exists a Strong CbV-normalizing reduction sequence consisting in reducing the outermost β-redex[1], and a Strong CbV-divergent reduction sequence consisting in constantly reducing the function argument Ω.

In other words, in a strong setting, the function argument need not be *fully* Strong CbV-normalized, but only reduced to a (weak) value. This feature is mandatory to be conservative over CbV and Open CbV.

10.1 Variants of Open CbV

In the call-by-name case, defining the reduction relation on possibly open terms and going under λ-abstractions is harmless, the reason being that CbN does not impose function arguments of β-redexes to have any special form.

On the contrary, in the case of CbV-like strategies, even only turning to open terms—that is, even without going into strong reduction—is already delicate, in particular due to the fact that function arguments need to be reduced before contracting a β-redex. For instance, let us denote the extension of Plotkin's call-by-value λ-calculus to open Λ-terms by the reduction relation $\to_{\beta_{OCbV}}$,

[1] As a matter of fact, there exist infinitely many, different Strong CbV-reduction sequences reaching a normal form. Namely, the ones reducing Ω a finite number of steps and then finally reducing the outermost β-redex to z.

and consider $(\lambda x.x)(y(\lambda z.z)) \in \Lambda$: the function argument $(y(\lambda z.z))$ possesses no β-redexes and is thus in normal form, but since it is an application instead of a λ-abstraction, then $\to_{\beta_{OCbV}}$ is not defined for β-redexes like $(\lambda x.x)(y(\lambda z.z))$. This extension is in fact dubbed "naïve" by the authors in [AG16]; they replace the naïve extension by the more general (and correct) Open CbV, giving an operational account of it[2]. The same authors provide a semantical account of Open CbV in their [AG18], using the same approach we use throughout this work: multi type systems.

Needless to say, correctly extending Open CbV in [AG16] to strong reduction is even harder to achieve. While some fundamental properties, such as confluence and standardization, hold also in such cases—as showed by Plotkin's himself [Plo75]—others break as soon as one considers open terms. As pointed out by Paolini and Ronchi Della Rocca [Pao01; PR99; RP04], denotational semantics that are *adequate*[3] for (weak and close) CbV are no longer adequate for the extended settings. Roughly, there are terms that are semantically divergent—that is, terms whose interpretation is the bottom of the domain, which represents the divergent terms of the strategy—while they are normal forms with respect to Plotkin's rules—and should thus have a non-bottom interpretation.

The discrepancy can be seen from a logical viewpoint. These terms diverge also if seen as (recursively typed) Linear Logic proof nets, as pointed out by Accattoli [Acc15], or as terms in the computational interpretation of sequent calculus due to Curien and Herbelin [CH00]. One may even trace the problems of CbV to Plotkin's seminal paper, where he points out an asymmetry between CbN and CbV with respect to continuation-passing style translations. This fact led to a number of studies [DL07; HZ09; Mar+99; Mog89; SF93; SW97] that introduced many proposals of improved calculi for CbV.

It is interesting to comment here that the operational and semantical accounts given by Accattoli and Guerrieri in [AG16] and semantical [AG18], respectively, is complemented by the implementative studies in [AG17] studies given by the same authors. All these different way of looking at Open CbV have, moreover, been successfully connected at a quantitative level:

- The size of (a class of) type derivations in the multi type system in [AG18] matches exactly the length of Open CbV-normalizing reduction sequences, and
- The GLAMOUr abstract machine in [AG17] is *reasonable* with respect to Open CbV.

10.2 The Value Substitution Calculus

Here we define the Accattoli and Paolini's Value Substitution Calculus [AP12] (shortened to VSC) and a new evaluation strategy, defined with the VSC as its framework, called the Strong CbV evaluation strategy. The idea of the Value Substitution Calculus, as any other LSC, is that β-contraction is decomposed via ESs—that is, the multiplicative step—and that the *by-value* restriction on evaluation is on the substitution rule—that is, the exponential step. This is an alternative to what we discussed above with respect to the implementation of the *by-value* restriction: if the restriction were instead imposed on β-contraction—as is the case in (weak and close) CbV, where function arguments have to be λ-abstractions—then we would encounter the same problems as in Open CbV: stuck β-redexes.

[2]Therein, the authors call β-redexes like $(\lambda x.x)(y(\lambda z.z))$—for which the reduction relation is not defined—*stuck* β-redexes.

[3]Recall that a semantics is *adequate* when the interpretation of an expression is not the bottom of the domain if and only if the term normalizes. Note that for a CbV semantics, adequacy is somewhat mandatory, as any semantics for CbN provides a non-adequate semantics for CbV—interpreting a term with its CbN-interpretation if it CbV-normalizes and with bottom if it does not CbV-normalize—which does not model the CbV behavior.

The VSC has a strong connection with Linear Logic proof nets, from which some of the terminology is taken [Acc15], and induces a peculiar use of evaluation contexts in the rewriting rules.

A delicate point is whether to consider variables as values. If they are not, abstract machines are faster—see [AS14]. We treat both cases, excluding variables from values but adding a postponable rule for substituting variables.

10.2.1 Terms and rewriting rules

As expected, the syntax for terms is simply the Λ_{L}-terms. Let us recall the definition of LSC and substitution evaluation contexts, with which we provide a definition for the VSC:

$$\begin{array}{lrcl} \text{LSC evaluation contexts} & D, D' & ::= & \langle\cdot\rangle \mid Dt \mid tD \mid \lambda x.D \mid D[x{\leftarrow}t] \mid t[x{\leftarrow}D] \\ \text{Substitution contexts} & S, S' & ::= & \langle\cdot\rangle \mid S[x{\leftarrow}t] \end{array}$$

As usual, the rewriting rules are first defined at top level in the form of multiplicative and exponential root-steps, and then closed by (LSC) evaluation contexts. Much like (weak and close) CbV, both root-steps work up to a substitution context S—in other words, up to sharing.

One big difference with respect to CbV, however, is that in the VSC we distinguish two exponential—that, substitution—rules, one for values and one for variables. This corresponds to the fact that in the VSC the substitution process applies both for variables and values. These are known in the literature as *theoretical values*[4]:

$$\textsc{Theoretical values} \qquad v_{\mathrm{T}}, w_{\mathrm{T}} \quad ::= \quad \mathsf{Var} \cup \mathsf{Val}$$

Let us now give the reduction relations of the VSC:

$$\begin{array}{rrcl} \textsc{Multiplicative} & S\langle\lambda x.t\rangle u & \mapsto_{\mathsf{m}} & S\langle t[x{\leftarrow}u]\rangle \\ \textsc{Exponential steps for values} & t[x{\leftarrow}S\langle v\rangle] & \mapsto_{\mathsf{e}_\lambda} & S\langle t\{x{\leftarrow}v\}\rangle \\ \textsc{Exponential steps for variables} & t[x{\leftarrow}S\langle y\rangle] & \mapsto_{\mathsf{e}_{\mathsf{var}}} & S\langle t\{x{\leftarrow}y\}\rangle \end{array}$$

$$\textsc{Contextual closure} \qquad \frac{t \mapsto_a t'}{D\langle t\rangle \to_a D\langle t'\rangle} \quad a \in \{\mathsf{m}, \mathsf{e}_\lambda, \mathsf{e}_{\mathsf{var}}\}$$

$$\begin{array}{lrcl} \textsc{Notations} & \to_{\mathsf{e}} & ::= & \to_{\mathsf{e}_\lambda} \cup \to_{\mathsf{e}_{\mathsf{var}}} \\ & \to_{\mathsf{vsc}_\lambda} & ::= & \to_{\mathsf{m}} \cup \to_{\mathsf{e}_\lambda} \\ & \to_{\mathsf{vsc}} & ::= & \to_{\mathsf{vsc}_\lambda} \cup \to_{\mathsf{e}_{\mathsf{var}}} \quad = \quad \to_{\mathsf{m}} \cup \to_{\mathsf{e}} \end{array}$$

It is important to notice that the substitution process in Strong CbV is implemented in a meta-level fashion, and separately for variables in $\to_{\mathsf{e}_{\mathsf{var}}}$ and for λ-abstractions in $\to_{\mathsf{e}_\lambda}$.

Conforming to all the previous definitions of sizes for reduction sequences, given $d\colon t \to_{\mathsf{vsc}} u$ we write $|d|$ for the length of d and $|d|_a$ for the number of $\to_a$ steps in d, for every $a \in \{\mathsf{m}, \mathsf{e}_\lambda, \mathsf{e}_{\mathsf{var}}\}$.

The split in the exponential reduction relation into $\to_{\mathsf{e}_\lambda}$ and $\to_{\mathsf{e}_{\mathsf{var}}}$ is motivated by the following postponement property, and by the fact that $\to_{\mathsf{e}_{\mathsf{var}}}$ is trivially strongly normalizing.

[4]For more on theoretical values, see for example Accattoli and Sacerdoti Coen's [AS14]. The authors provide therein a comparison of the impact on efficiency when defining the substitution process in terms of theoretical values—that is, $\mathsf{Var} \cup \mathsf{Val}$—or with respect to *practical* values—which are simply Val and for which we may sometimes avoid the word *practical* altogether. Note that we have defined the substitution processes of all our evaluation strategies in this work, except for the one for VSC, to only act on practical values.

Proposition 10.2.1 (Irrelevance of $\to_{\mathsf{e_{var}}}$)**.**
Let $d\colon t \to^*_{\mathsf{vsc}} u$. Then there is $d'\colon t \to^*_{\mathsf{vsc}_\lambda} \to^*_{\mathsf{e_{var}}} u$ with $|d'|_{\mathsf{m}} = |d|_{\mathsf{m}}$.
Moreover, $\to_{\mathsf{vsc}_\lambda}$ is weakly normalizing (resp. strongly normalizing) on t if and only if $\to_{\mathsf{vsc}}$ is weakly normalizing (resp. strongly normalizing) on t.

(Click here to see the complete proof in the Technical Appendix)

The following is due to Accattoli and Paolini [AP12]:

Theorem 10.2.2 (Confluence of $\to_{\mathsf{vsc}}$)**.**
$\to_{\mathsf{vsc}}$ is confluent.

VSC-normal forms are similar to normal forms of the LSC, except that they can only have ES containing *strong inert terms*:

$$\begin{array}{rrcl} \textsc{Strong inert terms} & i_s & ::= & x \mid i_s f_s \mid i_s[x{\leftarrow}j_s] \\ \textsc{strong values} & v_s & ::= & \lambda x.f_s \\ \textsc{Strong fireballs} & f_s & ::= & i_s \mid v_s \mid f_s[x{\leftarrow}i_s] \end{array}$$

A strong inert term is *compound* if it is of the form $S\langle i_s f_s\rangle$, and a strong fireball is *super* if its ESs only contain compound strong inert terms. Note that strong inert terms can be written as $(\dots((xt_1)t_2)\dots)t_n$, with $n \geq 0$; we call x the *head* variable of the term. Finally, let us point out one of the key properties of strong inert terms: the application $i_s t$ of any $t \in \Lambda_{\mathsf{L}}$ to i_s does not create any new multiplicative-redexes other than the ones in t.

Proposition 10.2.3 (Syntactic characterization of VSC-normal forms)**.**
Let $t \in \Lambda_{\mathsf{L}}$. t is in $\to_{\mathsf{vsc}_\lambda}$-normal form (resp. $\to_{\mathsf{vsc}}$-normal form) if and only if t is a strong fireball (resp. strong super fireball).

Proof. (Click here to see the complete proof in the Technical Appendix)
All the proofs of this proposition proceed by structural induction on t.

□

10.2.2 Structural Equivalence

The theory of the VSC comes with a notion of structural equivalence $\equiv$, that equates terms differing only for the position of ES. The basic idea is that the action of an ES taking part in an exponential step depends on the position of the ES itself only for inessential details—as long as the scope of binders is respected—and can thus be abstracted away. A strong justification comes from the Linear Logic interpretation of the VSC, in which structurally equivalent terms translate to the same (recursively typed) proof net—see [Acc15].

Structural equivalence $\equiv$ is defined as the least equivalence relation on Λ_{L}-terms closed by all LSC evaluation contexts and generated by the following root cases:

$$\begin{array}{rcll} t[y{\leftarrow}s][x{\leftarrow}u] & \equiv_{\mathrm{com}} & t[x{\leftarrow}u][y{\leftarrow}s] & \text{if } y \notin \mathsf{fv}(u) \text{ and } x \notin \mathsf{fv}(s) \\ t\,s[x{\leftarrow}u] & \equiv_{@\mathrm{r}} & (ts)[x{\leftarrow}u] & \text{if } x \notin \mathsf{fv}(t) \\ t[x{\leftarrow}u]s & \equiv_{@\mathrm{l}} & (ts)[x{\leftarrow}u] & \text{if } x \notin \mathsf{fv}(s) \\ t[x{\leftarrow}u[y{\leftarrow}s]] & \equiv_{[\cdot]} & t[x{\leftarrow}u][y{\leftarrow}s] & \text{if } y \notin \mathsf{fv}(t) \end{array}$$

Extending the VSC with $\equiv$ results in a smooth system, as $\equiv$ commutes with evaluation, and can thus be postponed. Additionally, the commutation is *strong*, as it preserves the number and kind

of steps: we say that it is a *strong bisimulation* with respect to $\to_{\mathsf{vsc}}$. In particular, the equivalence is not needed to compute and it does not break, or make more complex, any property of the VSC.

Proposition 10.2.4 (Strong Bisimulation of $\equiv$ and $\to_{\mathsf{vsc}}$)**.**

Let $\mathsf{a} \in \{\mathsf{m}, \mathsf{e}_\lambda, \mathsf{e}_{\mathsf{var}}, \mathsf{sm}, \mathsf{se}_\lambda, \mathsf{se}_{\mathsf{var}}\}$. *If* $t \equiv u$ *and* $t \to_{\mathsf{a}} t'$, *then there exists* $u' \in \Lambda_{\mathsf{vsc}}$ *such that* $u \to_{\mathsf{a}} u'$ *and* $t' \equiv u'$.

(Click here to see the complete proof in the Technical Appendix)

Generally speaking, strong bisimulations interact very well with the underlying rewriting system: they preserve normal forms and can be postponed at the end of an evaluation, without affecting the number and the kind of steps.

10.3 The Strong CbV strategy

Similarly to the ordinary λ-calculus, the VSC is non-deterministic but confluent—its confluence is proved in [AP12]. We now proceed to define the "*strong*" evaluation strategy $\to_{\mathsf{s}}$ in the setting of the VSC that we show to be normalizing—that, a Λ_{L}-term $\to_{\mathsf{s}}$-reduces to its $\to_{\mathsf{vsc}}$-normal form whenever there exists one. Its role is analogous to the leftmost-outermost strategy of the λ-calculus—see [AL16]. A notable difference, however, is that the strategy is itself non-deterministic, but in a harmless way, as it is diamond. Something similar happens with the weak and closed CbV evaluation strategy: it is defined as a non-deterministic reduction relation, which satisfies the diamond property, in a Weak LSC setting—where reduction can take place anywhere except under λ-abstractions.

Our evaluation strategy relies heavily on rigid Λ_{L}-terms, which are just like strong inert terms—defined in Subsect. 10.2.1 (Terms and rewriting rules)—in that their shape is also $(\dots((xt_1)t_2)\dots)t_n$, with $n \geq 0$, except that there are no restrictions on $t_1, \dots, t_n$. Rigid terms are defined as follows:

$$\textsc{Rigid terms} \quad r, r' ::= x \mid rt \mid r[x \leftarrow r']$$

Note that strong inert terms are rigid. This fact is necessary in the technical development. Of course, the converse does not hold: take for instance $y\,(\lambda x.\delta\delta)$.

We now need the right notion of evaluation contexts for the Strong CbV evaluation strategy $\to_{\mathsf{s}}$. First, we define a notion of *Open CbV evaluation context*, that is used to define the open evaluation strategy[5]. The Strong CbV evaluation strategy is finally obtained as the contextual closure of the open evaluation strategy under *strong evaluation contexts*, the latter being defined by mutual induction with *rigid contexts*:

$$\begin{array}{rrcl} & \multicolumn{3}{c}{\textsc{Term evaluation contexts}} \\ \textsc{Open} & D, D' & ::= & \langle\cdot\rangle \mid Dt \mid tD \mid D[x \leftarrow t] \mid t[x \leftarrow D] \\ \textsc{Strong} & S, S' & ::= & \langle\cdot\rangle \mid \lambda x.S \mid t[x \leftarrow R] \mid S[x \leftarrow r] \mid R \\ \textsc{Rigid} & R, R' & ::= & rS \mid Rt \mid R[x \leftarrow r] \mid r[x \leftarrow R] \end{array}$$

$$\textsc{Open strategy} \quad \frac{t \mapsto_a t'}{D\langle t\rangle \to_{\mathsf{wa}} D\langle t'\rangle}\ a \in \{\mathsf{m}, \mathsf{e}_\lambda, \mathsf{e}_{\mathsf{var}}\}$$

$$\textsc{Strong strategy} \quad \frac{t \to_{\mathsf{wa}} t'}{S\langle t\rangle \to_{\mathsf{sa}} S\langle t'\rangle}\ a \in \{\mathsf{m}, \mathsf{e}_\lambda, \mathsf{e}_{\mathsf{var}}\}$$

[5]The Open CbV evaluation contexts in this chapter are not to be confused with the open evaluation contexts from Chapter 6 (Open CbNeed), and similarly for the open evaluation strategy and the Open CbNeed one.

NOTATIONS

$$\begin{aligned} \to_{\mathsf{w}_\lambda} &:= \to_{\mathsf{wm}} \cup \to_{\mathsf{we}_\lambda} & \to_{\mathsf{s}_\lambda} &:= \to_{\mathsf{sm}} \cup \to_{\mathsf{se}_\lambda} \\ \to_{\mathsf{w}} &:= \to_{\mathsf{w}_\lambda} \cup \to_{\mathsf{we}_{\mathsf{var}}} & \to_{\mathsf{s}} &:= \to_{\mathsf{s}_\lambda} \cup \to_{\mathsf{se}_{\mathsf{var}}} \\ \to_{\mathsf{we}} &:= \to_{\mathsf{we}_\lambda} \cup \to_{\mathsf{we}_{\mathsf{var}}} & \to_{\mathsf{se}} &:= \to_{\mathsf{se}_\lambda} \cup \to_{\mathsf{se}_{\mathsf{var}}} \end{aligned}$$

Given a reduction sequence $d : t \to_{\mathsf{s}}^* u$, we note by $|d|$ the length of d, with $|d|_\lambda$ the number of $\to_{\mathsf{s}_\lambda}$ in d, and by $|d|_{\mathsf{m}}$ the number of $\to_{\mathsf{sm}}$ in d.

A crucial fact about the Strong CbV evaluation strategy is that $\to_{\mathsf{s}}$ is defined using $\to_{\mathsf{w}}$, both because the $\to_{\mathsf{w}}$ is the basis for $\to_{\mathsf{s}}$ and because the first production of strong evaluation contexts is the context hole $\langle\cdot\rangle$. Hence, all properties of $\to_{\mathsf{s}}$ are built over properties of $\to_{\mathsf{w}}$. Nonetheless, we omit statements specifically targeting $\to_{\mathsf{w}}$, as they are taken from [AG16].

Another evident fact is that Strong CbV is non-deterministic. However, it enjoys the diamond property:

Proposition 10.3.1 (Diamond property for Strong CbV)**.**
$\to_{\mathsf{s}}$ *is diamond.*

(Click here to see the complete proof in the Technical Appendix)

With I the identity combinator, let us give a few examples showing what the $\to_{\mathsf{s}}$ evaluation strategy is like:

1. Since $\to_{\mathsf{s}}$ is based on $\to_{\mathsf{w}}$, then
   - $(\mathsf{I}\mathsf{I})(\mathsf{I}\mathsf{I}) \to_{\mathsf{s}} (z[z\leftarrow\mathsf{I}])(\mathsf{I}\mathsf{I})$,
   - $(\mathsf{I}\mathsf{I})(\mathsf{I}\mathsf{I}) \to_{\mathsf{s}} (\mathsf{I}\mathsf{I})(z[z\leftarrow\mathsf{I}])$,
   - $(\mathsf{I}\mathsf{I})[x\leftarrow\mathsf{I}\mathsf{I}] \to_{\mathsf{s}} (z[z\leftarrow\mathsf{I}])[x\leftarrow\mathsf{I}\mathsf{I}]$, and
   - $(\mathsf{I}\mathsf{I})[x\leftarrow\mathsf{I}\mathsf{I}] \to_{\mathsf{s}} (\mathsf{I}\mathsf{I})[x\leftarrow(z[z\leftarrow\mathsf{I}])]$.
2. Since $x(\mathsf{I}\mathsf{I})$ is a rigid term and $\lambda y.\langle\cdot\rangle$ is a strong evaluation context, then $(x(\mathsf{I}\mathsf{I}))(\lambda y.\langle\cdot\rangle)$ is a rigid evaluation context. Then it is also a strong evaluation context and so
$$(x(\mathsf{I}\mathsf{I}))(\lambda y.\mathsf{I}\mathsf{I}) \to_{\mathsf{s}} (x(\mathsf{I}\mathsf{I}))(\lambda y.(z[z\leftarrow\mathsf{I}]))$$
3. Extending the previous example, note that $t[\tilde{x}\leftarrow(x(\mathsf{I}\mathsf{I}))(\lambda y.\langle\cdot\rangle)]$ is a strong evaluation context. Hence,
$$t[\tilde{x}\leftarrow(x(\mathsf{I}\mathsf{I}))(\lambda y.\mathsf{I}\mathsf{I})] \to_{\mathsf{s}} t[\tilde{x}\leftarrow(x(\mathsf{I}\mathsf{I}))(\lambda y.(z[z\leftarrow\mathsf{I}]))]$$
4. Since $\tilde{x}\tilde{x}$ is a rigid term, then $(x(\mathsf{I}\mathsf{I}))(\lambda y.\langle\cdot\rangle)[\tilde{y}\leftarrow\tilde{x}\tilde{x}]$ is a strong evaluation context. Then,
$$(x(\mathsf{I}\mathsf{I}))(\lambda y.(\mathsf{I}\mathsf{I}))[\tilde{y}\leftarrow\tilde{x}\tilde{x}] \to_{\mathsf{s}} (x(\mathsf{I}\mathsf{I}))(\lambda y.(z[z\leftarrow\mathsf{I}]))[\tilde{y}\leftarrow\tilde{x}\tilde{x}]$$

Finally, in order to prove the normalization property of Strong CbV with respect to the VSC, we need the following property:

Proposition 10.3.2 (Fullness of Strong CbV)**.**
Let $t \in \Lambda_{\mathsf{L}}$. If t is in $\to_{\mathsf{s}_\lambda}$-normal form (resp. $\to_{\mathsf{s}}$-normal form) then it is a strong fireball (resp. a strong super fireball).

(Click here to see the complete proof in the Technical Appendix)

Chapter 11

Multi types for Strong CbV

Introduction

We now present a Linear Logic-based multi type system for Strong CbV, from which we obtain an adequate and invariant semantics. We build on different type systems for call-by-value evaluation, as well as standard typing techniques in the literature.

The very basis of a multi type system for Strong CbV is the CbV multi type system we presented in Sect. 5.3 (Multi type system for CbV), which corresponds to the type system in Ehrhard's [Ehr12], where he provides an adaptation to Plotkin's CbV λ-calculus of de Carvalho's System R for CbN [Car07; Car18][1].

Other works crucially contributing to the foundation of our Strong CbV system are Guerrieri's [Gue18] and Accattoli and Guerrieri's [AG18]. Therein, the authors show that the same multi type system presented by Ehrhard in [Ehr12] for weak and close call-by-value evaluation can be used to characterize termination of Open CbV. In addition, it induces an *adequate* semantics, thus solving the inadequacy issue of the naïve extension of Plotkin's CbV λ-calculus to the open setting—see the Introduction to Chapter 10 (Strong CbV).

Besides providing exact bounds to the number of steps to Open CbV-normal forms, Accattoli and Guerrieri further use multi types type derivations to provide the exact size of Open CbV-normal forms. This is a novel feature of the Open CbV system with respect to the system for (weak and close) CbV, considering that the normal forms in the latter have a trivial size—since they are all answers, as proven in Proposition 4.6.1.2 (Syntactic characterization of closed normal forms - CbV).

As far as typing techniques are concerned, we use the *shrinking* typing technique, following the presentation given in [AGK20] for leftmost-outermost evaluation[2], adapted to the Strong CbV system. This technique is different from the notion of tight type derivations given for all of the previous cases, and has proved to be useful in the lifting of the semantical study of Open CbV in [AG18] into strong reduction. The name "*shrinking*" is due to the fact that we can ensure that type derivations satisfying the shrinking constraint indeed *shrink*—that is, they decrease in size—at each Strong CbV-step.

Furthermore, we use multi types to give our main operational result: the normalization of the

[1] More concretely, Ehrhard's system is the one obtained by translating the CbV λ-calculus—in its Intuitionistic Propositional Logic form, via de Curry-Howard correspondence—into Linear Logic, and then interpreting it according to the relational semantics of Linear Logic. It is also strongly related to other denotational semantics for CbV based on Linear Logic, such as Scott domains and coherent semantics.

[2] In fact, shrinking types have been extensively used in the literature to characterize leftmost evaluation; see for instance [BKV17; Car18; Kri93].

$\to_{\mathsf{s}}$ strategy with respect to the Value Substitution Calculus.

11.1 Shrinkingness

We begin by giving the mutually recursive definition of *linear* and *multi* types adapted for the Strong CbV multi type system:

$$\begin{array}{rl} \textsc{Strong CbV linear types} & L, L' ::= X \mid M \multimap N \\ \textsc{Strong CbV multi types} & M, N ::= [L_i]_{i\in I} \quad \text{(with I finite)} \end{array}$$

where X is any constant serving as a base or atomic type—besides the empty multi type $\mathbf{0}$, that is.

We can already see a key differentiating point between our linear and multi types and the ones for Open CbV in [AG18; Gue18], namely the presence of the ground type X in ours. This is not to be confused with the unavoidable role that tight constants play in the systems characterizing CbN, CbV and CbNeed-normalization—see Chapter 5 (Multi types for CbN, CbV and CbNeed)—because the class of type derivations isolated in the Strong CbV case is expressed in terms of shrinking types instead, as announced in the Introduction to this chapter. The reason then for the presence of constant X is that:

1. Any definition of shrinking types requires a type different from $\mathbf{0}$ to be used as a base/atomic type, in particular for the positive atomic occurrences; see Definition 49 (Shrinkingness and co-shrinkingness) below.
2. In Open CbV, normalizable and erasable terms coincide, and are all characterized by being typable with $\mathbf{0}$. But this is not the case in Strong CbV—see the explanations on $\lambda x.\Omega$ and $(\lambda y.z)(\lambda z.\Omega)$ in the Introduction of Chapter 10 (Strong CbV).

Type systems which do not rely at all on constants in the isolation of a class of minimal type derivations—that is, type derivations providing *exact* bounds on the normalization process—are called "*traditional*". The Strong CbV system in this chapter is one such system.

Of course, once we go beyond the idealistic setting of weak and close reduction, the isolation of a class of type derivations providing minimal quantitative information in their indices becomes considerably more difficult. We now proceed to give one for the Strong CbV system, which is based on the idea that the empty multi type $\mathbf{0}$ should be restricted via a notion of *polarity*—namely, positive and negative occurrences of types.

Definition 48 (Positive and negative occurrences of types)**.**

Let A be a linear type, a multi type, or a type context. The sets $\mathsf{Occ}_+(A)$ and $\mathsf{Occ}_-(A)$ of *positive* and *negative occurrences* in A are defined by mutual induction by (where L is a linear or a multi type, $\odot, \overline{\odot} \in \{+,-\}$ and $\odot \neq \overline{\odot}$):

$$\frac{}{L \in \mathsf{Occ}_+(L)} \qquad \frac{L \in \mathsf{Occ}_\odot(L_i)}{L \in \mathsf{Occ}_\odot([L_1,\dots,L_i,\dots,L_n])}$$

$$\frac{L \in \mathsf{Occ}_{\overline{\odot}}(M) \text{ or } L \in \mathsf{Occ}_\odot(N)}{L \in \mathsf{Occ}_\odot(M \multimap N)} \qquad \frac{L \in \mathsf{Occ}_\odot(M) \text{ or } L \in \mathsf{Occ}_\odot(\Gamma)}{L \in \mathsf{Occ}_\odot(x:M,\Gamma)}$$

Let us now define the class of type derivations providing minimal indices

Definition 49 (Shrinkingness and co-shrinkingness)**.**

Let M be a multi type. Then

1. M is *shrinking* (resp. *unitary shrinking*) if $|N| \geq 1$ (resp. $|N| = 1$) for all multi types $N \in \mathsf{Occ}_+(M)$;
2. M is *co-shrinking* (resp. *unitary co-shrinking*) if $|N| \geq 1$ (resp. $|N| = 1$) for all multi types $N \in \mathsf{Occ}_-(M)$.

Additionally, type context $\Gamma \coloneqq x_1 : M_1; \ldots; x_n : M_n$ is *co-shrinking* (resp. *unitary co-shrinking*) if each M_i is co-shrinking (resp. unitary co-shrinking).

Finally, type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash t : M$ is *shrinking* (resp. *unitary shrinking*) if Γ is co-shrinking (resp. unitary co-shrinking) and M is shrinking (resp. unitary shrinking).

For examples of (co-)shrinkingness, consider the following:

- $\mathbf{0} \in \mathsf{Occ}_-(\mathbf{0} \multimap N)$ and $N \in \mathsf{Occ}_+(\mathbf{0} \multimap N)$. Hence, $\mathbf{0} \in \mathsf{Occ}_-([L_1, \ldots, \mathbf{0} \multimap N, \ldots, L_n])$ and $N \in \mathsf{Occ}_+([L_1, \ldots, \mathbf{0} \multimap N, \ldots, L_n])$
- $M = [\mathbf{0} \multimap [X, X], [X]]$ is a shrinking multi type. However, it is not unitary shrinking, because $[X, X] \in \mathsf{Occ}_+(M)$ but $|[X, X]| = 2$. And it is not co-shrinking either, because $\mathbf{0} \in \mathsf{Occ}_-(M)$ but $|\mathbf{0}| = 0$. Consequently, for any given *shrinking* type context Γ, we have that $\Gamma \biguplus \{x : M\}$ is shrinking, but it is not unitary (because M is not unitary) nor is it co-shrinking (because M is not co-shrinking).
- Let Γ be co-shrinking but not unitary, M be non-shrinking, and N be unitary shrinking. Then, $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ is not shrinking (because M is not shrinking), and $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} u : N$ is shrinking but not unitary (because Γ is not unitary).

Finally, notice that unitary shrinkingness (resp. unitary co-shrinkingness) implies shrinkingness (resp. co-shrinkingness), but the converse is false. Moreover, note that $[X]$ is both unitary shrinking and unitary co-shrinking, while $\mathbf{0}$ is unitary co-shrinking but not shrinking, and $[\mathbf{0} \multimap [X]]$ is unitary shrinking but not co-shrinking.

While the most important notion to retain from Definition 49 is the one of (unitary) shrinking type derivations, sometimes we shall relax this definition in order for inductions to go through in proofs. For example, we shall see that while the Subject Reduction and Subject Expansion properties for the Strong CbV multi type system always assume a co-shrinking type context, they do *not always* assume that the inferred type—that is, the type on the right of the type judgement—is a unitary shrinking one.

11.2 Multi type system for Strong CbV

Let us proceed to the definition of the Strong CbV multi type system:

Definition 50 (The Strong CbV multi type system)**.**

The typing rules for the Strong CbV multi type system are in Fig. 11.1. As done for previous systems, we use a specific notation, in this case of the form

$$\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$$

to express that type derivation Φ, formed with the rules in Fig. 11.1, ends in type judgement $\Gamma \vdash^{(m,s)} t : M$. The same notation applies for when the inferred type is a Strong CbV linear type.

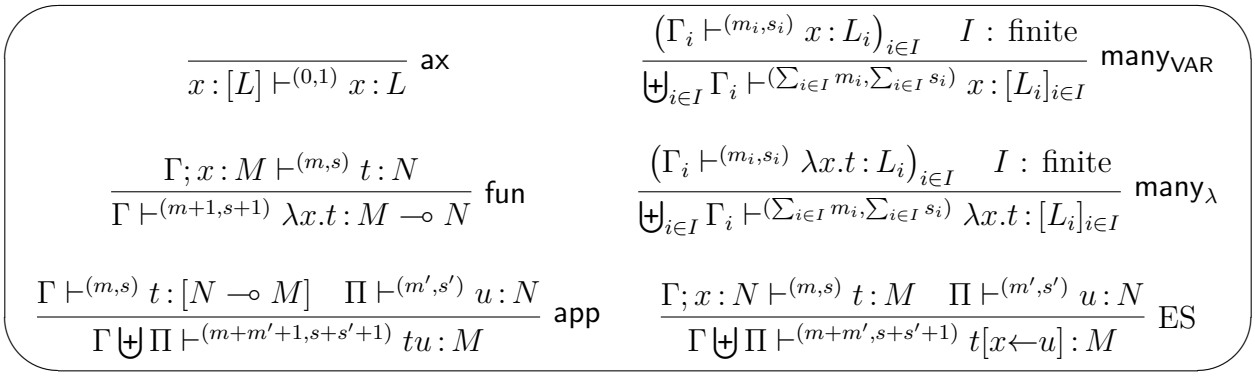

Figure 11.1: Type system for Strong CbV evaluation.

The typing rules in Fig. 11.1 are actually the same as in [Ehr12], up to the fact that they are extended to encompass the use of ESs in the calculus, and adapted to a split formulation of values, typing variables and λ-abstractions separately.

Indices in the Strong CbV multi type system. Given type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t:M$, we say that m is the *multiplicative* index of type judgement $\Gamma \vdash^{(m,s)} t:M$ and of type derivation Φ. It provides information on the number of multiplicative steps of Strong CbV-normalizing sequences starting at t and on the size of the corresponding Strong CbV-normal form.

More concretely, given $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t:M$, the Correctness theorem shows that $m \geq 2|d|_{\mathsf{m}} + |u|_{\mathrm{S}}$, where $d : t \to_{\mathsf{s}}^{*} u$ and u is in $\to_{\mathsf{s}}$-normal form. The presence of factor 2 multiplying $|d|_{\mathsf{m}}$ comes from the fact that both typing rules fun and app increase the multiplicative index. This is contrast with the Open CbNeed and Useful Open CbNeed multi type systems, where the multiplicative index was only increased by the app and app$_{\mathrm{gc}}$ typing rules, while the size index was only increased by typing rule app$_i$. This disentangled presentation of the number of reduction steps and the size of the normal form in the Open CbNeed and Useful Open CbNeed system cannot be so easily achieved in the Strong CbV multi type system for two reasons:

- The presence of the *tight* constants in the Open CbNeed and Useful Open CbNeed systems help us neatly separate applications that contribute to the number of multiplicative steps from those that contribute to the size of the normal form.
- The Strong CbV strategy performs *strong* reduction, and so the size of λ-abstractions must depend on its body. For instance, $|\lambda x.x|_{\mathrm{S}} = 1$ and $|\lambda x.\lambda y.y|_{\mathrm{S}} = 2$, while in Open CbNeed or in Useful Open CbNeed both expressions have null size.

On the other side, we say that s is the *general size* index of $\Gamma \vdash^{(m,s)} t:M$ and of Φ. Instead of providing quantitative information on Strong CbV-reduction sequences starting at t, the general size index serves the unique purpose of keeping count of all the typing rules, except for many$_{\mathsf{VAR}}$ and many$_{\lambda}$. This count is used in the proof of the Correctness theorem for Strong CbV, which is proven by induction on the general size index of the type derivation in question[3].

[3]The many$_{\mathsf{VAR}}$ and many$_{\lambda}$ being of a rather structural nature, we do not need to count them to provide the general size index needed in the Correctness theorem.

11.2.1 Strong CbV measurements

The correctness theorem for Strong CbV follows roughly the same schema as for the previous systems.

In particular, we prove that any $\to_{\mathsf{s}}$-reduction sequence starting in a typable Λ_{L}-term must be finite by showing that the general size index decreases at each $\to_{\mathsf{s}}$-step, as we just explained. Such decrement is proven in the Subject Reduction property, just like we did in all the other case studies with respect to some of the indices in the corresponding type derivations. This, however, provides a possibly *lax* upper bound on the $\to_{\mathsf{s}}$-normalization process, but we are in fact interested in attaining the highest level of adjustment of the type system with respect to the strategy: we would like to obtain *exact* bounds on the $\to_{\mathsf{s}}$-normalization process.

Regarding the weak and close strategies, we achieved this by identifying a class of type derivations (separately for each of the type systems) giving exact bounds on the normalization process. Recall, for instance, Proposition 5.2.2 (Typing properties of CbN-normal forms), which states that any type derivation for a $\to_{\mathrm{CbN}}$-normal form with norm as the right type gives null indices, corresponding precisely to the number of remaining $\to_{\mathrm{CbN}}$-steps to normal form.

We would therefore like an analogous Proposition for the Strong CbV case. Unfortunately, this cannot be achieved as simply as it was for the weak and close strategies. The reason is that the technique by which we count $\to_{\mathsf{sm}}$-steps, on the one hand, and the technique used to measure the size of $\to_{\mathsf{s}}$-normal forms, on the other hand, is the same. That is, there is no neat way to differentiate them:

- First, note that if $t \in \Lambda_{\mathsf{L}}$ is in $\to_{\mathsf{s}}$-normal form, then by Proposition 10.3.2 (Fullness of Strong CbV) it must be a strong super fireball. This means that $\to_{\mathsf{s}}$-normal forms are of an arbitrarily high complexity; in particular, strong inert terms have an arbitrarily big number of applicants. In contrast, recall that weak and close normal forms are answers.
  Now, since applications are only typable via the app rule, then the multiplicative index of any type derivation for t must be greater or equal to the number of application sub-terms in t. For example, any type derivation for $(x_1x_2)x_3 \in \Lambda_{\mathsf{L}}$ must have a multiplicative index that is greater or equal to 2, just like any type derivation for $(x_1x_2)[x_1 \leftarrow (x_3x_4)]$.
  Once again, this is in sheer contrast with what happens in the weak and close systems, where there exists a dedicated typing rule for typing *any* λ-abstraction without having to type the body of the abstraction: the norm typing rule.
- Second, note that each $\to_{\mathsf{sm}}$-step can be decomposed in a corresponding strong evaluation context and a $\mapsto_{\mathsf{m}}$-redex of the form $S\langle\lambda x.t\rangle u$ at its base. Since such a $\mapsto_{\mathsf{m}}$-redex is morally an application Λ_{L}-term, it can only be typed via an application of the app rule[4].

Summing up, the multiplicative index of a type derivation counts *both* the number of $\to_{\mathsf{sm}}$-steps and the number of application sub-terms of the $\to_{\mathsf{s}}$-normal form, and this information cannot be split in our type system.

Therefore, we shall not have null indices in (some) typings of $\to_{\mathsf{s}}$-normal forms, as we did in the weak and close cases. Instead, we shall have the quantitative information associated to the $\to_{\mathsf{s}}$-normal form[5]. That is, a notion of *size*, as given in the following:

Definition 51 (Strong CbV size of terms)**.**

[4]This is because the typing rules of the Strong CbV type system are syntactically-driven: only rule app is applicable to build an application Λ_{L}-term.

[5]Note that this is what happens also in the Open CbNeed and the Useful Open CbNeed cases.

The *Strong CbV size* $|t|_{\mathrm{S}}$ of a $t \in \Lambda_{\mathsf{L}}$ is the number of applications and λ-abstractions in t. Formally, $|t|_{\mathrm{S}}$ is given by

$$\begin{array}{rclcrcl} |x|_{\mathrm{S}} & := & 0 & & |tu|_{\mathrm{S}} & := & |t|_{\mathrm{S}} + |u|_{\mathrm{S}} + 1 \\ |\lambda x.t|_{\mathrm{S}} & := & |t|_{\mathrm{S}} + 1 & & |t[x \leftarrow u]|_{\mathrm{S}} & := & |t|_{\mathrm{S}} + |u|_{\mathrm{S}} \end{array}$$

11.2.2 The right amount of typing

Let us consider some examples that might clarify the role played by (co-)shrinkingness and unitary (co-)shrinkingness on the multiplicative and general size indices.

Let us begin by defining the *shrinkingness criterion*, crucial in proving that the general size index decreases by reduction—analogously, that it increases by expansion. A type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ is said to satisfy the shrinkingness criterion if Γ is co-shrinking and if t is an answer[6] then M is shrinking[7].

For example, consider $t := x(\lambda y.(\lambda z.z)y)$ and $u := x(\lambda y.z[z \leftarrow y])$, noting that $t \to_{\mathsf{sm}} u$. Now, let Φ be a type derivation for t of the following form

$$\dfrac{\dfrac{\dfrac{}{x : [\mathbf{0} \multimap [X]] \vdash^{(0,1)} x : \mathbf{0} \multimap [X]}\,\mathsf{ax}}{x : [\mathbf{0} \multimap [X]] \vdash^{(0,1)} x : [\mathbf{0} \multimap [X]]}\,\mathsf{many_{VAR}} \quad \dfrac{}{\emptyset \vdash^{(0,0)} \lambda y.(\lambda z.z)y : \mathbf{0}}\,\mathsf{many_{\lambda}}}{x : [\mathbf{0} \multimap [X]] \vdash^{(1,2)} x(\lambda y.(\lambda z.z)y) : [X]}\,\mathsf{app}$$

Since the Subject Reduction property is meant to provide a type derivation Ψ for u, with the exact same type context and right-hand side type, then it easy to show that the only possibility is that Ψ be of the following form:

$$\dfrac{\dfrac{\dfrac{}{x : [\mathbf{0} \multimap [X]] \vdash^{(0,1)} x : \mathbf{0} \multimap [X]}\,\mathsf{ax}}{x : [\mathbf{0} \multimap [X]] \vdash^{(0,1)} x : [\mathbf{0} \multimap [X]]}\,\mathsf{many_{VAR}} \quad \dfrac{}{\emptyset \vdash^{(0,0)} \lambda y.z[z \leftarrow y] : \mathbf{0}}\,\mathsf{many_{\lambda}}}{x : [\mathbf{0} \multimap [X]] \vdash^{(1,2)} x(\lambda y.z[z \leftarrow y]) : [X]}\,\mathsf{app}$$

As we can appreciate, the indices in Ψ have not decreased at all with respect to Φ. This has deep consequences in the overall proof technique, since the Correctness theorem is proven by induction on the general size index of type derivations. Therefore, the type context is assumed to be co-shrinking in the Subject Reduction property.

Unfortunately, this is not all that is needed in the Subject Reduction property. As expressed in the shrinkingness criterion, the right-hand side type must be shrinking as well. To see why, consider the diverging term Ω, which is not typable in the Strong CbV multi type system because normalizability equals typability in the Strong CbV multi type system. The only possibility for typing $\lambda y.\Omega$ is the following type derivation Θ:

$$\dfrac{}{\emptyset \vdash^{(0,0)} \lambda y.\Omega : \mathbf{0}}\,\mathsf{many_{\lambda}}$$

[6] That is, if it is a λ-abstraction up to ESs

[7] If Φ is shrinking then it satisfies the shrinkingness criterion, but the converse does not hold. The shrinkingness criterion is a relaxed form of shrinkingness, and is used to show that the general size index in the type derivation provided by the Subject Reduction property decreases. The latter, in turn, is used to prove the Correctness theorem by induction on the general size index of the type derivation.

But then the type derivation given by the Subject Reduction property can only be Θ itself, which implies that there is no decrement in the general size index. Once again, this would mean that we would not be able to prove the Correctness theorem by induction on the general size index.

Conversely, if we take $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} s : M$ and assume that M is shrinking—which entails that $\mathbf{0} \notin \mathsf{Occ}_{+}(M)$ and so $M \neq \mathbf{0}$—then s must be typed *at least once*. Therefore, type derivations must satisfy both conditions in the shrinkingness criterion for us to be able to prove the Correctness theorem by induction on the general size index.

Nevertheless, we are not merely interested in providing a Correctness theorem for the Strong CbV multi type system, but also in extracting quantitative information from type derivations. Unfortunately, the shrinkingness criterion does not suffice for such a purpose. Consider, for instance, type derivation Φ for t and Ψ for u: the multiplicative index does not decrease, even though it is supposed to measure multiplicative steps in Strong CbV-reduction sequences.

For this reason, we need to refine the shrinkingness criterion into the *unitary shrinkingness* one. We say that type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ satisfies the *unitary shrinkingness* criterion if Γ is unitary co-shrinking and if t is an answer then M is unitary shrinking[8].

To see how this affects the multiplicative index, let $\mathbf{X} \coloneqq [X] \multimap [X]$, and let Ω' be a type derivation for $\lambda y.(\lambda z.z)y$ of the following form:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{z : [X] \vdash^{(0,1)} z : X}\,\mathsf{ax}}{z : [X] \vdash^{(0,1)} z : [X]}\,\mathsf{many_{VAR}}}{\emptyset \vdash^{(1,2)} \lambda z.z : \mathbf{X}}\,\mathsf{fun} \quad \dfrac{\dfrac{}{y : [X] \vdash^{(0,1)} y : X}\,\mathsf{ax}}{y : [X] \vdash^{(0,1)} y : [X]}\,\mathsf{many}}{y : [X] \vdash^{(2,4)} (\lambda z.z)y : [X]}\,\mathsf{app}}{\emptyset \vdash^{(3,5)} \lambda y.(\lambda z.z)y : \mathbf{X}}\,\mathsf{fun} \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{z : [X] \vdash^{(0,1)} z : X}\,\mathsf{ax}}{z : [X] \vdash^{(0,1)} z : [X]}\,\mathsf{many_{VAR}}}{\emptyset \vdash^{(1,2)} \lambda z.z : \mathbf{X}}\,\mathsf{fun} \quad \dfrac{\dfrac{}{y : [X] \vdash^{(0,1)} y : X}\,\mathsf{ax}}{y : [X] \vdash^{(0,1)} y : [X]}\,\mathsf{many}}{y : [X] \vdash^{(2,4)} (\lambda z.z)y : [X]}\,\mathsf{app}}{\emptyset \vdash^{(3,5)} \lambda y.(\lambda z.z)y : \mathbf{X}}\,\mathsf{fun}}{\emptyset \vdash^{(6,10)} \lambda y.(\lambda z.z)y : [\mathbf{X}, \mathbf{X}]}\,\mathsf{many_{\lambda}}$$

Next, let Ω be a type derivation for t obtained by combining Ω' as follows:

$$\dfrac{\dfrac{\dfrac{}{x : [[\mathbf{X}, \mathbf{X}] \multimap [X]] \vdash^{(0,1)} x : [\mathbf{X}, \mathbf{X}] \multimap [X]}\,\mathsf{ax}}{x : [[\mathbf{X}, \mathbf{X}] \multimap [X]] \vdash^{(0,2)} x : [[\mathbf{X}, \mathbf{X}] \multimap [X]]}\,\mathsf{many_{VAR}} \quad \Omega'}{x : [[\mathbf{X}, \mathbf{X}] \multimap [X]] \vdash^{(7,13)} x(\lambda y.(\lambda z.z)y) : [X]}\,\mathsf{app}$$

Note that $\mathbf{X} \in \mathsf{Occ}_{-}([[\mathbf{X}, \mathbf{X}] \multimap [X]])$ and that $|\mathbf{X}| = 2$, and so the type context is *not unitary* co-shrinking. That is, because of the non-unitary co-shrinking type given to x, the multiplicative redex $(\lambda z.z)y$ must be typed twice in Ω', and then the multiplicative index coming from Ω' is not as expected: reducing the redex decreases the multiplicative index by 4 instead of by 2.

To see this, take Z' as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{z : [X] \vdash^{(0,1)} z : [X]}\,\mathsf{ax} \quad \dfrac{}{y : [X] \vdash^{(0,1)} y : [X]}\,\mathsf{ax}}{y : [X] \vdash^{(0,3)} z[z \leftarrow y] : [X]}\,\mathsf{ES}}{\emptyset \vdash^{(1,4)} \lambda y.z[z \leftarrow y] : \mathbf{X}}\,\mathsf{fun} \quad \dfrac{\dfrac{\dfrac{\dfrac{}{z : [X] \vdash^{(0,1)} z : [X]}\,\mathsf{ax} \quad \dfrac{}{y : [X] \vdash^{(0,1)} y : [X]}\,\mathsf{ax}}{y : [X] \vdash^{(0,3)} z[z \leftarrow y] : [X]}\,\mathsf{ES}}{\emptyset \vdash^{(1,4)} \lambda y.z[z \leftarrow y] : \mathbf{X}}\,\mathsf{fun}}{\emptyset \vdash^{(2,8)} \lambda y.z[z \leftarrow y] : [\mathbf{X}, \mathbf{X}]}\,\mathsf{many_{\lambda}}$$

[8]Once again, note the unitary shrinkingness of Φ implies that it satisfies the unitary shrinkingness criterion, but the converse does not hold.

We can then use Z' to derive $Z \rhd_{\mathrm{S}} x : [[\mathbf{X}, \mathbf{X}] \multimap [X]] \vdash^{(3,11)} x(\lambda y.z[z \leftarrow y]) : [X]$ as follows

$$\dfrac{\dfrac{\dfrac{}{x : [[\mathbf{X}, \mathbf{X}] \multimap [X]] \vdash^{(0,1)} x : [\mathbf{X}, \mathbf{X}] \multimap [X]}\ \mathsf{ax}}{x : [[\mathbf{X}, \mathbf{X}] \multimap [X]] \vdash^{(0,2)} x : [[\mathbf{X}, \mathbf{X}] \multimap [X]]}\ \mathsf{many}_{\mathsf{VAR}} \qquad Z'}{x : [[\mathbf{X}, \mathbf{X}] \multimap [X]] \vdash^{(3,11)} x(\lambda y.z[z \leftarrow y]) : [X]}\ \mathsf{app}$$

As explained above, the Correctness theorem for the Strong CbV system proves that the relation between the multiplicative index and the number of multiplicative steps in a normalizing sequence is *exactly* 2 in unitary shrinking type derivations. Thus, this excessive decrement stems from the fact that we *over*-type $\lambda y.(\lambda z.z)y$ in Ω'.

Therefore, we see that the type context must be unitary co-shrinking in order for the Subject Reduction property to provide a type derivation whose multiplicative index is decreased exactly by 2 by each multiplicative step.

Finally, to see why the right-hand side type must also be *unitary* shrinking, let us consider the following simple example. Let $\lambda x.t \in \Lambda_{\mathsf{L}}$ be such that $t \to_{\mathsf{sm}} u$. If $|N| = 1$, say $M = [L]$, then any type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} \lambda x.t : M$ must be of the following form:

$$\dfrac{\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} \lambda x.t : L}{\Gamma \vdash^{(m,s)} \lambda x.t : [L]}\ \mathsf{many}_{\lambda}$$

Note that there is exactly one premise to typing rule many_{λ}. If all the other conditions hold, we may then obtain a type derivation $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} \lambda x.u : L$ such that $m' = m - 2$ by application of the Subject Reduction property[9].

11.2.3 Strong CbV correctness

We can finally proceed to show the Strong CbV incarnation of the correctness sections that we saw for previous systems. Although it roughly follows the same general schema, there are certain subtleties to be considered given the open and strong setting of Strong CbV. Along the way, we may even encounter some strange statements that require short explanations to be properly understood, which the reader may find in the Technical Appendix. Nevertheless, we hope that the "Strong CbV measurements" and "The right amount of typing" sections have given the reader an intuitive idea of the interplay between the Strong CbV strategy, the Strong CbV type system and the class of (unitary) shrinking types.

As a starting point, let us show the Strong CbV type system enjoys relevance:

Lemma 11.2.1 (Relevance of the Strong CbV type system)**.**
Let $t \in \Lambda_{\mathsf{L}}$ and $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ be a type derivation. If $x \notin \mathsf{fv}(t)$ then $x \notin \mathsf{dom}(\Gamma)$.

(Click here to see the complete proof in the Technical Appendix)

The following is the base case in the proof of the Correctness theorem for Strong CbV:

Proposition 11.2.2 (Typing properties of Strong CbV-normal forms)**.**
Let f_s be a strong fireball and let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} f_s : M$ be a type derivation for it, with Γ a co-shrinking (resp. unitary co-shrinking) type context and such that if f_s is an answer then M is shrinking (resp. unitary shrinking). Then $|f_s|_{\mathrm{S}} \leq m$ (resp. $|f_s|_{\mathrm{S}} = m$).

[9]Moreover, the Subject Reduction property guarantees that $s' < s$, as explained above.

Proof. (Click here to see the complete proof in the Technical Appendix)
The proof is by structural induction on f_s.

□

Note that if Φ is unitary shrinking, then not only does Proposition 11.2.2 explicitly give the exact Strong CbV size of f_s, it also shows how to neatly separate this datum from the quantitative information on the $\to_{\mathsf{s}}$-normalization process. In other words, it is at this level that we can extract the Strong CbV size of f_s, and then be sure that the decrease by 1 in the multiplicative index in Proposition 11.2.5.1 (Shrinking Quantitative Subject Reduction for Strong CbV - $\to_{\mathsf{sm}}$-step) only corresponds to the $\to_{\mathsf{s}}$-normalization process.

Let us continue now with the usual Substitution lemma required for the proof of the Subject Reduction property:

Lemma 11.2.3 (Substitution for Strong CbV)**.**
Let v_{T} be a theoretical value and let

$$\Phi_t \rhd_{\mathrm{S}} \Gamma; x : N \vdash^{(m,s)} t : M$$
$$\Psi \rhd_{\mathrm{S}} \Pi \vdash^{(m',s')} v_{\mathrm{T}} : N$$

Then there exists type derivation $\Phi_{t\{x\leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} \Gamma \uplus \Pi \vdash^{(m'',s'')} t\{x\leftarrow v_{\mathrm{T}}\} : M$ such that $m'' = m + m'$ and $s'' \leq s + s'$.

Proof. (Click here to see the complete proof in the Technical Appendix)
The proof is by structural induction on t.

□

The main difference between this and the other substitutions lemmas in any of the previous systems is that Lemma 11.2.3 does not concern a *linear* substitution process, simply because in the VSC substitutions are realized via the meta-level substitution mechanism, which substitutes every occurrence of the variable at once. As a direct consequence of this, one may note the absence of any notion of splitting of the multi type assigned by the type context to the variable being substituted. This is because every occurrence of variable x in t—accounted for by the multi type N resulting from collecting every typed occurrence of x in Φ—must be simultaneously substituted by value v, morally being left with $x : \mathbf{0}$ after the substitution has taken place.

We are now able to prove the Subject Reduction property. The novelty in this system is that, since Strong CbV is defined as the contextual closure by strong evaluation contexts of the open strategy $\to_{\mathsf{w}}$, we first give a proof of Subject Reduction for $\to_{\mathsf{w}}$. We then build on it to give the desired proof of the Subject Reduction property for the whole $\to_{\mathsf{s}}$ strategy in Proposition 11.2.5 (Shrinking Quantitative Subject Reduction for Strong CbV) below.

Lemma 11.2.4 (Open Quantitative Subject Reduction for Strong CbV)**.**
Let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ be a type derivation.

1. Multiplicative*: If $t \to_{\mathsf{wm}} u$, then there exists a derivation $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m-2,s-1)} u : M$.*
2. Exponential*: If $t \to_{\mathsf{we}} u$, then there exists a derivation $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} u : M$ such that $m' = m$ and $s' < s$.*

Proof. (Click here to see the complete proof in the Technical Appendix)

The multiplicative part is proven by induction on the open evaluation context W such that $t = W\langle s\rangle \to_{\mathsf{wm}} W\langle s'\rangle = u$.

The exponential part is proven by induction on the open evaluation context W such that $t = W\langle s\rangle \to_{\mathsf{we}} W\langle s'\rangle = u$, using Lemma 11.2.3 (Substitution for Strong CbV) for the base case.

□

Proposition 11.2.5 (Shrinking Quantitative Subject Reduction for Strong CbV)**.**

Let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ be a type derivation, with Γ a co-shrinking (resp. unitary co-shrinking) type context. Moreover, suppose that if t is an answer then M is shrinking (resp. unitary shrinking).

1. Multiplicative*: If $t \to_{\mathsf{sm}} t'$, then $m \geq 2$, $s \geq 1$, and there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} t' : M$ such that $m' \leq m-2$ and $s' < s$ (resp. $m' = m-2$ and $s' = s-1$).*
2. Exponential*: If $t \to_{\mathsf{se}} t'$, then $s \geq 1$ and there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} t' : M$ such that $m' = m$ and $s' < s$.*

Proof. (Click here to see the complete proof in the Technical Appendix)

By induction on the strong evaluation context S such that $t = S\langle s\rangle \to_{\mathsf{s}} S\langle s'\rangle = u$, with $s \to_{\mathsf{wm}} s'$ or $s \to_{\mathsf{we}} s'$. The base case is given by Lemma 11.2.4 (Open Quantitative Subject Reduction for Strong CbV).

□

With all of this, we can now prove the desired Correctness result.

Theorem 11.2.6 (Shrinking Correctness for Strong CbV)**.**

Let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ be a shrinking (resp. unitary shrinking) type derivation. Then there exists $u \in \Lambda_{\mathsf{L}}$ such that

1. *u is in $\to_{\mathsf{s}}$-normal form,*
2. *there exists a reduction sequence $d : t \to_{\mathsf{s}}^{*} u$, and*
3. *$m \geq 2|d|_{\mathsf{m}} + |u|_{\mathrm{S}}$ (resp. $m = 2|d|_{\mathsf{m}} + |u|_{\mathrm{S}}$).*

Proof. (Click here to see the complete proof in the Technical Appendix)

By induction on the general size index of Φ, s. Case analysis on whether $t \to_{\mathsf{s}}$-reduces or not:

- If t is in $\to_{\mathsf{s}}$-normal form, then Proposition 10.3.2 (Fullness of Strong CbV) gives that t is a strong super fireball. The statement follows, both if Φ is unitary shrinking as well as if it is non-unitary shrinking, by Proposition 11.2.2 (Typing properties of Strong CbV-normal forms).
- Let $t \to_{\mathsf{s}} u$. The proof is split between multiplicative and exponential steps. Let us consider the multiplicative one here; *i.e.*, let $t \to_{\mathsf{sm}} u$. By Proposition 11.2.5.1 (Shrinking Quantitative Subject Reduction for Strong CbV), there exists type derivation $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} u : M$ such that if Φ is non-unitary (resp. unitary), then $m' \leq m-2$ (resp. $m' = m-2$) and $s' < s$ (resp. $s' = s-1$). The statement finally follows by application of the *i.h.* on Ψ.

□

Note that Theorem 11.2.6 implicitly states that unitary shrinking type derivations have *minimal* indices among all shrinking type derivations.

11.2.4 Strong CbV completeness

Let $t, u \in \Lambda_{\mathsf{L}}$ be such that $d : t \to_{\mathsf{s}}^{*} u$ is a normalizing sequence. The idea of proving a quantitative completeness theorem for Strong CbV is materialized into proving that every $\to_{\mathsf{s}}$-normalizing Λ_{L}-term t is typable with a type derivation whose multiplicative index is the Strong CbV size of u *plus* twice the number of $\to_{\mathsf{sm}}$-steps from t to u.

Again, this is proved following the standard schema applied to the previous systems: a Quantitative Subject Expansion property stating that typability can be pulled back along $\to_{\mathsf{s}}$-steps—increasing the general size index (by an arbitrary number) while also increasing the multiplicative index by exactly 2 units when it is a $\to_{\mathsf{sm}}$-step—and a lemma stating that every $\to_{\mathsf{s}}$-normal form has a type derivation whose multiplicative index is equal to the Strong CbV size of such normal form:

Proposition 11.2.7 (Shrinking typability of Strong CbV-normal forms)**.**

1. Inert*: For every strong inert term i_s and co-shrinking (resp. unitary co-shrinking) multi type M, there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} i_s : M$ for some $s \geq 1$, such that Γ is co-shrinking (resp. unitary co-shrinking) and $m \geq |i_s|_{\mathrm{S}}$ (resp. $m = |i_s|_{\mathrm{S}}$).*
2. Fireball*: For every strong fireball f_s, there exists a unitary shrinking type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} f_s : M$ such that $m = |f_s|_{\mathrm{S}}$ and $s \geq 1$.*

Proof. (Click here to see the complete proof in the Technical Appendix)

Both points are proved by mutual structural induction on the definition of strong inert terms and strong fireballs. Note that Proposition 11.2.7.1—only valid for strong inert terms and not strong fireballs in general—is required to make the induction go through in Proposition 11.2.7.2—which is the general case, covering both kinds of Strong CbV-normal forms.

□

We would like to stress here that this presentation of Proposition 11.2.7—where a kind of subproperty is first given for inert expressions before the more general case is covered—is analogous to the separate treatment of the so-called "normal terms" as appearing in Lemma 7.1.8 (Tight typability of normal terms)—see Sect. 7.1 (Multi type system for Open CbNeed).

As usual, Subject Expansion requires a Removal lemma at its base exponential case:

Lemma 11.2.8 (Removal for Strong CbV)**.**

Let $t \in \Lambda_{\mathsf{L}}$, let v_{T} be a theoretical value such that $x \notin \mathsf{fv}(v_{\mathrm{T}})$, and let

$$\Phi_{t\{x \leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t\{x \leftarrow v\} : M$$

Then there exist type derivations

$$\Phi_t \rhd_{\mathrm{S}} \Pi; x : N \vdash^{(m',s')} t : M$$
$$\Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m'',s'')} v : N$$

such that $\Gamma = \Pi \uplus \Delta$, $m = m' + m''$ and $s \leq s' + s''$.

Proof. (Click here to see the complete proof in the Technical Appendix)

By structural induction on t.

□

As what happened in Lemma 11.2.3 (Substitution for Strong CbV), the fact that substitutions in the VSC are realized via the meta-level substitution mechanism impacts the shape that the Removal lemma for Strong CbV takes.

Following an argument symmetrical to the one for Correctness, the fact that Strong CbV is defined as the contextual closure by strong evaluation contexts of the open strategy $\to_{\mathsf{w}}$ allows us to prove Subject Expansion by first covering the (simpler) open case.

Lemma 11.2.9 (Open Quantitative Subject Expansion for Strong CbV)**.**
Let $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} u : M$ be a type derivation.

1. Multiplicative*: If $t \to_{\mathsf{wm}} u$, then there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m+2,s')} t : M$ such that $s' > s$.*
2. Exponential*: If $t \to_{\mathsf{we}} u$, then there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s')} t : M$ such that $s' > s$.*

Proof. (Click here to see the complete proof in the Technical Appendix)
The multiplicative part is proven by induction on the open evaluation context W such that $t = W\langle s\rangle \to_{\mathsf{wm}} W\langle s'\rangle = u$.
The exponential part is proven by induction on the open evaluation context W such that $t = W\langle s\rangle \to_{\mathsf{we}} W\langle s'\rangle = u$, using Lemma 11.2.3 (Substitution for Strong CbV) for the base case.
□

Proposition 11.2.10 (Shrinking Quantitative Subject Expansion for Strong CbV)**.**
Let $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} t' : M$ be a type derivation, with Γ a unitary co-shrinking type context. Moreover, suppose that if u is an answer, then M is unitary shrinking.

1. Multiplicative*: If $t \to_{\mathsf{sm}} t'$, then there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m+2,s+1)} t : M$.*
2. Exponential*: If $t \to_{\mathsf{se}} t'$, then there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} t : M$ such that $m' = m$ and $s' > s$.*

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the strong evaluation context S such that $t = S\langle s\rangle \to_{\mathsf{s}} S\langle s'\rangle = u$, with $s \to_{\mathsf{wm}} s'$ or $s \to_{\mathsf{we}} s'$. The base case is given by Lemma 11.2.9 (Open Quantitative Subject Expansion for Strong CbV).
□

Theorem 11.2.11 (Shrinking Completeness for Strong CbV)**.**
Let $t \in \Lambda_{\mathsf{L}}$. If there exists $d : t \to_{\mathsf{s}}^{*} u$ such that u in $\to_{\mathsf{s}}$-normal form, then there exists a unitary shrinking type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(2|d|_{\mathsf{m}}+|u|_{\mathrm{S}},s)} t : M$, for some $s \geq 0$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the length $|d|$ of the reduction sequence $d : t \to_{\mathsf{s}}^{*} u$:

- If $|d| = 0$, then t is a strong super fireball by Proposition 10.3.2 (Fullness of Strong CbV). The statement follows by Proposition 11.2.7 (Shrinking typability of Strong CbV-normal forms).
- If $|d| > 0$, then d is of the following form

$$d \;:\; t \to_{\mathsf{s}} \underbrace{s \to_{\mathsf{s}}^{k-1} u}_{d'}$$

We can apply the *i.h.* with respect to d', getting *unitary* shrinking type derivation $\Theta \rhd_{\mathrm{S}} \Gamma \vdash^{(2|d'|+|u|_{\mathrm{S}},s')} s : M$. The statement follows by application of Proposition 11.2.10 (Shrinking Quantitative Subject Expansion for Strong CbV) on Ψ, yielding a an appropriate type derivation for t whose exact indices depend on whether $t \to_{\mathsf{sm}} s$ or $t \to_{\mathsf{se}} s$.
□

11.2.5 Size of Strong CbV-normal forms via multi types

In every previous system, we extracted quantitative information on the normalization process or on the structure of normal forms in a rather simple way: each index in these systems only gets incremented upon applications of particular typing rules—besides adding up the indices of the premises, that is. For instance, the multiplicative index in type derivations of the CbN system gets incremented only in applications of the app rule, and so this index represents exactly the number of app rules in the type derivation—see Sect. 5.2 (Multi type system for CbN).

We now give an alternative way of extracting the quantitative information regarding the

Strong CbV size of $\to_{\mathsf{s}}$-normal forms from a type derivation, namely from the type context and the right type of the final type judgement.

To do so, we first need to define the size of multi types and type contexts, which can be seen simply as counting the number of occurrences of $\multimap$. Formally, the *size* of linear and multi types is defined mutually inductively as follows

$$|X| = 0 \qquad |M \multimap N| = 1 + |M| + |N| \qquad |[L_1, \dots, L_n]| = \textstyle\sum_{i=1}^{n} |L_i|$$

Clearly, $|L| \geq 0$ and $|M| = 0$ if and only if $M = [X, \overset{n \in \mathbb{N}}{\dots}, X]$.

Given a type context $\Gamma \coloneqq x_1 : M_1; \dots; x_n : M_n$, we denote the list of its types by $\hat{\Gamma} \coloneqq (M_1; \dots; M_n)$. Note that, since any list of multi types $(M_1, \dots, M_n)$ can be seen as extracted from a type context $\Gamma \coloneqq \{x'_1 : M_1; \dots; x'_n : M_n\}$ for some list of variables $(x'_1, \dots, x'_n)$, we use the notation $\hat{\Gamma}$ for lists of multi types.

The *size* of a list of multi types $\hat{\Gamma} \coloneqq (M_1, \dots, M_n)$ is given by

$$|\hat{\Gamma}| \coloneqq \sum_{i=1}^{n} |M_i|$$

Clearly, $\mathsf{dom}(\Gamma) = \emptyset$ implies $|\hat{\Gamma}| = 0$.

Proposition 11.2.12 (Shrinking types bound the size of Strong CbV-normal forms)**.**

Let $t \in \Lambda_{\mathsf{L}}$ be in $\to_{\mathsf{s}}$-normal form and let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ be a type derivation, with Γ a co-shrinking type context.

1. Inert*: If t is a strong inert term, then $|M| + |t|_{\mathrm{S}} \leq |\hat{\Gamma}|$.*
2. Fireball*: If t is a strong fireball and M is shrinking, then $|t|_{\mathrm{S}} \leq |M| + |\hat{\Gamma}|$*

Proof. (Click here to see the complete proof in the Technical Appendix)

Both points are proved by mutual structural induction on the definition of strong inert terms and strong fireballs.

□

Proposition 11.2.13 (Strong CbV-normal forms have a minimal unitary shrinking type)**.**

Let $t \in \Lambda_{\mathsf{L}}$ be in $\to_{\mathsf{s}}$-normal form.

1. Inert*: If t is a strong inert term, then for every co-shrinking (resp. unitary co-shrinking) multi type M there exists a type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ such that Γ is a co-shrinking (resp. unitary co-shrinking) type context and $|M| + |t|_{\mathrm{S}} = |\hat{\Gamma}|$.*
2. Fireball*: If t is a strong fireball, then there exists a unitary shrinking derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ such that $m = |t|_{\mathrm{S}} = |M| + |\hat{\Gamma}|$. Moreover,*

$$m \;=\; \min\{m' \mid \exists\, \Psi \rhd_{\mathrm{S}} \Pi \vdash^{(m',s')} t : N,\ \textit{with } \Pi \textit{ co-shrinking and if } t \textit{ is an answer then } N \textit{ is shrinking}\}$$

Proof. (Click here to see the complete proof in the Technical Appendix)
Both points are proven by mutual induction on the definition of strong inert terms and of strong fireballs. Additionally, the minimality part of point (2) is a direct corollary of Proposition 11.2.2 (Typing properties of Strong CbV-normal forms).

□

11.2.6 Types and structural equivalence in the VSC

As closure for the Correctness and Completeness parts for the Strong CbV system, let us prove that typability in the theory of the Value Substitution Calculus is considered up to the structural equivalence $\equiv$ between Λ_{L}-terms:

Proposition 11.2.14 (Structural equivalence preserves typability and indices)**.**
Let $t, u \in \Lambda_{\mathsf{L}}$ be such that $t \equiv t'$.
Then there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ if and only if there exists type derivation $\Phi' \rhd \Gamma \vdash^{(m,s)} t' : M$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By structural induction on the LSC evaluation context D such that $t = D\langle u\rangle \equiv D\langle u'\rangle = t'$, with $u \equiv_a u'$ and $a \in \{\equiv_{\mathrm{com}}, \equiv_{@\mathrm{r}}, \equiv_{@\mathrm{l}}, \equiv_{[\cdot]}\}$.

□

Note that both the multiplicative and the size indices are preserved via structural equivalence as shown in Proposition 11.2.14. This is a crucial point regarding the (minimal) bounds provided by type derivations for Λ_{L}-terms modulo structural equivalence.

11.3 A semantical proof of VSC-normalization via Strong CbV

Here we extend the Subject Reduction and Subject Expansion properties of multi types to encompass the whole VSC, not just the Strong CbV strategy $\to_{\mathsf{s}}$. There are two reasons for this:

1. The first is that, in this way, we smoothly obtain a normalization theorem for $\to_{\mathsf{s}}$, by exploiting an elegant proof technique used in de Carvalho, Pagani and Tortora de Falco's [CPT11] and Mazza, Pellisier and Vial's [MPV18]. That is, we prove that if a fixed Λ_{L}-term is $\to_{\mathsf{vsc}}$-normalizable, then it is $\to_{\mathsf{s}}$-normalizable—see Theorem 11.3.3 ($\to_{\mathsf{s}}$ is normalizing) below. As a direct corollary, $\to_{\mathsf{vsc}}$-normalizability and $\to_{\mathsf{s}}$-normalizability are equivalent.
2. The second is to show that the Strong CbV type system provides an adequate semantics for the whole of the VSC, not just the strategy.
   We shall see that the extended properties are not of a quantitative nature, but rather of a *qualitative* one. This is because VSC-steps other than those of $\to_{\mathsf{s}}$ do not in general decrease the size index of type derivations.

The following result is only used for proving the invariance of the semantics presented below, but plays no role in proving Theorem 11.3.3 ($\to_{\mathsf{s}}$ is normalizing).

Proposition 11.3.1 (Qualitative Subject Reduction for VSC)**.**
Let $t, u \in \Lambda_{\mathsf{L}}$ be such that $t \to_{\mathsf{vsc}} u$, and let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ be a type derivation. Then there exists type derivation $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} u : M$ such that $m' \leq m$ and $s' \leq s$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the LSC evaluation context D such that $t = D\langle s\rangle \to_{\mathsf{vsc}} D\langle s'\rangle = u$, with $s \mapsto_{\mathsf{m}} s'$ or $s \mapsto_{\mathsf{e}} s'$.

□

Unlike its 'Reduction' counterpart, the following is crucial in proving Theorem 11.3.3 ($\to_{\mathsf{s}}$ is normalizing). Essentially, it is used to "pull back" the type derivation given by Proposition 11.2.7 (Shrinking typability of Strong CbV-normal forms).

Proposition 11.3.2 (Qualitative Subject Expansion for VSC)**.**
Let $t, u \in \Lambda_{\mathsf{L}}$ be such that $t \to_{\mathsf{vsc}} u$, and let $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} u : M$ be a type derivation. Then there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ such that $m' \leq m$ and $s' \leq s$.

Proof. (Click here to see the complete proof in the Technical Appendix)
By induction on the LSC evaluation context D such that $t = D\langle s\rangle \to_{\mathsf{vsc}} D\langle s'\rangle = u$, with $s \mapsto_{\mathsf{m}} s'$ or $s \mapsto_{\mathsf{e}} s'$.

□

Finally,

Theorem 11.3.3 ($\to_{\mathsf{s}}$ is normalizing)**.**
Let $t \in \Lambda_{\mathsf{L}}$. If there exists a reduction sequence $d : t \to_{\mathsf{vsc}}^{*} u$ for some u in $\to_{\mathsf{vsc}}$-normal form, then there exists $d' : t \to_{\mathsf{s}}^{*} u$.

Proof. (Click here to see the complete proof in the Technical Appendix)
The proof starts by taking the unitary shrinking type derivation Φ_u for u given by Proposition 11.2.7 (Shrinking typability of Strong CbV-normal forms), qualitatively expanding from Φ_u to a unitary shrinking type derivation Φ_t for t—via Proposition 11.3.2 (Qualitative Subject Expansion for VSC)—applying Theorem 11.2.6 (Shrinking Correctness for Strong CbV) to get a normalizing reduction sequence $d' : t \to_{\mathsf{s}}^{*} s$. Note that s is a super strong fireball—by Proposition 10.3.2 (Fullness of Strong CbV)—and so s is also in $\to_{\mathsf{vsc}}$-normal form—by Proposition 10.2.3 (Syntactic characterization of VSC-normal forms). Theorem 10.2.2 (Confluence of $\to_{\mathsf{vsc}}$) finally gives that $s = u$.

□

11.3.1 VSC semantics

Firstly, let us suppose we defined the semantics (of $t \in \Lambda_{\mathsf{L}}$ for $\vec{x}$ given by the Strong CbV system) as follows

$$[\![t]\!]_{\vec{x}}^{\text{Strong CbV}} \coloneqq \{((M_1, \dots, M_n), M) \mid \exists\, \Phi \rhd_{\mathrm{S}} x_1 : M_n; \dots; x_n : M_n \vdash^{(m,s)} t : M\} \tag{11.1}$$

This semantic is invariant by $\to_{\mathsf{vsc}}$-reduction, by Proposition 11.3.1 (Qualitative Subject Reduction for VSC) and Proposition 11.3.2 (Qualitative Subject Expansion for VSC). However, it is not adequate, which can be appreciated by easily adapting the counterexample given for Open CbNeed in Subsect. 7.1.3 (Open CbNeed semantics, page 97). Indeed, note that while $x\,(\lambda y.\Omega)$ is not $\to_{\mathsf{vsc}}$-normalizing, it is typable in the Strong CbV system:

$$\dfrac{\dfrac{}{x : [\mathbf{0} \multimap M] \vdash^{(0,1)} x : [\mathbf{0} \multimap M]}\ \mathsf{ax} \qquad \dfrac{}{\emptyset \vdash^{(0,0)} \lambda y.\Omega : \mathbf{0}}\ \mathsf{many}_{\lambda}}{x : [\mathbf{0} \multimap M] \vdash^{(1,2)} x\,(\lambda y.\Omega) : M}\ \mathsf{app}$$

Therefore, let us adapt the semantics to our results. We define the *semantics of t for $\vec{x}$* given by the Strong CbV system as

$$[\![t]\!]_{\vec{x}}^{\text{Strong CbV}} := \{((M_1,\dots,M_n),M) \mid \exists\, \Phi \rhd_{\mathrm{S}} x_1 : M_n;\dots;x_n : M_n \vdash^{(m,s)} t : M \text{ such that } \Phi \text{ is shrinking}\}$$

Being a restriction to the semantics defined in 11.1, this new semantics is also invariant with respect to $\to_{\mathsf{vsc}}$—and so invariant with respect to $\to_{\mathsf{s}} \subseteq \to_{\mathsf{vsc}}$. But most importantly, it is adequate with respect to $\to_{\mathsf{s}}$, as proven by Theorem 11.2.6 (Shrinking Correctness for Strong CbV) and Theorem 11.2.11 (Shrinking Completeness for Strong CbV).

Remarkably, the adequacy of the semantics with respect to $\to_{\mathsf{s}}$-normalization can be extended to $\to_{\mathsf{vsc}}$-weak normalization as follows:

- *Typability implies weak normalization.* Let t be typable in the Strong CbV system. By Theorem 11.2.6 (Shrinking Correctness for Strong CbV), t is $\to_{\mathsf{s}}$-normalizing. Since $\to_{\mathsf{s}}$ is a sub-relation of $\to_{\mathsf{vsc}}$, we have that t is $\to_{\mathsf{vsc}}$-weakly normalizing.
- *Weak normalization implies typability.* Let t be $\to_{\mathsf{vsc}}$-weakly normalizing. By Theorem 11.3.3, t is $\to_{\mathsf{s}}$-normalizing. Hence, t is typable in the Strong CbV system, given by Theorem 11.2.11 (Shrinking Completeness for Strong CbV).

Finally, note that since shrinking type derivations may contain arrow types, then the semantics also satisfies compositionality. This is a remarkable feature of the Strong CbV semantics, since the Open CbNeed and Useful Open CbNeed ones only satisfy invariance and adequacy, while lacking compositionality.

Chapter 12

Conclusion

As pioneered by Accattoli and Dal Lago in their "(Leftmost-outermost) Beta Reduction is Invariant, Indeed" [AL16], deriving a *reasonable* time cost model for the λ-calculus—that is, one proven to be polynomially related to the time cost model of Turing machines—requires implementing a series of key adaptations to the substitution process. Among them, the so-called "useful optimizations" are absolutely necessary in the case of strong reduction, where reduction goes under λs.

This work aimed at understanding how the reasonability arguments analyzed in [AL16] for the leftmost-outermost evaluation strategy may be adapted to the call-by-need setting, in particular by implementing the useful optimizations. We began by the most restricted/less general case and extended it following a principled and incremental approach, finally arriving at the Useful Open CbNeed evaluation strategy.

For all the evaluation strategies thus produced, we provided a multi type system characterizing normalization of the evaluation strategy. Additionally, these type systems are finely tuned to produce quantitative information about the normalization process in the form of upper bounds. In other words, we gave a multi type-theoretical presentation of qualitative and quantitative features of the normalization process of each of the evaluation strategies.

Our starting point was the adaptation in "Distilling Abstract Machines" [ABM14] of the weak, closed and head version of call-by-need to the "Linear Substitution Calculus"—or LSC for short—calling it the *CbNeed* evaluation strategy. We then study the operational differences and similarities between CbNeed and its call-by-name and call-by-value variants, which we here call CbN and CbV. We conclude that CbNeed may be seen as wisely combining erasure and duplication mechanisms from CbN and CbV.

These operational traits, namely duplication and erasure, can also be found in the multi type systems given for CbN and CbV in the literature. We combined them to derive the first multi type system for CbNeed which not only characterizes its normalization and provides upper bounds, but also *gives precise quantitative information* about it. We moreover prove an efficiency result relating CbNeed and CbV at the level of the multi type systems.

Next, we extend the CbNeed evaluation strategy to our novel *Open CbNeed*. This evaluation strategy should be seen as a well-balanced intermediate step between CbNeed and the (most general) strong version of call-by-need—*e.g.*, the one introduced in "Foundations of Strong Call by Need" [Bal+17].

Open CbNeed is in particular necessary in the reasonability study of strong call-by-need, as the useful optimizations are more naturally implemented in this setting and then extended to the strong setting. Following this reason, we take Open CbNeed and derive our Useful Open CbNeed

evaluation strategy. The useful optimizations rely heavily on the notions of applied and unapplied variable occurrences, so both Open CbNeed and Useful Open CbNeed are in fact formalized in the split LSC—a variant of the LSC where applicativity has a more *local* taste.

Regarding the multi type systems for these evaluation strategies, we first take the CbNeed one and extend it to the Open CbNeed multi type system by refining the ground types—which we call "tight types"—and adding the necessary typing rules to deal with them. The axioms in the Open CbNeed system are then slightly refined to produce a multi type system characterizing normalization of Useful Open CbNeed. The quantitative relations between these two type systems and their corresponding evaluation strategies are essentially the same as the ones in the CbNeed case.

By lack of time, we were unable to achieve the final goal of producing a reasonable and strong version of call-by-need, and a multi type system characterizing its normalization and providing quantitative information. Nonetheless, we did produce a further evaluation strategy, the so-called *Strong CbV*, which is a normalizing strategy of the "Value Substitution Calculus"—or VSC for short—introduced in [AP12]. Finally, we derived a multi type system for Strong CbV, which not only considerably contributes to the general quantitative understanding of strong reduction via multi type systems, but also allows us to produce interesting normalization results for the VSC.

Future work. During the preparation of this thesis, our main guiding principle was to attain a consistent and resource-aware formulation of a reasonable and strong version of call-by-need. Ideally, this would have been comprised of operational-semantical, (multi) type-theoretical and implementative analyses. We are still far behind this goal, even if the technical development of the results in this work are significantly lengthy.

Consequently, there are several lines of work that might be interesting to pursue, namely:

- The design of a reasonable and strong call-by-need strategy, taking Useful Open CbNeed as basis. Roughly put, this would mostly consist in iterating Useful Open CbNeed under λs.
- The derivation of a multi type system characterizing the reasonable and strong call-by-need strategy. This is probably attainable by extending the Useful Open CbNeed multi type system to produce one characterizing the reasonable and strong call-by-need.
- As explained in Sect. 1.4 on page 10, we have (partial) results on abstract machines implementing Open CbNeed and Useful Open CbNeed. Then, a next step could consist in finishing implementing them, proving that they are bilinearly related to their corresponding evaluation strategy, and using them to derive an abstract machine for the reasonable and strong call-by-need, satisfying the same properties of bilinearity.
- Finally, recall that CbNeed may be expressed as a combination of the (*wise*) erasure mechanism in CbN and the (*wise*) duplication mechanism in CbV. We believe that by making the inverse choices—namely, by picking the erasure mechanism in CbV and the duplication mechanism in CbN—one ends up with a maximal evaluation strategy. We could then derive a multi type system providing precise bounds for the maximal strategy by making the inverse choices of the ones made for the derivation of the CbNeed system from the CbN and CbV ones. Besides closing the diagram between the four strategies, this complementary result would also extend the understanding of erasure and duplication at the types level, which would be helpful in the derivation of a multi type system for the reasonable and strong call-by-need.

Chapter 13

Technical appendix

13.1 Proofs of Chapter 4 (CbN, CbV and CbNeed)

Let us begin by giving some basic properties that shall be used repeatedly in the upcoming proofs for this chapter:

Lemma 13.1.1 (Syntactic properties of CbN, CbV and CbNeed)**.**
Let $t \in \Lambda_{\mathsf{L}}$.

1. Shape of CbN and CbNeed-normal forms*:* $\mathsf{norm}(t)$ *if and only if* t *is an* answer*;* i.e.*, if and only if there exist substitution context* S *and* $v \in \mathsf{Val}$ *such that* $t = S\langle v\rangle$*.*
2. Shape of CbV-normal forms*:* $\mathsf{norm}_{\mathrm{CbV}}(t)$ *if and only if* $t = v[x_1 \leftarrow u_1]...[x_n \leftarrow u_n]$ *for some* $n \geq 0$ *and such that* $\mathsf{norm}_{\mathrm{CbV}}(t_i)$ *for every* $1 \leq i \leq n$*.*
3. Focusing through CbN evaluation contexts*: There exists exactly one way to rewrite* $t = C\langle u\rangle$*, with* C *a CbN evaluation context and* u *either in* Var *or in* Val*. Similarly, there exists exactly one way to rewrite* $t = E\langle u\rangle$*, with* E *a CbNeed evaluation context and* u *either in* Var *or in* Val*.*
4. *Every CbN evaluation context is also a CbV evaluation context and a CbNeed evaluation context.*

Proof.

1. *Shape of CbN and CbNeed-normal forms*: Trivial by a simple induction on predicate $\mathsf{norm}(.)$
2. *Shape of CbV-normal forms*: Trivial by a simple induciton on predicate $\mathsf{norm}_{\mathrm{CbV}}(.)$.
3. *Focusing through CbN evaluation contexts*: Follows easily by induction on CbN and CbNeed evaluation contexts.
4. It is clear that every grammar production for CbN evaluation contexts is contained among the ones for CbV and CbNeed evaluation contexts.

□

Proposition 13.1.2 (Diamond property for CbV)**.**
$\to_{\mathrm{CbV}}$ *is diamond.*

Proof. (Click here to go back to main chapter.)
Let $t \to_{\mathrm{CbV}} u$ and $t \to_{\mathrm{CbV}} s$. We proceed by induction on the shape of t:

- *Variable*: Impossible, since variables are in $\to_{\mathrm{CbV}}$-normal form.
- *Abstraction*: Impossible too, given that $\to_{\mathrm{CbV}}$ is defined in terms of *weak* LSC evaluation contexts, and so reduction does not enter the bodies of λ-abstractions.

- *Application*: Let $t \coloneqq t_1 t_2$. We proceed by case analysis on the subterm reduced by $t \to_{\mathrm{CbV}} u$ and $t \to_{\mathrm{CbV}} s$:
  - *Left-left*: The statement follows by application of the *i.h.* on t_1 in the case where
    $$u = t'_1 t_2 \ {}_{\mathrm{CbV}}\!\leftarrow t_1 t_2 \to_{\mathrm{CbV}} t''_1 t_2 = s$$
  - *Right-right*: The statement follows by application of the *i.h.* on t_2 in the case where
    $$u = t_1 t'_2 \ {}_{\mathrm{CbV}}\!\leftarrow t_1 t_2 \to_{\mathrm{CbV}} t_1 t''_2 = s$$
  - *Left-right*: Let
    $$u = V_1\langle m'_1\rangle t_2 \ {}_{\mathrm{CbV}}\!\leftarrow V_1\langle m_1\rangle t_2 = t_1 t_2 = t_1 V_2\langle m_2\rangle \to_{\mathrm{CbV}} t_1 V_2\langle m'_2\rangle = s$$
    Then, we can close the diagram by taking
    $$u = V_1\langle m'_1\rangle V_2\langle m_2\rangle \to_{\mathrm{CbV}} V_1\langle m'_1\rangle V_2\langle m'_2\rangle \ {}_{\mathrm{CbV}}\!\leftarrow V_1\langle m_1\rangle V_2\langle m'_2\rangle = s$$
    where $m = V_1\langle m'_1\rangle V_2\langle m'_2\rangle$.
  - *Right-left*: The case where $t \to_{\mathrm{CbV}} u$ reduces t_2 and $t \to_{\mathrm{CbV}} s$ reduces t_1 is proven analogous to the previous case.
- *Appended ES*: Let $t = t_1[x \leftarrow t_2]$. We proceed by case analysis on the subterm reduced by $t \to_{\mathrm{CbV}} u$ and $t \to_{\mathrm{CbV}} s$:
  - *Left-left*: The statement follows by application of the *i.h.* on t_1 in the case where
    $$u = t'_1[x \leftarrow t_2] \ {}_{\mathrm{CbV}}\!\leftarrow t_1[x \leftarrow t_2] \to_{\mathrm{CbV}} t''_1[x \leftarrow t_2] = s$$
  - *Right-right*: The statement follows by application of the *i.h.* on t_2 in the case where
    $$u = t_1[x \leftarrow t'_2] \ {}_{\mathrm{CbV}}\!\leftarrow t_1[x \leftarrow t_2] \to_{\mathrm{CbV}} t_1[x \leftarrow t''_2] = s$$
  - *Left-right*: Let
    $$u = V_1\langle m'_1\rangle[x \leftarrow t_2] \ {}_{\mathrm{CbV}}\!\leftarrow V_1\langle m_1\rangle[x \leftarrow t_2] = t_1[x \leftarrow t_2] = t_1[x \leftarrow V_2\langle m_2\rangle] \to_{\mathrm{CbV}} t_1[x \leftarrow V_2\langle m'_2\rangle] = s$$
    Then, we can close the diagram by taking
    $$u = V_1\langle m'_1\rangle[x \leftarrow V_2\langle m_2\rangle] \to_{\mathrm{CbV}} V_1\langle m'_1\rangle[x \leftarrow V_2\langle m'_2\rangle] \ {}_{\mathrm{CbV}}\!\leftarrow V_1\langle m_1\rangle[x \leftarrow V_2\langle m'_2\rangle] = s$$
    where $m = V_1\langle m'_1\rangle[x \leftarrow V_2\langle m'_2\rangle]$.
  - *Right-left*: The case where $t \to_{\mathrm{CbV}} u$ reduces t_2 and $t \to_{\mathrm{CbV}} s$ reduces t_1 is proven analogous to the previous case.

□

(Click here to go back to main chapter.)

Proposition 13.1.3 (Syntactic characterization of closed normal forms)**.**
Let t be a closed term.

1. CbN and CbNeed*: For $\mathsf{r} \in \{\mathrm{CbN}, \mathrm{CbNeed}\}$, t is r-normal if and only if $\mathsf{norm}(t)$.*
2. CbV*: t is* CbV*-normal if and only if $\mathsf{norm}_{\mathrm{CbV}}(t)$.*

Proof. *(Click here to go back to main chapter.)*

1. First, we prove that $t \in \Lambda_{\mathsf{L}}$ is in $\to_{\mathrm{CbN}}$-normal form if and only if $\mathsf{norm}(t)$.
   ⇒: Let t be a closed $\to_{\mathrm{CbN}}$-normal form. We prove the statement by structural induction on t, proceeding by case analysis on its shape:
   - *Variable*: Impossible, because t is closed by hypothesis.
   - *Abstraction*: If $t = \lambda y.u$, then we can derive that $\mathsf{norm}(t)$.
   - *Application*: Suppose $t = us$. Then u would be closed and in $\to_{\mathrm{CbN}}$-normal form—or else t would not be so—and hence application of the *i.h.* would give that $\mathsf{norm}(u)$ by *i.h.*. However, Lemma 13.1.1.1 (Shape of CbN and CbNeed-normal forms) would then give that $u = S\langle\lambda y.m\rangle$, and so $t = S\langle\lambda y.m\rangle s$. Absurd, because t is in $\mathsf{m}_{\mathrm{CbN}}$-normal form.
   - *Explicit substitution*: Let $t = u[x{\leftarrow}s]$. Note that since t is in $\to_{\mathrm{CbN}}$-normal form, then so must be u. There are four possible cases:
     - If $u = S\langle v\rangle$, then application of Lemma 13.1.1.1 (Shape of CbN and CbNeed-normal forms) gives that $\mathsf{norm}(u)$. Hence, $\mathsf{norm}(u[x{\leftarrow}s])$.
     - If u is closed, then we can apply the *i.h.* on it and get that $\mathsf{norm}(u)$. Hence, $\mathsf{norm}(u[x{\leftarrow}s])$.
     - Suppose $u \coloneqq C\langle\!\langle y\rangle\!\rangle$. But then it must be—since t is closed—that $t = D\langle\cdot\rangle C\langle\!\langle y\rangle\!\rangle[y{\leftarrow}m]$, for some CbN context D and term m, which would finally give that t is not in $\mathsf{e}_{\mathrm{CbN}}$-normal form. This is absurd.
     - Suppose none of the cases above holds. By Lemma 13.1.1.3 (Focusing through CbN and CbNeed evaluation contexts), we would have that $u = C\langle\lambda y.m\rangle$, for some CbN evaluation context that is not a substitution context—the case where Cis a substitution context is covered above. But then Lemma 13.1.1.1 (Shape of CbN and CbNeed-normal forms) would imply that u is not in $\to_{\mathrm{CbN}}$-normal form, thus giving us that neither is $t = u[x{\leftarrow}s]$. This is absurd.

   ⇐: We prove this statement by induction on the derivation of $\mathsf{norm}(t)$, dropping the hypothesis that t is closed; thus, we prove a stronger statement. We proceed by case analysis on the last derivation rule in $\mathsf{norm}(t)$:
   - *Abstraction*: This case is trivial, as CbN does not reduce under λ-abstractions.
   - *Explicit substitution*: Let $\mathsf{norm}(t)$ be derived as follows:

     $$\frac{\mathsf{norm}(u)}{\mathsf{norm}(u[x{\leftarrow}s])}$$

     where $t = u[x{\leftarrow}s]$. By *i.h.*, u is in $\to_{\mathrm{CbN}}$-normal form. By Lemma 13.1.1.1 (Shape of CbN and CbNeed-normal forms), there exists substitution context S and $v \in \mathsf{Val}$ such that $u = S\langle v\rangle$. We can thus conclude that $t = u[x{\leftarrow}s] = (S[x{\leftarrow}s])\langle v\rangle$ is in $\mathsf{m}_{\mathrm{CbN}}$-normal form.
     Moreover, note that Lemma 13.1.1.3 (Focusing through CbN and CbNeed evaluation contexts) proves that there cannot exist any CbN evaluation context D—including substitution contexts—such that $u = D\langle x\rangle$, as we already have that $u = S\langle v\rangle$. Hence, we can also conclude that $t = u[x{\leftarrow}s]$ is in $\mathsf{e}_{\mathrm{CbN}}$-normal form.

   Next, we turn to prove that $t \in \Lambda_{\mathsf{L}}$ is in $\to_{\mathrm{CbNeed}}$-normal form if and only if $\mathsf{norm}(t)$.
   ⇒: Let t be a closed $\to_{\mathrm{CbNeed}}$-normal form. We prove the statement by structural induction on t that $\mathsf{norm}(t)$, proceeding by case analysis on the shape of t:
   - *Variable*: Impossible, because t is closed by hypothesis.
   - *Abstraction*: If $t = \lambda y.u$, then we can derive that $\mathsf{norm}(t)$.

- *Application*: Suppose $t = us$. Then u would be closed and in $\to_{\text{CbNeed}}$-normal form—or else t would not be so—and hence application of the *i.h.* would give that $\mathsf{norm}(u)$ by *i.h.*. However, Lemma 13.1.1.1 (Shape of CbN and CbNeed-normal forms) would then give that $u = S\langle\lambda y.m\rangle$, and so $t = S\langle\lambda y.m\rangle s$. Absurd, because t is in m_{CbN}-normal form.
- *Explicit substitution*: Let $t = u[x{\leftarrow}s]$. Note that since t is in $\to_{\text{CbNeed}}$-normal form, then so must be u. There are four possible cases:
  - If $u = S\langle v\rangle$, then application of Lemma 13.1.1.1 (Shape of CbN and CbNeed-normal forms) gives that $\mathsf{norm}(u)$. Hence, $\mathsf{norm}(u[x{\leftarrow}s])$.
  - If u is closed, then we can apply the *i.h.* on it and get that $\mathsf{norm}(u)$. Hence, $\mathsf{norm}(u[x{\leftarrow}s])$.
  - Suppose $u \coloneqq E\langle\!\langle y\rangle\!\rangle$. But then it must be—since t is closed—that

    $$t = E'\langle\langle\cdot\rangle\rangle E\langle\!\langle y\rangle\!\rangle[y{\leftarrow}m]$$

    for some CbNeed context E' and $m \in \Lambda_{\mathsf{L}}$, which would finally give that t is not in $\mathsf{e}_{\text{CbNeed}}$-normal form. This is absurd.
  - Suppose none of the cases above holds. By Lemma 13.1.1.3 (Focusing through CbN and CbNeed evaluation contexts), we would have that $u = C\langle\lambda y.m\rangle$, for some CbN evaluation context that is not a substitution context—the case where Cis a substitution context is covered above. But then Lemma 13.1.1.1 (Shape of CbN and CbNeed-normal forms) would imply that u is not in $\to_{\text{CbN}}$-normal form. Say $u = E\langle\tilde{t}\rangle \to_{\text{CbN}} E\langle\tilde{t'}\rangle$. But since E is also a CbNeed evaluation context—by Lemma 13.1.1.4—then u would not be in $\to_{\text{CbNeed}}$-normal form, thus giving us that neither is $t = u[x{\leftarrow}s]$. This is absurd.

$\Leftarrow$**:** We prove this statement by induction on the derivation of $\mathsf{norm}(t)$, dropping the hypothesis that t is closed; thus, we are proving a stronger statement. We proceed by case analysis on the last derivation rule in $\mathsf{norm}(t)$:

- *Abstraction*: This case is trivial, as CbN does not reduce under λ-abstractions.
- *Explicit substitution*: Let $\mathsf{norm}(t)$ be derived as follows:

  $$\frac{\mathsf{norm}(u)}{\mathsf{norm}(u[x{\leftarrow}s])}$$

  where $t = u[x{\leftarrow}s]$. By *i.h.*, u is in $\to_{\text{CbNeed}}$-normal form. By Lemma 13.1.1.1 (Shape of CbN and CbNeed-normal forms), there exists substitution context S and $v \in \mathsf{Val}$ such that $u = S\langle v\rangle$. We can thus conclude that $t = u[x{\leftarrow}s] = (S[x{\leftarrow}s])\langle v\rangle$ is in $\mathsf{m}_{\text{CbNeed}}$-normal form.
  Moreover, note that Lemma 13.1.1.3 (Focusing through CbN and CbNeed evaluation contexts) proves that there cannot exist any CbN evaluation context E'—including substitution contexts—such that $u = E'\langle x\rangle$, as we already have that $u = S\langle v\rangle$. Hence, we can also conclude that $t = u[x{\leftarrow}s]$ is in $\mathsf{e}_{\text{CbNeed}}$-normal form.

2. Finally, we prove that t is in $\to_{\text{CbV}}$-normal form if and only if $\mathsf{norm}_{\text{CbV}}(t)$.

   $\Rightarrow$**:** Let t be a closed $\to_{\text{CbV}}$-normal form. We prove that $\mathsf{norm}_{\text{CbV}}(t)$ proceeding by induction on the shape of t:

   - *Variable*: Impossible, because t is closed by hypothesis.
   - *Abstraction*: If $t = \lambda y.u$, then we can derive $\mathsf{norm}_{\text{CbV}}(t)$.

- *Application*: Suppose $t = us$. Then u would be closed and in $\to_{\mathrm{CbV}}$-normal form—or else t would not be so—and hence application of the *i.h.* would give that $\mathsf{norm}_{\mathrm{CbV}}(u)$ by *i.h.*. However, Lemma 13.1.1.2 (Shape of CbV-normal forms) would then give that $u = S\langle\lambda y.m\rangle$—with some irrelevant restrictions on S in this analysis—and so $t = S\langle\lambda y.m\rangle s$. Absurd, because t is in $\mathsf{m}_{\mathrm{CbV}}$-normal form.
- *Explicit substitution*: Let $t = u[x{\leftarrow}s]$. Note that since t is in $\to_{\mathrm{CbV}}$-normal form, then so must be u and s. In addition, s must be closed—since t is—and so $\mathsf{norm}_{\mathrm{CbV}}(s)$. There are four possible cases:
  - If $u = v[x_1{\leftarrow}t_1]...[x_n{\leftarrow}t_n]$, for some $n \geq 0$, then it must be that each t_i, for $0 \leq i \leq n$ is in $\to_{\mathrm{CbV}}$-normal form—since u is—thus giving that $\mathsf{norm}_{\mathrm{CbV}}(t_i)$. We can then derive $\mathsf{norm}_{\mathrm{CbV}}(t)$ since $t = u[x{\leftarrow}s] = v[x_1{\leftarrow}t_1]...[x_n{\leftarrow}t_n][x{\leftarrow}s]$.
  - If u is closed, then we can apply the *i.h.* on it and get that $\mathsf{norm}_{\mathrm{CbV}}(u)$. Hence, $\mathsf{norm}_{\mathrm{CbV}}(u[x{\leftarrow}s])$.
  - Suppose $u \coloneqq V\langle\!\langle y \rangle\!\rangle$. But then it must be—since t is closed—that $t = V'\langle\!\langle V\langle\!\langle y \rangle\!\rangle[y{\leftarrow}m] \rangle\!\rangle$, for some CbV context V' and term $m \in \Lambda$, which would finally give that t is not in $\mathsf{e}_{\mathrm{CbV}}$-normal form—because $\mathsf{norm}_{\mathrm{CbV}}(s)$ implies, via Lemma 13.1.1.2 (Shape of CbV-normal forms), that s is an answer. This is absurd.
  - Suppose none of the cases above holds. By Lemma 13.1.1.3 (Focusing through CbN and CbNeed evaluation contexts), we would have that $u = C\langle\lambda y.m\rangle$, for some CbN evaluation context that is not a substitution context—the case where C is a substitution context is covered above. Note that Lemma 13.1.1.4 gives that C is also a CbV evaluation context.
    But then Lemma 13.1.1.1 (Shape of CbN and CbNeed-normal forms) would imply that u is not in $\to_{\mathrm{CbN}}$-normal form. Say $u = E\langle\tilde{t}\rangle \to_{\mathrm{CbN}} E\langle\tilde{t'}\rangle$. But since E is also a CbV evaluation context—by Lemma 13.1.1.4—then u would not be in $\to_{\mathrm{CbV}}$-normal form, thus giving us that neither is $t = u[x{\leftarrow}s]$. This is absurd.

$\Leftarrow$: We prove this statement by induction on the derivation of $\mathsf{norm}_{\mathrm{CbV}}(t)$, dropping the hypothesis that t is closed; thus, we prove a stronger statement. We proceed by case analysis on the last derivation rule in $\mathsf{norm}_{\mathrm{CbV}}(t)$:

- *Abstraction*: This case is trivial, as CbV does not reduce under λ-abstractions.
- *Explicit substitution*: Let $\mathsf{norm}_{\mathrm{CbV}}(t)$ be derived as follows:

$$\frac{\mathsf{norm}_{\mathrm{CbV}}(u) \quad \mathsf{norm}_{\mathrm{CbV}}(s)}{\mathsf{norm}_{\mathrm{CbV}}(u[x{\leftarrow}s])}$$

  where $t = u[x{\leftarrow}s]$. By *i.h.*, u is in $\to_{\mathrm{CbV}}$-normal form. By Lemma 13.1.1.1 (Shape of CbV-normal forms), $t = v[x_1{\leftarrow}t_1]...[x_n{\leftarrow}t_n][x{\leftarrow}s]$, for some $n \geq 0$ such that $\mathsf{norm}_{\mathrm{CbV}}(t_i)$ for every $0 \leq i \leq n$. By *i.h.*, t_i is in $\to_{\mathrm{CbV}}$-normal form. Hence, it is clear that $t = v[x_1{\leftarrow}t_1]...[x_n{\leftarrow}t_n][x{\leftarrow}s][x{\leftarrow}s]$ is in $\mathsf{m}_{\mathrm{CbV}}$-normal form.
  Moreover, $t \neq V\langle\!\langle x \rangle\!\rangle$ for any CbV context V, and so t is also in $\mathsf{e}_{\mathrm{CbV}}$-normal form.

□

(Click here to go back to main chapter.)

13.2 Proofs of Chapter 5 (Multi types for CbN, CbV and CbNeed)

13.2.1 CbN correctness

Lemma 13.2.1 (Relevance of the CbN type system)**.**
Let $t \in \Lambda_{\mathsf{L}}$ and let $\Phi \rhd_{CbN} \Gamma \vdash^{(m,e)} t : L$ be a type derivation. If $x \notin \mathsf{fv}(t)$ then $x \notin \mathsf{dom}(\Gamma)$.

Proof. (Click here to go back to main chapter.)
By structural induction on $\Phi \rhd_{\mathrm{CbN}} \Gamma \vdash^{(m,e)} t : L$ and proceeding by case analysis on the last derivation rule of Φ. The cases of rules ax, norm, many and app are all trivial. The only two (relatively) interesting cases are those of rules fun and ES; we only cover the first one here:
Let Φ be derived as follows:

$$\frac{\Psi \rhd_{\mathrm{CbN}} \Gamma; x : M \vdash^{(m,e)} u : L'}{\Gamma \vdash^{(m,e)} \lambda x.u : M \multimap L'}\ \mathsf{fun}$$

where $t = \lambda x.u$ and $L = M \multimap L'$. By *i.h.* on Ψ, $\mathsf{dom}(\Gamma; x : M) \subseteq \mathsf{fv}(u)$.
On the one hand, if $M = \mathbf{0}$, then $\mathsf{dom}(\Gamma) \subseteq \mathsf{fv}(u)$. Then, if $x \notin \mathsf{fv}(u)$ we have that $\mathsf{dom}(\Gamma) \subseteq \mathsf{fv}(u) = \mathsf{fv}(t)$, and if $x \in \mathsf{fv}(u)$ we have that $\mathsf{dom}(\Gamma) = \mathsf{dom}(\Gamma; x : M) \setminus \{x\} \subseteq \mathsf{fv}(u) \setminus \{x\} = \mathsf{fv}(\lambda x.u)$.
On the other hand, if $M \neq \mathbf{0}$, then $\mathsf{dom}(\Gamma) \cup \{x\} \subseteq \mathsf{fv}(u)$, and so $\mathsf{dom}(\Gamma) = \mathsf{dom}(\Gamma; x : M) \setminus \{x\} = \mathsf{fv}(u) \setminus \{x\} = \mathsf{fv}(\lambda x.u)$

□

(Click here to go back to main chapter.)

Proposition 13.2.2 (Typing properties of CbN-normal forms)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be such that $\mathsf{norm}(t)$, and $\Phi \rhd_{CbN} \Gamma \vdash^{(m,e)} t : \mathsf{norm}$ be a type derivation. Then $\Gamma = \emptyset$ and $(m, e) = (0, 0)$.

Proof. (Click here to go back to main chapter.)
By induction on the derivation of $\mathsf{norm}(t)$:

- *Base case*: Let $t \coloneqq \lambda x.u$, with $\mathsf{norm}(\lambda x.u)$. Thus, Φ can only be derived as follows

$$\frac{}{\emptyset \vdash^{(0,0)} \lambda x.u : \mathsf{norm}}\ \mathsf{norm}$$

  which satisfies the statement.
- *Inductive case*: Let $\mathsf{norm}(t)$ be derived as follows

$$\frac{\mathsf{norm}(u)}{\mathsf{norm}(u[x \leftarrow s])}$$

  with $t = u[x \leftarrow s]$. Thus, Φ has the following shape.

$$\frac{\Psi \rhd_{\mathrm{CbN}} \Pi; x : M \vdash^{(m',e')} u : \mathsf{norm} \quad \Theta \rhd_{\mathrm{CbN}} \Delta \vdash^{(m'',e'')} s : M \quad M \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m+m',e+e')} u[x \leftarrow s] : \mathsf{norm}}\ \mathrm{ES}$$

  with $\Gamma = \Pi \uplus \Delta$ and $(m, e) = (m + m', e + e')$. By *i.h.* on Ψ, $\Gamma; x : M = \mathbf{0}$ and $(m', e') = (0, 0)$. Since $M = \mathbf{0}$, then Θ must be of the following form

$$\frac{}{\emptyset \vdash^{(0,0)} s : \mathbf{0}}\ \mathsf{many}$$

  Therefore, $\Pi \uplus \Delta = \emptyset$ and $(m + m', e + e') = (0, 0)$.

□

(Click here to go back to main chapter.)

Lemma 13.2.3 (Linear Substitution for CbN)**.**
Let $\Phi_{C\langle\langle x\rangle\rangle} \triangleright_{CbN} \Gamma; x : M \vdash^{(m,e)} C\langle\langle x\rangle\rangle : L$ *be a type derivation. Then* $e \geq 1$ *and there exists a splitting* $M = [L'] \uplus N$ *such that for every derivation* $\Psi \triangleright_{CbN} \Pi \vdash^{(m',e')} t : L'$ *there is a derivation*

$$\Phi_{C\langle\langle t\rangle\rangle} \triangleright_{CbN} (\Gamma \biguplus \Pi); x : N \vdash^{(m+m',e+e'-1)} C\langle\langle t\rangle\rangle : L$$

Proof. (Click here to go back to main chapter.)
By proceed by induction on C.

- *Empty context*: Let $C \coloneqq \langle\cdot\rangle$. Then $\Phi_{C\langle\langle x\rangle\rangle}$ must be of the following form

$$\frac{}{x : [\mathsf{norm}] \vdash^{(0,1)} x : L}\ \mathsf{ax}$$

  That is, $\Gamma = \emptyset$ and $N = \mathbf{0}$. Hence, the statement holds with respect to $\Phi_{C\langle\langle t\rangle\rangle} \coloneqq \Psi$.
- *Left on an application*: Let $C \coloneqq Du$. The last rule of Φ can only be app, and so $\Phi_{C\langle\langle x\rangle\rangle}$ has the form:

$$\frac{x\colon M_\Delta; \Delta \vdash^{(m_\Delta,e_\Delta)} D\langle\langle x\rangle\rangle : N \multimap L \quad x\colon M_\Sigma; \Sigma \vdash^{(m_\Sigma,e_\Sigma)} u : N}{x\colon (M_\Delta \biguplus M_\Sigma); (\Delta \biguplus \Sigma) \vdash^{(m_\Delta+m_\Sigma+1,e_\Delta+e_\Sigma)} D\langle\langle x\rangle\rangle u : L}\ \mathsf{app}$$

  where $\Gamma = \Delta \biguplus \Sigma$, $\Delta(x) = \Sigma(x) = \mathbf{0}$, $M_\Delta \uplus M_\Sigma = M$, $m = m_\Delta + m_\Sigma + 1$, and $e = e_\Delta + e_\Sigma$. By *i.h.*, there exists a splitting $M_\Delta = [L'] \uplus O$ such that for every derivation $\Psi \triangleright_{\mathrm{CbN}} \Pi \vdash^{(m',e')} t : L'$ there exists a derivation

$$\Phi_{D\langle\langle t\rangle\rangle} \triangleright_{\mathrm{CbN}} x : O; \Delta \biguplus \Pi \vdash^{(m_\Delta+m',e_\Delta+e'-1)} D\langle\langle t\rangle\rangle : N \multimap L$$

  By applying an app rule we obtain:

$$\frac{x : O; \Delta \uplus \Pi \vdash^{(m_\Delta+m',e_\Delta+e'-1)} D\langle\langle t\rangle\rangle : N \multimap L \quad x\colon M_\Sigma; \Sigma \vdash^{(m_\Sigma,e_\Sigma)} u : N}{\Phi_{C\langle\langle t\rangle\rangle} \triangleright_{\mathrm{CbN}} x\colon (O \uplus M_\Sigma); (\Delta \uplus \Pi \uplus \Sigma) \vdash^{(m_\Delta+m'+m_\Sigma+1,e_\Delta+e'+e_\Sigma-1)} D\langle\langle t\rangle\rangle u : L}\ \mathsf{app}$$

  Now, by defining $N \coloneqq O \uplus M_\Sigma$, we obtain $M = M_\Delta \uplus M_\Sigma = [L'] \uplus O \uplus M_\Sigma = [L'] \uplus N$. Therefore by applying the equalities on the type context the last obtained judgement is in fact:

$$x\colon N; (\Gamma \uplus \Pi) \vdash^{(m_\Delta+m'+m_\Sigma+1,e_\Delta+e'+e_\Sigma-1)} D\langle\langle t\rangle\rangle u : L$$

  and by applying those on the indices we obtain:

$$x\colon N; (\Gamma \uplus \Pi) \vdash^{(m+m',e+e'-1)} D\langle\langle t\rangle\rangle u : L$$

  as required.
- *Left of a substitution*: Let $C \coloneqq D[y \leftarrow u]$. Note that $x \neq y$, because the hypothesis $C\langle\langle x\rangle\rangle$ implies that C does not capture x.
  The last rule of $\Phi_{C\langle\langle x\rangle\rangle}$ can only be ES, and so Φ has the form:

$$\frac{x\colon M_\Delta; y\colon M'; \Delta \vdash^{(m_\Delta,e_\Delta)} D\langle\langle x\rangle\rangle : L \quad x\colon M_\Sigma; \Sigma \vdash^{(m_\Sigma,e_\Sigma)} u : M'}{x\colon (M_\Delta \uplus M_\Sigma); (\Delta \uplus \Sigma) \vdash^{(m_\Delta+m_\Sigma,e_\Delta+e_\Sigma)} D\langle\langle x\rangle\rangle[y \leftarrow u] : L}\ \mathrm{ES}$$

where $\Gamma = \Delta \uplus \Sigma$, $\Delta(x) = \Sigma(x) = \mathbf{0}$, $M_\Delta \uplus M_\Sigma = M$, $m = m_\Delta + m_\Sigma$, and $e = e_\Delta + e_\Sigma$.
By *i.h.*, there exists a splitting $M_\Delta = [L'] \uplus O$ such that for every derivation $\Psi \rhd_{\mathrm{CbN}} \Pi \vdash^{(m',e')} t : L'$ there exists a derivation

$$\Phi_{D\langle\langle t\rangle\rangle} \rhd_{\mathrm{CbN}} x : O; y\colon M'; \Delta \uplus \Pi \vdash^{(m_\Delta + m', e_\Delta + e' - 1)} D\langle\langle t\rangle\rangle : L$$

Note that by Lemma 5.2.1 and the fact that we are working up to α-equivalence, we can prove that $y \notin \mathsf{dom}(\Pi)$. By applying a rule ES we obtain

$$\frac{x : O; y\colon M'; \Delta \uplus \Pi \vdash^{(m_\Delta + m', e_\Delta + e' - 1)} D\langle\langle t\rangle\rangle : L \quad x\colon M_\Sigma; \Sigma \vdash^{(m_\Sigma, e_\Sigma)} u : M'}{\Phi_{C\langle\langle x\rangle\rangle} \rhd_{\mathrm{CbN}} x : O \uplus M_\Sigma; \Delta \uplus \Pi \uplus \Sigma \vdash^{(m_\Delta + m' + m_\Sigma, e_\Delta + e' + e_\Sigma - 1)} D\langle\langle t\rangle\rangle[y \leftarrow u] : L}\ \mathrm{ES}$$

Now, by defining $N \coloneqq O \uplus M_\Sigma$, we obtain $M = M_\Delta \uplus M_\Sigma = [L'] \uplus O \uplus M_\Sigma = [L'] \uplus N$. Therefore by applying the equalities on the type context the last obtained judgement is in fact:

$$x\colon N; (\Gamma \uplus \Pi) \vdash^{(m_\Delta + m' + m_\Sigma, e_\Delta + e' + e_\Sigma - 1)} D\langle\langle t\rangle\rangle[y \leftarrow u] : L$$

and by applying those on the indices we obtain:

$$x\colon N; (\Gamma \uplus \Pi) \vdash^{(m + m', e + e' - 1)} D\langle\langle t\rangle\rangle[y \leftarrow u] : L$$

as required.

□

(Click here to go back to main chapter.)
The following is required to apply Lemma 5.2.3 (Linear Substitution for CbN) in the proof of Proposition 5.2.4.2 (Quantitative Subject Reduction for CbN - exponential case) to obtain the right indices.

Lemma 13.2.4 (Splitting multi types of CbN type derivations)**.**
Let $t \in \Lambda$, $M \coloneqq N \biguplus O$, and let $\Phi \rhd_{CbN} \Gamma \vdash^{(m,e)} t : M$ be a type derivation. Then there exist type derivations

$$\Psi \rhd_{CbN} \Pi \vdash^{(m',e')} t : N$$
$$\Theta \rhd_{CbN} \Delta \vdash^{(m'',e'')} t : O$$

such that $\Gamma = \Pi \biguplus \Delta$ and $(m, e) = (m' + m'', e' + e'')$.

Proof.
Trivial, given that the many typing rule is the only one that can derive a multi type on the right—*i.e.*, the derived type of the subject, in this case M.

□

Proposition 13.2.5 (Quantitative Subject Reduction for CbN)**.**
Let $\Phi \rhd_{CbN} \Gamma \vdash^{(m,e)} t : L$ be a type derivation.

1. Multiplicative*: If $t \to_{\mathsf{m}_{\mathrm{CbN}}} u$, then $m \geq 1$ and there exists a derivation*

$$\Psi \rhd_{CbN} \Gamma \vdash^{(m-1,e)} u : L$$

2. Exponential*: If $t \to_{\mathsf{e}_{\mathrm{CbN}}} u$, then $e \geq 1$ and there exists a derivation*

$$\Psi \rhd_{CbN} \Gamma \vdash^{(m,e-1)} u : L$$

Proof. (Click here to go back to main chapter.)

1. *Multiplicative steps*: By induction on $t \to_{\mathsf{m}_{\mathrm{CbN}}} u$. Cases:
   - *Step at top level*: Let $t \coloneqq S\langle \lambda x.s\rangle m \to_{\mathsf{m}_{\mathrm{CbN}}} S\langle s[x\leftarrow m]\rangle =: u$. This case is itself by induction on S. Two sub-cases:
     - *Empty substitution context*: Let $S \coloneqq \langle\cdot\rangle$. By construction the derivation Φ is of the form:
     $$\dfrac{\dfrac{x : M; \Gamma_s \vdash^{(m_s,e_s)} s : L}{\Gamma_s \vdash^{(m_s,e_s)} \lambda x.s : M \to L}\,\mathsf{fun} \quad \Gamma_m \vdash^{(m_m,e_m)} m : M}{\Gamma_s \uplus \Gamma_m \vdash^{(m_s+m_m+1,e_s+e_m)} (\lambda x.s)m : L}\,\mathsf{app}$$
     where $\Gamma = \Gamma_s \uplus \Gamma_m$, $m = m_s + m_m$, and $e = e_s + e_m$. Note that $m \geq 1$ as required. We can derive Ψ, satisfying the statement, as follows:
     $$\dfrac{x : M; \Gamma_s \vdash^{(m_s,e_s)} s : L \quad \Gamma_m \vdash^{(m_m,e_m)} m : M}{\Gamma_s \uplus \Gamma_m \vdash^{(m_s+m_m,e_s+e_m)} s[x\leftarrow m] : L}\,\mathrm{ES}$$
     - *Non-empty substitution context*: Let $S \coloneqq S'[y\leftarrow\tilde{t}]$. Then Φ has the following form:
     $$\dfrac{\dfrac{y : N; \Gamma_s \vdash^{(m_s,e_s)} S'\langle\lambda x.s\rangle : M \to L \quad \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}},e_{\tilde{t}})} \tilde{t} : N}{\Gamma_s \uplus \Gamma_{\tilde{t}} \vdash^{(m_s+m_{\tilde{t}},e_s+e_{\tilde{t}})} S'\langle\lambda x.s\rangle[y\leftarrow\tilde{t}] : M \to L}\,\mathrm{ES} \quad \Gamma_m \vdash^{(m_m,e_m)} m : M}{\Gamma_s \uplus \Gamma_{\tilde{t}} \uplus \Gamma_m \vdash^{(m_s+m_{\tilde{t}}+m_m+1,e_s+e_{\tilde{t}}+e_m)} S'\langle\lambda x.s\rangle[y\leftarrow\tilde{t}]m : L}\,\mathsf{app}$$
     where $\Gamma = \Gamma_s \uplus \Gamma_{\tilde{t}} \uplus \Gamma_m$, $m = m_s + m_{\tilde{t}} + m_m + 1$, and $e = e_s + e_{\tilde{t}} + e_m$. Note that $m \geq 1$ as required.
     Let us consider now the following derivation, obtained by removing the last application of rule ES from Φ:
     $$\dfrac{y : N; \Gamma_s \vdash^{(m_s,e_s)} S'\langle\lambda x.s\rangle : M \to L \quad \Gamma_m \vdash^{(m_m,e_m)} m : M}{y : N; \Gamma_s \uplus \Gamma_m \vdash^{(m_s+m_m+1,e_s+e_m)} S'\langle\lambda x.s\rangle m : L}\,\mathsf{app}$$
     By *i.h.*, there exists type derivation
     $$\Theta \triangleright\ y : N; \Gamma_s \uplus \Gamma_m \vdash^{(m_s+m_m,e_s+e_m)} S'\langle s[x\leftarrow m]\rangle : L$$
     Then, we apply rule ES with respect to y and $\tilde{t}$, obtaining the following derivation Ψ, satisfying the statement:
     $$\dfrac{y : N; \Gamma_s \uplus \Gamma_m \vdash^{(m_s+m_m,e_s+e_m)} S'\langle s[x\leftarrow m]\rangle : L \quad \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}},e_{\tilde{t}})} \tilde{t} : N}{\Gamma_s \uplus \Gamma_{\tilde{t}} \uplus \Gamma_m \vdash^{(m_s+m_{\tilde{t}}+m_m,e_s+e_{\tilde{t}}+e_m)} S'\langle s[x\leftarrow m]\rangle[y\leftarrow\tilde{t}] : L}\,\mathrm{ES}$$
   - *Contextual closure*: Let $t \coloneqq C\langle s\rangle \to_{\mathsf{m}_{\mathrm{CbN}}} C\langle m\rangle =: u$. Cases of C:
     - *Left on an application*: Let $C \coloneqq D\tilde{t}$. The last typing rule in Φ is necessarily app and Φ is of the form
     $$\dfrac{\Gamma_s \vdash^{(m_s,e_s)} D\langle s\rangle : M \multimap L \quad \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}},e_{\tilde{t}})} \tilde{t} : M}{\Gamma_s \uplus \Gamma_{\tilde{t}} \vdash^{(m_s+m_{\tilde{t}}+1,e_s+e_{\tilde{t}})} D\langle s\rangle\tilde{t} : L}\,\mathsf{app}$$
     With $\Gamma = \Gamma_s \uplus \Gamma_{\tilde{t}}$, $m = m_s + m_{\tilde{t}} + 1$, and $e = e_s + e_{\tilde{t}}$.

By *i.h.*, $m_s \geq 1$ and there exists a derivation $\Gamma_s \vdash^{(m_s-1,e_s)} D\langle m\rangle : M \multimap L$, thus allowing us to construct Ψ as follows:

$$\frac{\Gamma_s \vdash^{(m_s-1,e_s)} D\langle m\rangle : M \multimap L \quad \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}},e_{\tilde{t}})} \tilde{t} : M}{\Gamma_s \uplus \Gamma_{\tilde{t}} \vdash^{(m_s+m_{\tilde{t}},e_s+e_{\tilde{t}})} D\langle m\rangle\tilde{t} : L}\ \mathsf{app}$$

Note that $(m_s + m_{\tilde{t}}, e_s + e_{\tilde{t}}) = (m-1, e)$.

– Let $C \coloneqq D[x{\leftarrow}\tilde{t}]$. The last typing rule in Φ is necessarily ES and Φ is of the form

$$\frac{\Gamma_s, x : M \vdash^{(m_s,e_s)} D\langle s\rangle : L \quad \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}},e_{\tilde{t}})} \tilde{t} : M}{\Gamma_s \uplus \Gamma_{\tilde{t}} \vdash^{(m_s+m_{\tilde{t}},e_s+e_{\tilde{t}})} D\langle s\rangle[x{\leftarrow}\tilde{t}] : L}\ \mathrm{ES}$$

With $\Gamma = \Gamma_s \uplus \Gamma_{\tilde{t}}$, $m = m_s + m_{\tilde{t}}$, and $e = e_s + e_{\tilde{t}}$.
By *i.h.*, $m_s \geq 1$, and so $m \geq 1$, and there exists a derivation $\Gamma_s, x : M \vdash^{(m_s-1,e_s)} D\langle m\rangle : L$, thus allowing us to construct Ψ as follows:

$$\frac{\Gamma_s, x : M \vdash^{(m_s-1,e_s)} D\langle m\rangle : L \quad \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}},e_{\tilde{t}})} \tilde{t} : M}{\Gamma_s \uplus \Gamma_{\tilde{t}} \vdash^{(m_s+m_{\tilde{t}}-1,e_s+e_{\tilde{t}})} D\langle s\rangle[x{\leftarrow}\tilde{t}] : L}\ \mathrm{ES}$$

Note that $(m_s + m_{\tilde{t}} - 1, e_s + e_{\tilde{t}}) = (m-1, e)$.

2. *Exponential steps*: By induction on $t \to_{\mathsf{e}_{\mathrm{CbN}}} u$.
   - *Step at top level*: Let $t \coloneqq C\langle\!\langle x\rangle\!\rangle[x{\leftarrow}s] \to_{\mathsf{e}_{\mathrm{CbN}}} C\langle\!\langle s\rangle\!\rangle[x{\leftarrow}s] =: u$. The last typing rule in Φ is necessarily ES and Φ is of the form

$$\frac{\Phi_{C\langle\!\langle x\rangle\!\rangle} \triangleright\ \Gamma_C, x : M \vdash^{(m_C,e_C)} C\langle\!\langle x\rangle\!\rangle : L \quad \Pi \vdash^{(m',e')} s : M}{\Gamma_C \uplus \Pi \vdash^{(m_C+m_s,e_C+e_s)} C\langle\!\langle x\rangle\!\rangle[x{\leftarrow}s] : L}\ \mathrm{ES}$$

where $\Gamma = \Gamma_C \uplus \Pi$, $m = m_C + m'$, and $e = e_C + e'$.
Let $M = [L'] \uplus N$ be the splitting of M given by Lemma 5.2.3 (Linear Substitution for CbN) applied to $\Phi_{C\langle\!\langle x\rangle\!\rangle}$. By Lemma 13.2.4 (Splitting multi types of CbN type derivations), there exist type derivations

$$\Psi_{L'} \triangleright_{\mathrm{CbN}} \Pi_{L'} \vdash^{(m'_{L'},e'_{L'})} s : L'$$
$$\Psi_N \triangleright_{\mathrm{CbN}} \Pi_N \vdash^{(m'_N,e'_N)} s : N$$

such that $\Pi = \Pi_{L'} \uplus \Pi_N$, $m' = m'_{L'} + m'_N$, and $e' = e'_{L'} + e'_N$.
Now, by Lemma 5.2.3 (Linear Substitution for CbN) on $\Phi_{C\langle\!\langle x\rangle\!\rangle}$ with respect to $\Psi_{L'}$, we obtain type derivation

$$\Phi_{C\langle\!\langle s\rangle\!\rangle} \triangleright_{\mathrm{CbN}} x : N; \Gamma_C \uplus \Pi_{L'} \vdash^{(m_C+m'_{L'},e_C+e'_{L'}-1)} C\langle\!\langle s\rangle\!\rangle : L$$

Then, Ψ can be derived as follows:

$$\frac{x : N; \Gamma_C \uplus \Pi_{L'} \vdash^{(m_C+m'_{L'},e_C+e'_{L'}-1)} C\langle\!\langle s\rangle\!\rangle : L \quad \Pi_N \vdash^{(m'_N,e'_N)} s : N}{\Gamma_C \uplus \Pi_{L'} \uplus \Pi_N \vdash^{(m_C+m_{L'}+m_N,e_C+e_{L'}+e_N-1)} C\langle\!\langle s\rangle\!\rangle[x{\leftarrow}s] : L}\ \mathrm{ES}$$

Now, note that the last type judgement is in fact

$$\Gamma_C \uplus \Pi \vdash^{(m_C+m',e_C+e'-1)} C\langle\!\langle s\rangle\!\rangle[x{\leftarrow}s] : L$$

which in turn is

$$\Gamma \vdash^{(m,e-1)} C\langle\!\langle s\rangle\!\rangle[x{\leftarrow}s] : L$$

as required.

- *Contextual closure*: As in the $\to_{\mathsf{m}_{\mathrm{CbN}}}$ case. Note that indeed those cases do not depend on the details of the step itself, but only on the context enclosing it.

□

(Click here to go back to main chapter.)

Theorem 13.2.6 (Tight Correctness for CbN)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be closed and $\Phi \triangleright_{CbN} \Gamma \vdash^{(m,e)} t : L$ be a type derivation. Then there exists $u \in \Lambda_{\mathsf{L}}$ such that

1. $\mathsf{norm}(u)$,
2. *there exists a reduction sequence $d : t \to^*_{\mathrm{CbN}} u$, and*
3. $|d|_{\mathsf{m}} \leq m$ *and* $|d|_{\mathsf{e}} \leq e$.

Moreover, if Φ is tight then $(m, e) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}})$.

Proof. (Click here to go back to main chapter.)
By induction on $m + e$, and proceeding by case analysis on whether $t \to_{\mathrm{CbN}}$-reduces or not. Note that if t is in $\to_{\mathrm{CbN}}$-normal form, then we only have to prove the *moreover* part, that states that if Φ is tight then $(m, e) = (0, 0)$, which follows from Proposition 5.2.2 (Typing properties of CbN-normal forms).

Otherwise, if $t \to_{\mathrm{CbN}} s$ for some $s \in \Lambda_{\mathsf{L}}$, then there are two cases:

1. *Multiplicative steps*: Let $t \to_{\mathsf{m}_{\mathrm{CbN}}} s$. By Proposition 5.2.4.1 (Quantitative Subject Reduction for CbN - multiplicative steps), there exists $\Psi \triangleright_{\mathrm{CbN}} \Gamma \vdash^{(m-1,e)} s : L$. By *i.h.* on Ψ, there exist u and d' such that $\mathsf{norm}(u)$ and $d' : s \to^*_{\mathrm{CbN}} u$, $|d'|_{\mathsf{m}} \leq m - 1$ and $|d'|_{\mathsf{e}} \leq e$. Just note that $t \to_{\mathsf{m}} s$ and so, if $d : t \to^*_{\mathrm{CbN}} u$ is d' preceeded by such a step, we have $|d|_{\mathsf{m}} \leq m$ and $|d|_{\mathsf{e}} \leq e$. If Φ is tight, then so is Ψ. Then $|d'|_{\mathsf{m}} = m - 1$ and $|d'|_{\mathsf{e}} = e$ by *i.h.*, that give $|d|_{\mathsf{m}} = m$ and $|d|_{\mathsf{e}} = e$.
2. *Exponential steps*: Let $t \to_{\mathsf{e}_{\mathrm{CbN}}} s$. By Proposition 5.2.4.2 (Quantitative Subject Reduction for CbN - exponential steps), there exists $\Psi \triangleright_{\mathrm{CbN}} \Gamma \vdash^{(m,e-1)} s : L$. By *i.h.*, there exist u and d' such that $\mathsf{norm}(u)$ and $d' : s \to^*_{\mathrm{CbN}} u$, $|d'|_{\mathsf{m}} \leq m$ and $|d'|_{\mathsf{e}} \leq e - 1$. Just note that $t \to_{\mathsf{e}} s$ and so, if $d : t \to^*_{\mathrm{CbN}} u$ is d' preceeded by such a step, we have $|d|_{\mathsf{m}} \leq m$ and $|d|_{\mathsf{e}} \leq e$.
   Finally, if Φ is tight, then so is Ψ. Then $|d'|_{\mathsf{m}} = m$ and $|d'|_{\mathsf{e}} = e-1$ by *i.h.*, that give $|d|_{\mathsf{m}} = m$ and $|d|_{\mathsf{e}} = e$.

□

(Click here to go back to main chapter.)

13.2.2 CbN completeness

Proposition 13.2.7 (Tight typability of CbN-normal forms)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be such that $\mathsf{norm}(t)$. Then there exists a tight type derivation $\Phi \triangleright_{CbN} \emptyset \vdash^{(0,0)} t : \mathsf{norm}$.

Proof. (Click here to go back to main chapter.)
By induction on the derivation of $\mathsf{norm}(t)$:

- *Base case*: Let $t := \lambda x.u$, with $\mathsf{norm}(\lambda x.u)$. Then, we can derive Φ as follows:

$$\frac{}{\vdash^{(0,0)} \lambda x.u : \mathsf{norm}}\ \mathsf{norm}$$

- *Inductive case*: Let $\mathsf{norm}(t)$ be derived as follows

$$\frac{\mathsf{norm}(u)}{\mathsf{norm}(u[x{\leftarrow}s])}$$

where $t = u[x{\leftarrow}s]$. By *i.h.*, there exists tight type derivation $\Psi \triangleright_{\mathrm{CbN}} \vdash^{(0,0)} u : \mathsf{norm}$. Then, we can derive Φ as follows:

$$\frac{\Psi \triangleright \ \vdash^{(0,0)} u : \mathsf{norm} \quad \overline{\vdash^{(0,0)} s : \mathbf{0}}\ \mathsf{many}}{\vdash^{(0,0)} u[x{\leftarrow}s] : \mathsf{norm}}\ \mathrm{ES}$$

□

(Click here to go back to main chapter.)

Lemma 13.2.8 (Linear Removal for CbN)**.**
Let $\Phi \triangleright_{CbN} \Gamma; x : M \vdash^{(m,e)} C\langle\langle u\rangle\rangle : L$ be a type derivation, where $x \notin \mathsf{fv}(u)$. Then there exist type derivations

$$\begin{array}{lll} \Psi & \triangleright_{CbN} & \Pi \vdash^{(m',e')} u : L' \\ \Theta & \triangleright_{CbN} & \Delta; x : (M \uplus [L']) \vdash^{(m'',e'')} C\langle\langle x\rangle\rangle : L \end{array}$$

such that $\Gamma = \Pi \biguplus \Delta$ and $(m, e) = (m' + m'', e' + e'' - 1)$.

Proof. (Click here to go back to main chapter.)
By induction on C:

- *Empty context*: Let $C \coloneqq \langle\cdot\rangle$. Then $\Phi \triangleright_{\mathrm{CbN}} \Gamma; x : M \vdash^{(m,e)} u : L$. By Lemma 5.2.1 (Relevance of the CbN type system), $x \notin \mathsf{fv}(u)$ implies $M = \mathbf{0}$. Then we simply take
  - $\Phi_u \coloneqq \Phi$, that implies $L' \coloneqq L$, $\Gamma_u \coloneqq \Gamma$, $m_u \coloneqq m$, and $e_u \coloneqq e$, and
  - Φ_x defined as the axiom

$$\frac{}{x : [L] \vdash^{(0,1)} x : L}\ \mathsf{ax}$$

  and for which Γ' is empty, $m' = 0$, and $e' = 1$.

  Then the statement holds:
  - *Type contexts*: $\Gamma = \emptyset \uplus \Gamma = \emptyset \uplus \Gamma_u = \Gamma' \uplus \Gamma_u$ and
  - *Indices*: $(m, e) = (m_u, e_u) = (0 + m_u, 1 + e_u - 1) = (m' + m_u, e' + e_u - 1)$.
- *Left of an application*: Let $C \coloneqq Ds$. Then Φ has the form

$$\frac{\Phi_{D\langle u\rangle} \triangleright \ \Gamma_{D\langle u\rangle}; x : M \vdash^{(m_{D\langle u\rangle}, e_{D\langle u\rangle})} D\langle u\rangle : N \multimap L \quad \Gamma_s \vdash^{(m_s, e_s)} s : N}{\Gamma_{D\langle u\rangle} \uplus \Gamma_s; x : M \vdash^{(m_{D\langle u\rangle} + m_s + 1, e_{D\langle u\rangle} + e_s)} D\langle u\rangle s : L}\ \mathsf{app}$$

  where $x \notin \mathsf{dom}(\Gamma_s)$, by Lemma 5.2.1 (Relevance of the CbN type system), because $x \notin \mathsf{fv}(s)$ by hypothesis), $\Gamma = \Gamma_{D\langle u\rangle} \uplus \Gamma_s$, $m = m_{D\langle u\rangle} + m_s + 1$, and $e = e_{D\langle u\rangle} + e_s$.
  Applying the *i.h.* to $\Phi_{D\langle u\rangle}$ provides a type L' and derivations:

$$\begin{array}{l} \Phi_u \triangleright_{\mathrm{CbN}} \Gamma_u \vdash^{(m_u, e_u)} u : L' \\ \Phi_{D\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbN}} \Gamma''; x : M \uplus [L'] \vdash^{(m'', e'')} D\langle\langle x\rangle\rangle : N \multimap L \end{array}$$

  such that $\Gamma_{D\langle u\rangle} = \Gamma'' \uplus \Gamma_u$ and $(m_{D\langle u\rangle}, e_{D\langle u\rangle}) = (m'' + m_u, e'' + e_u - 1)$.

Then $\Phi_{C\langle\langle x\rangle\rangle}$ is given by:

$$\frac{\Phi_{D\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbN}} \Gamma''; x : M \uplus [L'] \vdash^{(m'',e'')} D\langle\langle x\rangle\rangle : N \multimap L \quad \Gamma_s \vdash^{(m_s,e_s)} s : N}{(\Gamma'' \uplus \Gamma_s); x : M \uplus [L'] \vdash^{(m''+m_s+1,e''+e_s)} D\langle x\rangle s : L}\ \mathsf{app}$$

that, by taking $\Gamma' \coloneqq \Gamma'' \uplus \Gamma_s$, $m' = m'' + m_s + 1$, and $e' = e'' + e_s$, verifies the statement because:

- *Type contexts*: $\Gamma = \Gamma_{D\langle u\rangle} \uplus \Gamma_s = \Gamma'' \uplus \Gamma_u \uplus \Gamma_s = \Gamma' \uplus \Gamma_u$, and
- *Indices*: $(m, e) = (m_{D\langle u\rangle} + m_s + 1, e_{D\langle u\rangle} + e_s) = (m'' + m_u + m_s + 1, e'' + e_u - 1 + e_s) = (m' + m_u, e' + e_u - 1)$.

- *Left of a substitution*: Let $C \coloneqq D[y{\leftarrow}s]$. Then Φ has the form

$$\frac{\Phi_{D\langle u\rangle} \triangleright\ \Gamma_{D\langle u\rangle}; x : M; y : N \vdash^{(m_{D\langle u\rangle}, e_{D\langle u\rangle})} D\langle u\rangle : L \quad \Gamma_s \vdash^{(m_s,e_s)} s : N}{\Gamma_{D\langle u\rangle} \uplus \Gamma_s; x : M \vdash^{(m_{D\langle u\rangle}+m_s, e_{D\langle u\rangle}+e_s)} D\langle u\rangle[y{\leftarrow}s] : L}\ \mathrm{ES}$$

where $x \notin \mathsf{dom}(\Gamma_s)$, by Lemma 5.2.1 (Relevance of the CbN type system), because $x \notin \mathsf{fv}(s)$ by hypothesis), $\Gamma = \Gamma_{D\langle u\rangle} \uplus \Gamma_s$, $m = m_{D\langle u\rangle} + m_s$, and $e = e_{D\langle u\rangle} + e_s$.
Applying the *i.h.* to $\Phi_{D\langle u\rangle}$ provides a type L' and derivations:

$$\begin{array}{l}\Phi_u \triangleright_{\mathrm{CbN}} \Gamma_u \vdash^{(m_u,e_u)} u : L' \\ \Phi_{D\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbN}} \Gamma''; x : M \uplus [L']; y : N \vdash^{(m'',e'')} D\langle\langle x\rangle\rangle : L\end{array}$$

such that $\Gamma_{D\langle u\rangle} = \Gamma'' \uplus \Gamma_u$ and $(m_{D\langle u\rangle}, e_{D\langle u\rangle}) = (m'' + m_u, e'' + e_u - 1)$.
Then $\Phi_{C\langle\langle x\rangle\rangle}$ is given by:

$$\frac{\Phi_{D\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbN}} \Gamma''; x : M \uplus [L']; y : N \vdash^{(m'',e'')} D\langle\langle x\rangle\rangle : L \quad \Gamma_s \vdash^{(m_s,e_s)} s : N}{(\Gamma'' \uplus \Gamma_s); x : M \uplus [L'] \vdash^{(m''+m_s,e''+e_s)} D\langle x\rangle[y{\leftarrow}s] : L}\ \mathrm{ES}$$

that, by taking $\Gamma' \coloneqq \Gamma'' \uplus \Gamma_s$, $m' = m'' + m_s$, and $e' = e'' + e_s$, verifies the statement because:

- *Type contexts*: $\Gamma = \Gamma_{D\langle u\rangle} \uplus \Gamma_s = \Gamma'' \uplus \Gamma_u \uplus \Gamma_s = \Gamma' \uplus \Gamma_u$, and
- *Indices*: $(m, e) = (m_{D\langle u\rangle} + m_s, e_{D\langle u\rangle} + e_s) = (m'' + m_u + m_s, e'' + e_u - 1 + e_s) = (m' + m_u, e' + e_u - 1)$.

□

(Click here to go back to main chapter.)

The following is required to apply Lemma 5.2.7 (Linear Removal for CbN) in the proof of Proposition 5.2.8.2 (Quantitative Subject Expansion for CbN - exponential case) to obtain the right indices.

Lemma 13.2.9 (Merging multi types of CbN type derivations)**.**
Let $t \in \Lambda_{\mathsf{L}}$. For any two type derivations

$$\begin{array}{l}\Phi_N \triangleright_{CbN} \Gamma_N \vdash^{(m_N,e_N)} t : N \\ \Phi_O \triangleright_{CbN} \Gamma_O \vdash^{(m_O,e_O)} t : O\end{array}$$

there exists type derivation

$$\Phi_{N \uplus O} \triangleright_{CbN} \Gamma_N \uplus \Gamma_O \vdash^{(m_N+m_O, e_N+e_O)} t : N \uplus O$$

Proof.

Trivial, given that the many typing rule is the only one that can derive a multi type on the right—*i.e.*, the derived type of the subject, in this case N and O.

□

Proposition 13.2.10 (Quantitative Subject Expansion for CbN)**.**

Let $\Phi \rhd_{CbN} \Gamma \vdash^{(m,e)} u : L$ be a type derivation.

1. Multiplicative*: If $t \to_{\mathsf{m}_{\mathrm{CbN}}} u$, then there exists a derivation*

$$\Psi \rhd_{CbN} \Gamma \vdash^{(m+1,e)} t : L$$

2. Exponential*: If $t \to_{\mathsf{e}_{\mathrm{CbN}}} u$, then there exists a derivation*

$$\Psi \rhd_{CbN} \Gamma \vdash^{(m,e+1)} t : L$$

Proof. (Click here to go back to main chapter.)

1. *Multiplicative steps*: By induction on $t \to_{\mathsf{m}_{\mathrm{CbV}}} u$:
   - *Step at top level*: Let $t := S\langle \lambda x.s \rangle m \to_{\mathsf{m}_{\mathrm{CbN}}} S\langle s[x \leftarrow m] \rangle = u$. This case is itself by induction on S. Two sub-cases:
     - *Empty substitution context*: Let $S := \langle \cdot \rangle$. By construction the derivation Φ is of the form:
     $$\dfrac{\dfrac{x : M; \Gamma_s \vdash^{(m_s,e_s)} s : L}{\Gamma_s \vdash^{(m_s,e_s)} \lambda x.s : M \to L}\,\mathsf{fun} \quad \Gamma_m \vdash^{(m_m,e_m)} m : M}{\Gamma_s \uplus \Gamma_m \vdash^{(m_s+m_m+1,e_s+e_m)} (\lambda x.s)m : L}\,\mathsf{app}$$
     With $\Gamma = \Gamma_s \uplus \Gamma_m$, $m = m_s + m_m$, and $e = e_s + e_m$. Note that $m \geq 1$ as required. We construct the following derivation Ψ, verifying the statement:
     $$\dfrac{x : M; \Gamma_s \vdash^{(m_s,e_s)} s : L \quad \Gamma_m \vdash^{(m_m,e_m)} m : M}{\Gamma_s \uplus \Gamma_m \vdash^{(m_s+m_m,e_s+e_m)} s[x \leftarrow m] : L}\,\mathrm{ES}$$
     - *Non-empty substitution context*: Let $S := S'[y \leftarrow \tilde{t}]$. Then Φ has the following structure:
     $$\dfrac{\dfrac{y : N; \Gamma_s \vdash^{(m_s,e_s)} S'\langle \lambda x.s \rangle : M \to L \quad \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}},e_{\tilde{t}})} \tilde{t} : N}{\Gamma_s \uplus \Gamma_{\tilde{t}} \vdash^{(m_s+m_{\tilde{t}},e_s+e_{\tilde{t}})} S'\langle \lambda x.s \rangle[y \leftarrow \tilde{t}] : M \to L}\,\mathrm{ES} \quad \Gamma_m \vdash^{(m_m,e_m)} m : M}{\Gamma_s \uplus \Gamma_{\tilde{t}} \uplus \Gamma_m \vdash^{(m_s+m_{\tilde{t}}+m_m+1,e_s+e_{\tilde{t}}+e_m)} S'\langle \lambda x.s \rangle[y \leftarrow \tilde{t}]m : L}\,\mathsf{app}$$
     With $\Gamma = \Gamma_s \uplus \Gamma_{\tilde{t}} \uplus \Gamma_m$, $m = m_s + m_{\tilde{t}} + m_m + 1$, and $e = e_s + e_{\tilde{t}} + e_m$. Note that $m \geq 1$ as required.
     Consider the following derivation, obtained by removing the rule ES:
     $$\dfrac{y : N; \Gamma_s \vdash^{(m_s,e_s)} S'\langle \lambda x.s \rangle : M \to L \quad \Gamma_m \vdash^{(m_m,e_m)} m : M}{y : N; \Gamma_s \uplus \Gamma_m \vdash^{(m_s+m_m+1,e_s+e_m)} S'\langle \lambda x.s \rangle m : L}\,\mathsf{app}$$
     By *i.h.*, we obtain type derivation
     $$\Theta \rhd\ y : N; \Gamma_s \uplus \Gamma_m \vdash^{(m_s+m_m,e_s+e_m)} S'\langle s[x \leftarrow m] \rangle : L$$

Now, we apply a rule ES with respect to y and $\tilde{t}$, obtaining the following derivation Ψ, satisfying the statement:

$$\frac{y : N; \Gamma_s \uplus \Gamma_m \vdash^{(m_s+m_m, e_s+e_m)} S'\langle s[x{\leftarrow}m]\rangle : L \quad \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}}, e_{\tilde{t}})} \tilde{t} : N}{\Gamma_s \uplus \Gamma_{\tilde{t}} \uplus \Gamma_m \vdash^{(m_s+m_{\tilde{t}}+m_m, e_s+e_{\tilde{t}}+e_m)} S'\langle s[x{\leftarrow}m]\rangle[y{\leftarrow}\tilde{t}] : L}\ \text{ES}$$

- *Contextual closure*: We have $t \coloneqq C\langle s\rangle \to_{\mathsf{m}_{\mathrm{CbN}}} C\langle m\rangle = u$. Cases of C:
  - *Left on an application*: Let $C \coloneqq D\tilde{t}$. The last typing rule in Φ is necessarily app and Φ is of the form

$$\frac{\Gamma_s \vdash^{(m_s,e_s)} D\langle s\rangle : M \multimap L \quad \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}}, e_{\tilde{t}})} \tilde{t} : M}{\Gamma_s \uplus \Gamma_{\tilde{t}} \vdash^{(m_s+m_{\tilde{t}}+1, e_s+e_{\tilde{t}})} D\langle s\rangle\tilde{t} : L}\ \mathsf{app}$$

    With $\Gamma = \Gamma_s \uplus \Gamma_{\tilde{t}}$, $m = m_s + m_{\tilde{t}} + 1$, and $e = e_s + e_{\tilde{t}}$.
    By *i.h.*, $m_s \geq 1$ and there exists a derivation $\Gamma_s \vdash^{(m_s-1,e_s)} D\langle m\rangle : M \multimap L$, thus allowing us to construct Ψ as follows:

$$\frac{\Gamma_s \vdash^{(m_s-1,e_s)} D\langle m\rangle : M \multimap L \quad \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}}, e_{\tilde{t}})} \tilde{t} : M}{\Gamma_s \uplus \Gamma_{\tilde{t}} \vdash^{(m_s+m_{\tilde{t}}, e_s+e_{\tilde{t}})} D\langle m\rangle\tilde{t} : L}\ \mathsf{app}_b$$

    Note that $(m_s + m_{\tilde{t}}, e_s + e_{\tilde{t}}) = (m - 1, e)$.
  - Let $C = D[x{\leftarrow}\tilde{t}]$. The last typing rule in Φ is necessarily ES and Φ is of the form

$$\frac{\Gamma_s, x : M \vdash^{(m_s,e_s)} D\langle s\rangle : L \quad \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}}, e_{\tilde{t}})} \tilde{t} : M}{\Gamma_s \uplus \Gamma_{\tilde{t}} \vdash^{(m_s+m_{\tilde{t}}, e_s+e_{\tilde{t}})} D\langle s\rangle[x{\leftarrow}\tilde{t}] : L}\ \text{ES}$$

    With $\Gamma = \Gamma_s \uplus \Gamma_{\tilde{t}}$, $m = m_s + m_{\tilde{t}}$, and $e = e_s + e_{\tilde{t}}$.
    By *i.h.*, $m_s \geq 1$, and so $m \geq 1$, and there exists a derivation $\Gamma_s, x : M \vdash^{(m_s-1,e_s)} D\langle m\rangle : L$, thus allowing us to construct Ψ as follows:

$$\frac{\Gamma_s, x : M \vdash^{(m_s-1,e_s)} D\langle m\rangle : L \quad \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}}, e_{\tilde{t}})} \tilde{t} : M}{\Gamma_s \uplus \Gamma_{\tilde{t}} \vdash^{(m_s+m_{\tilde{t}}-1, e_s+e_{\tilde{t}})} D\langle s\rangle[x{\leftarrow}\tilde{t}] : L}\ \text{ES}$$

    Note that $(m_s + m_{\tilde{t}} - 1, e_s + e_{\tilde{t}}) = (m - 1, e)$.

2. *Exponential steps*: By induction on $t \to_{\mathsf{e}_{\mathrm{CbV}}} u$.
   - *Step at top level*: Let $t \coloneqq C\langle\!\langle x\rangle\!\rangle[x{\leftarrow}s] \to_{\mathsf{e}_{\mathrm{CbN}}} C\langle\!\langle s\rangle\!\rangle[x{\leftarrow}s] = u$. The last typing rule in Φ is necessarily ES and Φ is of the form

$$\frac{\Phi_{C\langle\!\langle x\rangle\!\rangle} \triangleright\ \Gamma_C, x : M \vdash^{(m_C,e_C)} C\langle\!\langle x\rangle\!\rangle : L \quad \Pi \vdash^{(m',e')} s : M}{\Gamma_C \uplus \Pi \vdash^{(m_C+m_s, e_C+e_s)} C\langle\!\langle x\rangle\!\rangle[x{\leftarrow}s] : L}\ \text{ES}$$

     With $\Gamma = \Gamma_C \uplus \Pi$, $m = m_C + m'$, and $e = e_C + e'$.
     Let $M \coloneqq [L'] \uplus N$ be the splitting of M given by application of Lemma 5.2.7 (Linear Removal for CbN) on $\Phi_{C\langle\!\langle x\rangle\!\rangle}$. By Lemma 13.2.9 (Merging multi types of CbN type derivations), there exist type derivations

$$\Psi_{L'} \triangleright_{\mathrm{CbN}} \Pi_{L'} \vdash^{(m'_{L'}, e'_{L'})} s : L'$$
$$\Psi_N \triangleright_{\mathrm{CbN}} \Pi_N \vdash^{(m'_N, e'_N)} s : N$$

such that $\Pi = \Pi_{L'} \uplus \Pi_N$, $m' = m'_{L'} + m'_N$, and $e' = e'_{L'} + e'_N$.
Now, by applying again the linear substitution lemma to $\Phi_{C\langle\langle x \rangle\rangle}$ with respect to $\Psi_{L'}$, we obtain a derivation

$$\Phi_{C\langle\langle s \rangle\rangle} \triangleright_{\mathrm{CbN}} x : N; \Gamma_C \uplus \Pi_{L'} \vdash^{(m_C + m'_{L'}, e_C + e'_{L'} - 1)} C\langle\langle s \rangle\rangle : L$$

Then Ψ is built as follows:

$$\frac{x : N; \Gamma_C \uplus \Pi_{L'} \vdash^{(m_C + m'_{L'}, e_C + e'_{L'} - 1)} C\langle\langle s \rangle\rangle : L \quad \Pi_N \vdash^{(m'_N, e'_N)} s : N}{\Gamma_C \uplus \Pi_{L'} \uplus \Pi_N \vdash^{(m_C + m_{L'} + m_N, e_C + e_{L'} + e_N - 1)} C\langle\langle s \rangle\rangle[x \leftarrow s] : L} \ \mathrm{ES}$$

Now, note that the last judgement is in fact

$$\Gamma_C \uplus \Pi \vdash^{(m_C + m', e_C + e' - 1)} C\langle\langle s \rangle\rangle[x \leftarrow s] : L$$

which in turn is

$$\Gamma \vdash^{(m, e-1)} C\langle\langle s \rangle\rangle[x \leftarrow s] : L$$

as required.

- *Contextual closure*: As in the $\rightarrow_{\mathsf{m}_{\mathrm{CbN}}}$ case. Note that indeed those cases do not depend on the details of the step itself, but only on the context enclosing it.

□

(Click here to go back to main chapter.)

Theorem 13.2.11 (Tight Completeness for CbN)**.**

Let $t \in \Lambda_{\mathsf{L}}$ be closed. If there exists $d : t \rightarrow^*_{\mathrm{CbN}} u$ for some $u \in \Lambda_{\mathsf{L}}$ in $\rightarrow_{\mathrm{CbN}}$-normal form, then there exists a type derivation $\Phi \triangleright_{CbN} \emptyset \vdash^{(|d|_{\mathsf{m}}, |d|_{\mathsf{e}})} t : \mathsf{norm}$.

Proof. *(Click here to go back to main chapter.)*

By induction on the length $|d|$ of the reduction sequence d: $t \rightarrow^*_{\mathrm{CbN}} u$:

- *Base case*: Let $k \coloneqq 0$. Then $t = u$ and t is in $\rightarrow_{\mathrm{CbN}}$-normal form. By Proposition 4.6.1.1 (Syntactic characaterization of closed normal forms - CbN), we have that $\mathsf{norm}(t)$. In addition, Proposition 5.2.6 (Tight typability of CbN-normal forms) yields tight type derivation $\Phi \triangleright_{\mathrm{CbN}} \emptyset \vdash^{(0,0)} t : \mathsf{norm}$, which satisfies the statement—in particular, because $|d|_{\mathsf{m}} = |d|_{\mathsf{e}} = 0$.
- *Inductive case*: Let $k > 0$; *i.e.*, $t \rightarrow_{\mathrm{CbN}} s \rightarrow^{k-1}_{\mathrm{CbN}} u$. Let d' be the evaluation $s \rightarrow^{k-1}_{\mathrm{CbN}} u$. By *i.h.*, there exists tight type derivation

  $$\Psi \triangleright_{\mathrm{CbN}} \emptyset \vdash^{(|d'|_{\mathsf{m}}, |d'|_{\mathsf{e}})} s : \mathsf{norm}$$

  Case analysis on the kind of reduction step in $t \rightarrow_{\mathrm{CbN}} s$:
  - *Multiplicative step*: Let $t \rightarrow_{\mathsf{m}_{\mathrm{CbN}}} s$. By Proposition 5.2.8.1 (Quantitative Subject Expansion for CbN - Multiplicative), there exists (tight) type derivation

    $$\Psi' \triangleright_{\mathrm{CbN}} \emptyset \vdash^{(|d'|_{\mathsf{m}} + 1, |d'|_{\mathsf{e}})} s : \mathsf{norm}$$

    which satisfies the statement—in particular, because $|d|_{\mathsf{m}} = |d'|_{\mathsf{m}} + 1$ and $|d|_{\mathsf{e}} = |d'|_{\mathsf{e}}$.
  - *Exponential step*: Let $t \rightarrow_{\mathsf{e}_{\mathrm{CbN}}} s$. By Proposition 5.2.8.2 (Quantitative Subject Expansion for CbN - Exponential), there exists (tight) type derivation

    $$\Psi' \triangleright_{\mathrm{CbN}} \emptyset \vdash^{(|d'|_{\mathsf{m}}, |d'|_{\mathsf{e}} + 1)} s : \mathsf{norm}$$

    which satisfies the statement—in particular, because $|d|_{\mathsf{m}} = |d'|_{\mathsf{m}}$ and $|d|_{\mathsf{e}} = |d'|_{\mathsf{e}} + 1$.

□

(Click here to go back to main chapter.)

13.2.3 CbV correctness

Lemma 13.2.12 (Relevance of the CbV type system)**.**
Let $t \in \Lambda_{\mathsf{L}}$ and let $\Phi \triangleright_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} t : L$ be a type derivation. If $x \notin \mathsf{fv}(t)$ then $x \notin \mathsf{dom}(\Gamma)$.

Proof. (Click here to go back to main chapter.)
By structural induction on $\Phi \triangleright_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} t : M$ and proceeding by case analysis on the last derivation rule of Φ.

□

(Click here to go back to main chapter.)

Proposition 13.2.13 (Typing properties of CbV-normal forms)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be such that $\mathsf{norm}_{\mathrm{CbV}}(t)$, and $\Phi \triangleright_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} t : \mathbf{0}$ be a type derivation. Then $\Gamma = \emptyset$ and $(m, e) = (0, 0)$.

Proof. (Click here to go back to main chapter.)
By Proposition 4.6.1.2 (Syntactic characterization of closed normal forms - CbV), $\mathsf{norm}_{\mathrm{CbV}}(t)$. We proceed by case analysis on the derivation of $\mathsf{norm}_{\mathrm{CbV}}(t)$:

- *Base case*: Let $t = \lambda x.u$, with $\mathsf{norm}_{\mathrm{CbV}}(\lambda x.u)$. Note that Φ can only be derived as follows

$$\frac{}{\emptyset \vdash^{(0,0)} \lambda x.u : \mathbf{0}}\ \mathsf{many}$$

  which satisfies the statement.
- *Inductive case*: Let $\mathsf{norm}_{\mathrm{CbV}}(t)$ be derived as follows

$$\frac{\mathsf{norm}_{\mathrm{CbV}}(u) \quad \mathsf{norm}_{\mathrm{CbV}}(s)}{\mathsf{norm}(u[x \leftarrow s])}$$

  with $t = u[x \leftarrow s]$. Thus, Φ has the following shape.

$$\frac{\Psi \triangleright_{\mathrm{CbV}} \Pi; x : M \vdash^{(m',e')} u : \mathsf{norm} \quad \Theta \triangleright_{\mathrm{CbV}} \Delta \vdash^{(m'',e'')} s : M \quad M \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m+m',e+e')} u[x \leftarrow s] : \mathsf{norm}}\ \mathrm{ES}$$

  with $\Gamma = \Pi \uplus \Delta$ and $(m, e) = (m + m', e + e')$. By *i.h.* on Ψ, $\Gamma; x : M = \mathbf{0}$ and $(m', e') = (0, 0)$. Since $M = \mathbf{0}$, we can also apply the *i.h.* on Θ, obtaining that it must be of the following form

$$\frac{}{\emptyset \vdash^{(0,0)} s : \mathbf{0}}\ \mathsf{many}$$

  Therefore, $\Pi \biguplus \Delta = \emptyset$ and $(m + m', e + e') = (0, 0)$.

□

(Click here to go back to main chapter.)

Lemma 13.2.14 (Linear Substitution for CbV)**.**
Let $\Phi \triangleright_{\mathrm{CbV}} \Gamma; x : M \vdash^{(m,e)} V\langle\langle x \rangle\rangle : N$ be a type derivation and let $v \in \mathsf{Val}$. Then $e \geq 1$ and there exists a splitting $M = O \uplus P$ such that for every type derivation $\Psi \triangleright_{\mathrm{CbV}} \Pi \vdash^{(m',e')} v : O$, there exists type derivation

$$\Theta \triangleright_{\mathrm{CbV}} \left(\Gamma \biguplus \Pi\right); x : P \vdash^{(m+m',e+e'-1)} V\langle\langle v \rangle\rangle : N$$

Proof. (Click here to go back to main chapter.)

By induction on V:

- *Context hole*: Let $V \coloneqq \langle\cdot\rangle$. Then $V\langle\!\langle v \rangle\!\rangle = v$ and $V\langle\!\langle x \rangle\!\rangle = x$. Hence Φ consists of only an application of rule ax, and so $N = M$ and $\mathsf{dom}(\Gamma) = \emptyset$, with $m = 0$ and $e = 1$. Let $O \coloneqq M$ and $P \coloneqq \mathbf{0}$. Thus, every $\Psi \rhd_{\mathrm{CbV}} \Pi \vdash^{(m',e')} v : O$ coincides with type derivation $\Phi' \rhd_{\mathrm{CbV}} \Gamma \uplus \Pi; x : P \vdash^{(m+m',e+e'-1)} V\langle\!\langle v \rangle\!\rangle : N$, since $\Gamma \uplus \Pi; x : P = \Gamma$ and $N = O$ and $(m + m', e + e' - 1) = (m', e')$.
- *Left of an application*: Let $V \coloneqq V't$. Then, Φ has the following form

$$\frac{x : M_1, \Gamma_1 \vdash^{(m_1,e_1)} V'\langle\!\langle x \rangle\!\rangle : [N' \multimap N] \qquad x : M_2, \Gamma_2 \vdash^{(m_2,e_2)} t : N'}{x : M, \Gamma \vdash^{(m,e)} V'\langle\!\langle x \rangle\!\rangle t : N}\ \mathsf{app}$$

  where $M = M_1 \uplus M_2$, $\Gamma = \Gamma_1 \uplus \Gamma_2$, $m = m_1 + m_2 + 1$ and $e = e_1 + e_2$. By *i.h.*, there exists a splitting $M_1 = O \uplus P'$ such that, for every derivation $\Psi \rhd_{\mathrm{CbV}} \Pi \vdash^{(m',e')} v : O$, there exists a type derivation with conclusion $(\Gamma_1 \uplus \Pi); x : P' \vdash^{(m_1+m',e_1+e'-1)} V'\langle\!\langle v \rangle\!\rangle : [N' \multimap N]$.
  Thus, we can derive Φ' as follows:

$$\frac{(\Gamma_1 \uplus \Pi); x : P' \vdash^{(m_1+m',e_1+e'-1)} V'\langle\!\langle v \rangle\!\rangle : [N' \multimap N] \qquad x : M_2; \Gamma_2 \vdash^{(m_2,e_2)} t : N'}{x : (M_2 \uplus P'); \Gamma \uplus \Pi \vdash^{(m+m',e+e'-1)} V'\langle\!\langle v \rangle\!\rangle t : N}\ \mathsf{app}$$

  where $P = M_2 \uplus P'$ and $M = M_1 \uplus M_2 = O \uplus P' \uplus M_2 = O \uplus P$.
- *Right of an application*: Let $V \coloneqq tV'$. Then, Φ has the form

$$\frac{x : M_1, \Gamma_1 \vdash^{(m_1,e_1)} t : [N' \multimap N] \qquad x : M_2; \Gamma_2 \vdash^{(m_2,e_2)} V'\langle\!\langle x \rangle\!\rangle : N'}{x : M, \Gamma \vdash^{(m,e)} t\, V'\langle\!\langle x \rangle\!\rangle : N}\ \mathsf{app}$$

  where $M = M_1 \uplus M_2$, $\Gamma = \Gamma_1 \uplus \Gamma_2$, $m = m_1 + m_2 + 1$ and $e = e_1 + e_2$. By *i.h.*, there exists a splitting $M_2 = O \uplus P'$ such that, for every derivation $\Psi \rhd_{\mathrm{CbV}} \Pi \vdash^{(m',e')} v : O$, there exists a derivation with conclusion $\Gamma_2 \uplus \Pi; x : P' \vdash^{(m_2+m',e_2+e'-1)} V'\langle\!\langle v \rangle\!\rangle : N'$. So, we can derive Φ' as follows:

$$\frac{x : M_1; \Gamma_1 \vdash^{(m_1,e_1)} t : [N' \multimap N] \qquad \Gamma_2 \uplus \Pi, ; x : P' \vdash^{(m_2+m',e_2+e'-1)} V'\langle\!\langle v \rangle\!\rangle : N'}{x : (M_1 \uplus P'); \Gamma \uplus \Pi \vdash^{(m+m',e+e'-1)} tV'\langle\!\langle v \rangle\!\rangle : N}\ \mathsf{app}$$

  where $P = M_1 \uplus P'$ and $M = M_1 \uplus M_2 = M_1 \uplus O \uplus P' = O \uplus P$.
- *Left of an ES*: Let $V \coloneqq V'[y \leftarrow t]$. We can suppose—without loss of generality—that $y \notin \mathsf{fv}(t) \cup \mathsf{fv}(v) \cup \{x\}$, and hence $y \notin \mathsf{dom}(\Pi)$ by Lemma 5.3.1 (Relevance of the CbV type system). Hence, Φ has the form:

$$\frac{x : M_1; y : N'; \Gamma_1 \vdash^{(m_1,e_1)} V'\langle\!\langle x \rangle\!\rangle : N \qquad x : M_2, \Gamma_2 \vdash^{(m_2,e_2)} t : N'}{x : M, \Gamma \vdash^{(m,e)} V'\langle\!\langle x \rangle\!\rangle [y \leftarrow t] : N}\ \mathsf{ES}$$

  where $M = M_1 \uplus M_2$, $\Gamma = \Gamma_1 \uplus \Gamma_2$, $m = m_1 + m_2$ and $e = e_1 + e_2$. By *i.h.*, there exists a splitting $M_1 = O \uplus P'$ such that, for every derivation $\Psi \rhd_{\mathrm{CbV}} \Pi \vdash^{(m',e')} v : O$, there exists a derivation of $\Gamma_1 \uplus \Pi, x : P', y : N' \vdash^{(m_1+m',e_1+e'-1)} V'\langle\!\langle v \rangle\!\rangle : N$.
  Therefore, we can construct the following derivation Φ'

$$\frac{\Gamma_1 \uplus \Pi; x : P'; y : N' \vdash^{(m_1+m',e_1+e'-1)} V'\langle\!\langle v \rangle\!\rangle : N \qquad x : M_2; \Gamma_2 \vdash^{(m_2,e_2)} t : N'}{x : (M_2 \uplus P'); \Gamma \uplus \Pi \vdash^{(m+m',e+e'-1)} V'\langle\!\langle v \rangle\!\rangle [y \leftarrow t] : N}\ \mathrm{ES}$$

  where $P = M_2 \uplus P'$ and $M = M_1 \uplus M_2 = O \uplus P' \uplus M_2 = O \uplus P$.

- *Right of an ES*: Let $V \coloneqq t[y{\leftarrow}V']$. We can suppose—without loss of generality—that $y \notin \mathsf{fv}(v) \cup \{x\}$, and hence $y \notin \mathsf{dom}(\Pi)$ by Lemma 5.3.1 (Relevance of the CbV type system).Hence, Φ has the following form:

$$\dfrac{x : M_1, y : N', \Gamma_1 \vdash^{(m_1,e_1)} t : N \qquad x : M_2, \Gamma_2 \vdash^{(m_2,e_2)} V'\langle\!\langle x \rangle\!\rangle : N'}{x : M, \Gamma \vdash^{(m,e)} t[y{\leftarrow}V'\langle\!\langle x \rangle\!\rangle] : N}\ \text{ES}$$

where $M = M_1 \uplus M_2$, $\Gamma = \Gamma_1 \biguplus \Gamma_2$, $m = m_1 + m_2$ and $e = e_1 + e_2$.By *i.h.*, there exists a splitting $M_2 = O \uplus P'$ such that, for every derivation $\Psi \rhd_{\mathrm{CbV}} \Pi \vdash^{(m',e')} v : O$, there exists type derivation $\Gamma_2 \biguplus \Pi, x : P' \vdash^{(m_2+m',e_2+e'-1)} V'\langle\!\langle v \rangle\!\rangle : N'$.
Hence, we can derive Φ' as follows:

$$\dfrac{x : M_1, y : N', \Gamma_1 \vdash^{(m_1,e_1)} t : N \qquad \Gamma_2 \biguplus \Pi, x : P' \vdash^{(m_2+m',e_2+e'-1)} V'\langle\!\langle v \rangle\!\rangle : N'}{x : M_1 \uplus P'; \Gamma \biguplus \Pi \vdash^{(m+m',e+e'-1)} t[y{\leftarrow}V'\langle\!\langle v \rangle\!\rangle] : N}\ \text{ES}$$

where $P = M_1 \uplus P'$ and $M = M_1 \uplus M_2 = M_1 \uplus O \uplus P' = O \uplus P$.

□

(Click here to go back to main chapter.)

The following is required to apply Lemma 5.3.3 (Linear Substitution for CbV) in the proof of Proposition 5.3.4.2 (Quantitative Subject Reduction for CbV - exponential case) to obtain the right indices.

Lemma 13.2.15 (Splitting multi types of CbV type derivations)**.** *Let $v \in \mathsf{Val}$, $M \coloneqq N \biguplus O$, and let $\Phi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} v : M$ be a type derivation. Then there exist type derivations*

$$\Psi \rhd_{\mathrm{CbV}} \Pi \vdash^{(m',e')} v : N$$
$$\Theta \rhd_{\mathrm{CbV}} \Delta \vdash^{(m'',e'')} v : O$$

such that $\Gamma = \Pi \biguplus \Delta$ and $(m, e) = (m' + m'', e' + e'')$.

Proof.

Trivial, given that the many typing rule is the only one that can derive a multi type on the right—*i.e.*, the derived type of the subject, in this case M—for values.

□

Proposition 13.2.16 (Quantitative Subject Reduction for CbV)**.**
Let $\Phi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} t : M$ be a type derivation.

1. Multiplicative*: If $t \to_{\mathsf{m}_{\mathrm{CbV}}} u$, then $m \geq 1$ and there exists a derivation*

$$\Psi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m-1,e)} u : M$$

2. Exponential*: If $t \to_{\mathsf{e}_{\mathrm{CbV}}} u$, then $e \geq 1$ and there exists a derivation*

$$\Psi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m,e-1)} u : M$$

Proof. (Click here to go back to main chapter.)

1. *Multiplicative steps*: By induction on $t \to_{\mathsf{m}_{\mathrm{CbV}}} u$. Cases:

- *Step at top level*: Let $t \coloneqq S\langle \lambda x.s\rangle m \mapsto_{\mathsf{m}_{\mathrm{CbV}}} S\langle s[x\leftarrow m]\rangle =: u$. This case is itself by induction on S. Two sub-cases:
  - *Empty substitution context*: Let $S \coloneqq \langle\cdot\rangle$. Then $t = S\langle \lambda x.s\rangle m = (\lambda x.s)m$ and $t' = S\langle s[x\leftarrow m]\rangle = s[x\leftarrow m]$. Hence, Φ has the following form

$$\dfrac{\dfrac{\dfrac{\Psi \triangleright_{\mathrm{CbV}} \Pi, x:O \vdash^{(m',e')} s : M}{\Pi \vdash^{(m',e')} \lambda x.s : O \multimap M}\,\mathsf{fun}}{\Pi \vdash^{(m',e')} \lambda x.s : [O \multimap M]}\,\mathsf{many} \qquad \Theta \triangleright_{\mathrm{CbV}} \Delta \vdash^{(m'',e'')} m : O}{\Pi \uplus \Delta \vdash^{(1+m'+m'',e'+e'')} (\lambda x.s)m : M}\,\mathsf{app}$$

    where $\Gamma \coloneqq \Pi \uplus \Delta$, $m \coloneqq 1 + m' + m''$ and $e \coloneqq e' + e''$.
    Therefore, $m \geq 1$. We can then Φ' as follows:

$$\dfrac{\Psi \triangleright_{\mathrm{CbV}} \Pi, x:O \vdash^{(m',e')} s : M \quad \Theta \triangleright_{\mathrm{CbV}} \Delta \vdash^{(m'',e'')} m : O}{\Gamma \vdash^{(m'+m'',e'+e'')} s[x\leftarrow m] : M}\,\mathrm{ES}$$

    where $(m' + m'', e' + e'') = (m - 1, e)$.
  - *Non-empty substitution context*: Let $S' \coloneqq [y_1\leftarrow u_1]\ldots[y_{n-1}\leftarrow u_{n-1}]$. Then, $t = S\langle \lambda x.s\rangle m = S'\langle \lambda x.s\rangle [y_n\leftarrow u_n]m$ and $t' = S\langle s[x\leftarrow m]\rangle = S'\langle s[x\leftarrow m]\rangle [y_n\leftarrow u_n]$. Hence, Φ has the following form:

$$\dfrac{\dfrac{\Psi'' \triangleright_{\mathrm{CbV}} \Pi, y_n : N_n \vdash^{(m'',e'')} S'\langle \lambda x.s\rangle : M \quad \Psi_n \triangleright_{\mathrm{CbV}} \Gamma_n \vdash^{(m_n,e_n)} u_n : N_n}{\Pi \uplus \Gamma_n \vdash^{(m''+m_n,e''+e_n)} S\langle \lambda x.s\rangle : M}\,\mathrm{ES} \qquad \Theta \triangleright_{\mathrm{CbV}} \Gamma'_0 \vdash^{(m'_0,e'_0)} m : O}{\Pi \uplus \Gamma_n \uplus \Gamma'_0 \vdash^{(m''+m_n+m'_0+1,e''+e_n+e'_0)} S\langle \lambda x.s\rangle m : M}\,\mathsf{app}$$

    where $\Gamma \coloneqq \Pi \uplus \Gamma_n \uplus \Gamma'_0$ and $(m, e) \coloneqq (m'' + m_n + m'_0 + 1, e'' + e_n + e'_0)$. Note that $m \geq 1$ as required.
    Let us consider now the following type derivation, which we call Ψ:

$$\dfrac{\Psi'' \triangleright_{\mathrm{CbV}} \Pi, y_n : N_n \vdash^{(m'',e'')} S'\langle \lambda x.s\rangle : M \qquad \Theta \triangleright_{\mathrm{CbV}} \Gamma'_0 \vdash^{(m'_0,e'_0)} m : O}{\Pi \uplus \Gamma'_0, y_n : N_n \vdash^{(m''+m'_0+1,e''+e'_0)} S'\langle \lambda x.s\rangle m : M}\,\mathsf{app}$$

    By *i.h.* on Ψ—since $S'\langle \lambda x.s\rangle m \mapsto_{\mathsf{m}} S'\langle s[x\leftarrow m]\rangle$—there exists type derivation

$$\Psi' \triangleright_{\mathrm{CbV}} \Pi \uplus \Gamma'_0, y_n : N_n \vdash^{(m',e')} S'\langle s[x\leftarrow m]\rangle : M$$

    where $(m', e') = (m'' + m'_0, e'' + e'_0)$.
    We can then derive Φ' as follows:

$$\dfrac{\Psi' \triangleright_{\mathrm{CbV}} \Pi \uplus \Gamma'_0, y_n : N_n \vdash^{(m',e')} S'\langle s[x\leftarrow m]\rangle : M \qquad \Psi_n \triangleright_{\mathrm{CbV}} \Gamma_n \vdash^{(m_n,e_n)} u_n : N_n}{\Gamma \vdash^{(m'+m_n,e'+e_n)} S'\langle s[x\leftarrow m]\rangle [y_n\leftarrow u_n] : M}\,\mathrm{ES}$$

    where $(m' + m_n, e' + e_n) = (m'' + m'_0 + m_n, e'' + e'_0 + e_n) = (m - 1, e)$.
- *Contextual closure*: Let $t \coloneqq V\langle t'\rangle \to_{\mathsf{m}_{\mathrm{CbV}}} V\langle u'\rangle =: u$, with $t \mapsto_{\mathsf{m}_{\mathrm{CbV}}} t'$. We proceed by induction on V:
  - *Left of an application*: Let $V \coloneqq V's$. Then, $t = V'\langle t'\rangle s \to_{\mathsf{m}_{\mathrm{CbV}}} V'\langle u'\rangle s = t'$. Hence, Φ has the following form:

$$\dfrac{\Psi_1 \triangleright_{\mathrm{CbV}} \Gamma_1 \vdash^{(m_1,e_1)} u : [N \multimap M] \qquad \Psi_2 \triangleright_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2+1,e_1+e_2)} t : M}\,\mathsf{app}$$

where $\Gamma = \Gamma_1 \uplus \Gamma_2$ and $m = m_1 + m_2 + 1$ and $e = e_1 + e_2$.
By application of the *i.h.* on Ψ_1, there exists type derivation $\Psi \triangleright_{\mathrm{CbV}} \Gamma_1 \vdash^{(m_1-1,e_1)} u' : [N \multimap M]$ where $m' \coloneqq m_1 - 1$ and $e' \coloneqq e_1$.
Thus, we can derive Φ' as follows:

$$\frac{\Psi \triangleright_{\mathrm{CbV}} \Gamma_1 \vdash^{(m',e')} u' : [N \multimap M] \qquad \Psi_2 \triangleright_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1-1+m_2+1,e_1+e_2)} t' : M}\ \mathsf{app}$$

– *Right of an application*: Let $V \coloneqq sV'$. Then, $t = sV'\langle t' \rangle \to_{\mathsf{m}_{\mathrm{CbV}}} sV'\langle u' \rangle = u$. Hence, Φ is of the following form:

$$\frac{\Psi_1 \triangleright_{\mathrm{CbV}} \Gamma_1 \vdash^{(m_1,e_1)} s : [N \multimap M] \qquad \Psi_2 \triangleright_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} u : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2+1,e_1+e_2)} t : M}\ \mathsf{app}$$

where $\Gamma \coloneqq \Gamma_1 \uplus \Gamma_2$ and $m \coloneqq m_1 + m_2 + 1$ and $e \coloneqq e_1 + e_2$.
By application of the *i.h.* on Ψ_2, there exists type derivation $\Psi \triangleright_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2-1,e_2)} u' : N$.
Thus, we can derive Φ' as follows:

$$\frac{\Psi_1 \triangleright_{\mathrm{CbV}} \Gamma_1 \vdash^{(m_1,e_1)} s : [N \multimap M] \qquad \Psi \triangleright_{\mathrm{CbV}} \Gamma_2 \vdash^{(m',e')} u' : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2-1+1,e_1+e_2)} t' : M}\ \mathsf{app}$$

– *Left of an ES*: Let $V \coloneqq V'[x \leftarrow s]$. Then, $t = V'\langle t' \rangle[x \leftarrow s] \to_{\mathsf{m}_{\mathrm{CbV}}} V'\langle u' \rangle[x \leftarrow s] = u$. Hence, Φ has the following form:

$$\frac{\Gamma_1, x : N \vdash^{(m_1,e_1)} u : M \qquad \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,e_1+e_2)} t : M}\ \mathrm{ES}$$

where $\Gamma \coloneqq \Gamma_1 \uplus \Gamma_2$ and $m \coloneqq m_1 + m_2$ and $e \coloneqq e_1 + e_2$.
By application of the *i.h.* on Ψ_1, there exists type derivation $\Psi \triangleright_{\mathrm{CbV}} \Gamma_1, x : N \vdash^{(m_1-1,e_1)} u' : M$.
Thus, we can derive Φ' as follows:

$$\frac{\Psi \triangleright_{\mathrm{CbV}} \Gamma_1, x : N \vdash^{(m',e')} u' : M \qquad \Psi_2 \triangleright_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1-1+m_2,e_1+e_2)} t' : M}\ \mathrm{ES}$$

– *Right of an ES*: Let $V \coloneqq sV'$. Then, $ts[x \leftarrow V'\langle t' \rangle] \to_{\mathsf{m}_{\mathrm{CbV}}} s[x \leftarrow V'\langle u' \rangle] = u$. Hence, Φ has the following form:

$$\frac{\Gamma_1, x : N \vdash^{(m_1,e_1)} u : M \qquad \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,e_1+e_2)} t : M}\ \mathrm{ES}$$

where $\Gamma \coloneqq \Gamma_1 \uplus \Gamma_2$ and $m \coloneqq m_1 + m_2$ and $e \coloneqq e_1 + e_2$.
By application of the *i.h.* on Ψ_2, there exists type derivation $\Psi \triangleright_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2-1,e_2)} s' : N$.
Thus, we can derive Φ' as follows:

$$\frac{\Psi_1 \triangleright_{\mathrm{CbV}} \Gamma_1, x : N \vdash^{(m_1,e_1)} u : M \qquad \Psi \triangleright_{\mathrm{CbV}} \Gamma_2 \vdash^{(m',e')} s' : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2-1,e_1+e_2)} t' : M}\ \mathrm{ES}$$

2. *Exponential steps*: By induction on $t \to_{\mathsf{e}_{\mathrm{CbV}}} u$. Cases:
   - *Step at top level*: Let $t \coloneqq V\langle\!\langle x \rangle\!\rangle[x \leftarrow S\langle v\rangle] \mapsto_{\mathsf{e}_{\mathrm{CbV}}} S\langle V\langle\!\langle v\rangle\!\rangle[x\leftarrow v]\rangle =: t'$ with $S \coloneqq [y_1\leftarrow u_1]\dots[y_n\leftarrow u_n]$ for some $n \geq 0$. This case is itself by induction on S. Two subcases:
     - *Empty substitution context*: Let $S \coloneqq \langle\cdot\rangle$ and so $t = V\langle\!\langle x\rangle\!\rangle[x\leftarrow v]$ and $t' = V\langle\!\langle v\rangle\!\rangle[x\leftarrow v]$. Hence, Φ has the following form

$$\frac{\Psi \rhd_{\mathrm{CbV}} \Pi, x:O \vdash^{(m',e')} V\langle\!\langle x\rangle\!\rangle : M \qquad \Theta \rhd_{\mathrm{CbV}} \Gamma_0 \vdash^{(m_0,e_0)} v : O}{\Pi \uplus \Gamma_0 \vdash^{(m'+m_0,e'+e_0)} V\langle\!\langle x\rangle\!\rangle[x\leftarrow v] : M}\ \mathrm{ES}$$

     where $\Gamma \coloneqq \Pi \uplus \Gamma_0$, and $m \coloneqq m' + m_0$ and $e \coloneqq e' + e_0$.
     Let $O = O' \uplus O''$ be the splitting of O given by Lemma 5.3.3 (Linear Substitution for CbV). By Lemma 13.2.15 (Splitting multi types of CbV type derivations), there exist a splitting $\Gamma_0 = \Gamma_0' \uplus \Gamma_0''$ and the derivations $\Theta' \rhd_{\mathrm{CbV}} \Gamma_0' \vdash^{(m_0',e_0')} v : O'$ and $\Theta'' \rhd_{\mathrm{CbV}} \Gamma_0'' \vdash^{(m_0'',e_0'')} v : O''$, with $m_0 = m_0' + m_0''$ and $e_0 = e_0' + e_0''$.
     Moreover, note that, by Lemma 5.3.3 (Linear Substitution for CbV), there exists type derivation $\Psi' \rhd_{\mathrm{CbV}} \Pi \uplus \Gamma_0', x:O'' \vdash^{(m'+m_0',e'+e_0'-1)} V\langle\!\langle v\rangle\!\rangle : M$. We can then derive Φ' as follows:

$$\frac{\Psi' \rhd_{\mathrm{CbV}} \Pi \uplus \Gamma_0', x:O'' \vdash^{(m'+m_0',e'+e_0'-1)} V\langle\!\langle v\rangle\!\rangle : M \qquad \Theta'' \rhd_{\mathrm{CbV}} \Gamma_0'' \vdash^{(m_0'',e_0'')} v : O''}{\Pi \uplus \Gamma_0' \uplus \Gamma_0'' \vdash^{(m'+m_0'+m_0'',e'+e_0'+e_0''-1)} V\langle\!\langle v\rangle\!\rangle[x\leftarrow v] : M}\ \mathrm{ES}$$

     where $\Pi \uplus \Gamma_0' \uplus \Gamma_0'' = \Pi \uplus \Gamma_0 = \Gamma$ and $(m' + m_0' + m_0'', e' + e_0' + e_0'' - 1) = (m' + m_0, e' + e_0 - 1) = (m, e-1)$.
     - *Non-empty substitution context*: Let $S' \coloneqq [y_1\leftarrow u_1]\dots[y_{n-1}\leftarrow u_{n-1}]$: then, $S\langle v\rangle = S'\langle v\rangle[y_n\leftarrow u_n]$ and so $t = V\langle\!\langle x\rangle\!\rangle[x\leftarrow S'\langle v\rangle[y_n\leftarrow u_n]]$ and $t' = S\langle V\langle\!\langle v\rangle\!\rangle[x\leftarrow v]\rangle = S'\langle V\langle\!\langle v\rangle\!\rangle[x\leftarrow v]\rangle[y_n\leftarrow u_n]$. Hence, Φ has the following form

$$\dfrac{\Theta \rhd_{\mathrm{CbV}} \Gamma_0', x:N \vdash^{(m_0',e_0')} V\langle\!\langle x\rangle\!\rangle : M \qquad \dfrac{\Theta'' \rhd_{\mathrm{CbV}} y_n:N_n, \Gamma_0'' \vdash^{(m_0'',e_0'')} S'\langle v\rangle : N \quad \Theta_n \rhd_{\mathrm{CbV}} \Gamma_n \vdash^{(m_n,e_n)} u_n : N_n}{\Gamma_0'' \uplus \Gamma_n \vdash^{(m_0''+m_n,e_0''+e_n)} S\langle v\rangle : N}\ \mathsf{ES}}{\Gamma_0' \uplus \Gamma_0'' \uplus \Gamma_n \vdash^{(m_0'+m''+m_n,e_0'+e''+e_n)} V\langle\!\langle x\rangle\!\rangle[x\leftarrow S\langle v\rangle] : M}\ \mathrm{ES}$$

     where $\Gamma \coloneqq \Gamma_0' \uplus \Gamma_0'' \uplus \Gamma_n$ and $(m, e) \coloneqq (m_0' + m'' + m_n, e_0' + e'' + e_n)$.
     Let us consider now the following derivation Ψ

$$\frac{\Theta \rhd_{\mathrm{CbV}} \Gamma_0', x:N \vdash^{(m_0',e_0')} V\langle\!\langle x\rangle\!\rangle : M \quad \Theta'' \rhd_{\mathrm{CbV}} y_n:N_n, \Gamma_0'' \vdash^{(m_0'',e_0'')} S'\langle v\rangle : N}{\Gamma_0' \uplus \Gamma_0'', y_n:N_n \vdash^{(m_0'+m'',e_0'+e'')} V\langle\!\langle x\rangle\!\rangle[x\leftarrow S'\langle v\rangle] : M}\ \mathrm{ES}$$

     By application of the *i.h.* on Ψ—since $V\langle\!\langle x\rangle\!\rangle[x\leftarrow S'\langle v\rangle] \mapsto_{\mathsf{e}_{\mathrm{CbV}}} S'\langle V\langle\!\langle v\rangle\!\rangle[x\leftarrow v]\rangle$—one has $e_0' + e'' \geq 1$ and there exists type derivation

$$\Psi' \rhd_{\mathrm{CbV}} \Gamma_0' \uplus \Gamma_0'', y_n:N_n \vdash^{(m',e')} S'\langle V\langle\!\langle v\rangle\!\rangle[x\leftarrow v]\rangle : M$$

     where $(m', e') \coloneqq (m_0' + m'', e_0' + e'' - 1)$.
     Hence, we can derive Φ' as follows:

$$\frac{\Psi' \rhd_{\mathrm{CbV}} \Gamma_0' \uplus \Gamma_0'', y_n:N_n \vdash^{(m',e')} S'\langle V\langle\!\langle v\rangle\!\rangle[x\leftarrow v]\rangle : M \qquad \Theta_n \rhd_{\mathrm{CbV}} \Gamma_n \vdash^{(m_n,e_n)} u_n : N_n}{\Gamma \vdash^{(m'+m_n,e'+e_n)} S'\langle V\langle\!\langle v\rangle\!\rangle[x\leftarrow v]\rangle[y_n\leftarrow u_n] : M}\ \mathrm{ES}$$

     where $(m' + m_n, e' + e_n) = (m_0' + m'' + m_n, e_0' + e'' - 1 + e_n) = (m, e-1)$.

- *Contextual closure*: Let $t \coloneqq V\langle t' \rangle \to_{\mathsf{e}_{\mathrm{CbV}}} V\langle u' \rangle =: u$, with $t \mapsto_{\mathsf{m}_{\mathrm{CbV}}} t'$. We proceed by induction on V:
  - *Left of an application*: Let $V \coloneqq V's$. Then, $t = V'\langle t' \rangle s \to_{\mathsf{e}_{\mathrm{CbV}}} V'\langle u' \rangle s = t'$. Hence, Φ has the following form:

$$\frac{\Psi_1 \rhd_{\mathrm{CbV}} \Gamma_1 \vdash^{(m_1,e_1)} u : [N \multimap M] \qquad \Psi_2 \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2+1,e_1+e_2)} t : M}\ \mathsf{app}$$

    where $\Gamma = \Gamma_1 \uplus \Gamma_2$ and $m = m_1 + m_2 + 1$ and $e = e_1 + e_2$.
    By application of the *i.h.* on Ψ_1, there exists type derivation $\Psi \rhd_{\mathrm{CbV}} \Gamma_1 \vdash^{(m_1,e_1-1)} u' : [N \multimap M]$.
    Thus, we can derive Φ' as follows:

$$\frac{\Psi \rhd_{\mathrm{CbV}} \Gamma_1 \vdash^{(m',e')} u' : [N \multimap M] \qquad \Psi_2 \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2+1,e_1-1+e_2)} t' : M}\ \mathsf{app}$$

  - *Right of an application*: Let $V \coloneqq sV'$. Then, $t = sV'\langle t' \rangle \to_{\mathsf{e}_{\mathrm{CbV}}} sV'\langle u' \rangle = u$. Hence, Φ is of the following form:

$$\frac{\Psi_1 \rhd_{\mathrm{CbV}} \Gamma_1 \vdash^{(m_1,e_1)} s : [N \multimap M] \qquad \Psi_2 \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} u : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2+1,e_1+e_2)} t : M}\ \mathsf{app}$$

    where $\Gamma \coloneqq \Gamma_1 \uplus \Gamma_2$ and $m \coloneqq m_1 + m_2 + 1$ and $e \coloneqq e_1 + e_2$.
    By application of the *i.h.* on Ψ_2, there exists type derivation $\Psi \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2-1)} u' : N$.
    Thus, we can derive Φ' as follows:

$$\frac{\Psi_1 \rhd_{\mathrm{CbV}} \Gamma_1 \vdash^{(m_1,e_1)} s : [N \multimap M] \qquad \Psi \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m',e')} u' : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2+1,e_1+e_2-1)} t' : M}\ \mathsf{app}$$

  - *Left of an ES*: Let $V \coloneqq V'[x \leftarrow s]$. Then, $t = V'\langle t' \rangle[x \leftarrow s] \to_{\mathsf{e}_{\mathrm{CbV}}} V'\langle u' \rangle[x \leftarrow s] = u$. Hence, Φ has the following form:

$$\frac{\Gamma_1, x : N \vdash^{(m_1,e_1)} u : M \qquad \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,e_1+e_2)} t : M}\ \mathrm{ES}$$

    where $\Gamma \coloneqq \Gamma_1 \uplus \Gamma_2$ and $m \coloneqq m_1 + m_2$ and $e \coloneqq e_1 + e_2$.
    By application of the *i.h.* on Ψ_1, there exists type derivation $\Psi \rhd_{\mathrm{CbV}} \Gamma_1, x : N \vdash^{(m_1,e_1-1)} u' : M$.
    Thus, we can derive Φ' as follows:

$$\frac{\Psi \rhd_{\mathrm{CbV}} \Gamma_1, x : N \vdash^{(m',e')} u' : M \qquad \Psi_2 \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,e_1-1+e_2)} t' : M}\ \mathrm{ES}$$

  - *Right of an ES*: Let $V \coloneqq sV'$. Then, $ts[x \leftarrow V'\langle t' \rangle] \to_{\mathsf{m}_{\mathrm{CbV}}} s[x \leftarrow V'\langle u' \rangle] = u$. Hence, Φ has the following form:

$$\frac{\Gamma_1, x : N \vdash^{(m_1,e_1)} u : M \qquad \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,e_1+e_2)} t : M}\ \mathrm{ES}$$

where $\Gamma \coloneqq \Gamma_1 \uplus \Gamma_2$ and $m \coloneqq m_1 + m_2$ and $e \coloneqq e_1 + e_2$.
By application of the *i.h.* on Ψ_2, there exists type derivation $\Psi \triangleright_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2-1)} s' : N$.
Thus, we can derive Φ' as follows:

$$\frac{\Psi_1 \triangleright_{\mathrm{CbV}} \Gamma_1, x : N \vdash^{(m_1,e_1)} u : M \qquad \Psi \triangleright_{\mathrm{CbV}} \Gamma_2 \vdash^{(m',e')} s' : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,e_1+e_2-1)} t' : M}\ \mathrm{ES}$$

□

(Click here to go back to main chapter.)

Theorem 13.2.17 (Tight correctness for CbV)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be closed and $\Phi \triangleright_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} t : M$ be a type derivation. Then there exists $u \in \Lambda_{\mathsf{L}}$ such that

1. $\mathsf{norm}_{\mathrm{CbV}}(u)$,
2. *there exists a reduction sequence $d : t \to^*_{\mathrm{CbV}} u$, and*
3. $|d|_{\mathsf{m}} \leq m$ *and* $|d|_{\mathsf{e}} \leq e$.

Moreover, if Φ is tight then $(m, e) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}})$.

Proof. (Click here to go back to main chapter.)
By induction on $m + e$, and proceeding by case analysis on whether $t \to_{\mathrm{CbV}}$-reduces or not. Note that if t is in $\to_{\mathrm{CbV}}$-normal form, then we only have to prove the *moreover* part, that states that if Φ is tight then $(m, e) = (0, 0)$, which follows from Proposition 5.3.2 (Typing properties of CbN-normal forms).
Otherwise, if $t \to_{\mathrm{CbV}} s$ for some $s \in \Lambda_{\mathsf{L}}$, then there are two cases:

1. *Multiplicative steps*: Let $t \to_{\mathsf{m}_{\mathrm{CbV}}} s$. By Proposition 5.3.4.1 (Quantitative Subject Reduction for CbV - multiplicative steps), there exists $\Psi \triangleright_{\mathrm{CbV}} \Gamma \vdash^{(m-1,e)} s : M$. By *i.h.*, there exist u and d' such that $\mathsf{norm}_{\mathrm{CbV}}(u)$ and $d' : s \to^*_{\mathrm{CbV}} u$, $|d'|_{\mathsf{m}} \leq m - 1$ and $|d'|_{\mathsf{e}} \leq e$. Just note that $t \to_{\mathsf{m}} s$ and so, if $d : t \to^*_{\mathrm{CbV}} u$ is d' preceeded by such a step, we have $|d|_{\mathsf{m}} \leq m$ and $|d|_{\mathsf{e}} \leq e$. If Φ is tight, then so is Ψ. Then $|d'|_{\mathsf{m}} = m - 1$ and $|d'|_{\mathsf{e}} = e$ by *i.h.*, that give $|d|_{\mathsf{m}} = m$ and $|d|_{\mathsf{e}} = e$.
2. *Exponential steps*: Let $t \to_{\mathsf{e}_{\mathrm{CbV}}} s$. By Proposition 5.3.4.2 (Quantitative Subject Reduction for CbV - exponential steps), there exists $\Psi \triangleright_{\mathrm{CbV}} \Gamma \vdash^{(m,e-1)} s : M$. By *i.h.* on Ψ, there exist u and d' such that $\mathsf{norm}_{\mathrm{CbV}}(u)$ and $d' : s \to^*_{\mathrm{CbV}} u$, $|d'|_{\mathsf{m}} \leq m$ and $|d'|_{\mathsf{e}} \leq e - 1$. Just note that $t \to_{\mathsf{e}} s$ and so, if $d : t \to^*_{\mathrm{CbV}} u$ is d' preceeded by such a step, we have $|d|_{\mathsf{m}} \leq m$ and $|d|_{\mathsf{e}} \leq e$. Finally, if Φ is tight, then so is Ψ. Then $|d'|_{\mathsf{m}} = m$ and $|d'|_{\mathsf{e}} = e - 1$ by *i.h.*, that give $|d|_{\mathsf{m}} = m$ and $|d|_{\mathsf{e}} = e$.

□

(Click here to go back to main chapter.)

13.2.4 CbV completeness

Proposition 13.2.18 (Tight typability of CbV -normal forms)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be such that $\mathsf{norm}_{\mathrm{CbV}}(t)$. Then there exists tight type derivation $\Phi \triangleright_{\mathrm{CbV}} \emptyset \vdash^{(0,0)} t : \mathbf{0}$.

Proof. (Click here to go back to main chapter.)

By Proposition 4.6.1.2 (Syntactic characterization of closed normal forms - CbV), we have that $\mathsf{norm}_{\mathrm{CbV}}(t)$. The statement is then proven by induction on the derivation of $\mathsf{norm}_{\mathrm{CbV}}(t)$, proceeding by case analysis on the last derivation rule:

- *Abstraction*: Let $\mathsf{norm}_{\mathrm{CbV}}(t)$ derived as follows:

$$\frac{}{\mathsf{norm}_{\mathrm{CbV}}(\lambda x.u)}$$

  where $t = \lambda x.u$. Then, we can derive Φ as follows:

$$\frac{}{\emptyset \vdash^{(0,0)} \lambda x.u : \mathbf{0}}\ \mathsf{many}$$

- *ES*: Let $\mathsf{norm}_{\mathrm{CbV}}(t)$ be derived as follows:

$$\frac{\mathsf{norm}_{\mathrm{CbV}}(u) \quad \mathsf{norm}_{\mathrm{CbV}}(s)}{\mathsf{norm}_{\mathrm{CbV}}(u[x{\leftarrow}s])}$$

  where $t = u[x{\leftarrow}s]$. By separate applications of the *i.h.* on $\mathsf{norm}_{\mathrm{CbV}}(u)$ and $\mathsf{norm}_{\mathrm{CbV}}(s)$, there exist type derivations

$$\Psi \rhd_{\mathrm{CbV}} \emptyset \vdash^{(0,0)} u : \mathbf{0}$$
$$\Theta \rhd_{\mathrm{CbV}} \emptyset \vdash^{(0,0)} s : \mathbf{0}$$

  We can then derive type derivation Φ for t as follows:

$$\frac{\Psi \rhd_{\mathrm{CbV}} \emptyset; x : \mathbf{0} \vdash^{(0,0)} u : \mathbf{0} \qquad \Theta \rhd_{\mathrm{CbV}} \emptyset \vdash^{(0,0)} s : \mathbf{0}}{\emptyset \vdash^{(0,0)} u[x{\leftarrow}s] : \mathbf{0}}\ \mathrm{ES}$$

□

(Click here to go back to main chapter.)

Lemma 13.2.19 (Linear Removal for CbV)**.**

Let $\Phi \rhd_{\mathrm{CbV}} \Gamma; x : M \vdash^{(m,e)} V\langle\langle v \rangle\rangle : N$ be a type derivation, where $v \in \mathsf{Val}$ and $x \notin \mathsf{fv}(v)$. Then there exist type derivations

$$\Psi \quad \rhd_{\mathrm{CbV}} \quad \Pi \vdash^{(m',e')} v : O$$
$$\Theta \quad \rhd_{\mathrm{CbV}} \quad \Delta; x : (M \uplus O) \vdash^{(m'',e'')} V\langle\langle x \rangle\rangle : N$$

such that $\Gamma = \Pi \biguplus \Delta$ and $(m, e) = (m' + m'', e' + e'' - 1)$.

Proof. (Click here to go back to main chapter.)

By induction on V:

- *Context hole*: Let $V \coloneqq \langle\cdot\rangle$; that is, $V\langle\langle x \rangle\rangle = x$ and $V\langle\langle v \rangle\rangle = v$.
  Note that since $x \notin \mathsf{fv}(v)$, then Lemma 5.3.1 (Relevance of the CbV type system) gives that $M = \mathbf{0}$. Thus, let $O \coloneqq N$ and $\Pi \coloneqq \Gamma$ and $\Delta \coloneqq \mathbf{0}$–noting that $\Gamma = \Pi \uplus \Delta$. The statement holds by taking $\Psi \coloneqq \Phi$ and deriving Θ as follows:

$$\frac{}{x : O \vdash^{(0,1)} x : O}\ \mathsf{ax}$$

- *Left of an application*: Let $V = V't$; that is, $V\langle\!\langle x\rangle\!\rangle = V'\langle\!\langle x\rangle\!\rangle t$ and $V\langle\!\langle v\rangle\!\rangle = V'\langle\!\langle v\rangle\!\rangle t$. Hence, Φ has the following form

$$\frac{\Phi_1 \triangleright_{\mathrm{CbV}} \Gamma_1, x : M_1 \vdash^{(m_1,e_1)} V'\langle\!\langle v\rangle\!\rangle : [O \multimap N] \qquad \Phi_2 \triangleright_{\mathrm{CbV}} \Gamma_2, x : M_2 \vdash^{(m_2,e_2)} t : O}{\Gamma, x : M \vdash^{(m,e)} V'\langle\!\langle v\rangle\!\rangle t : N}\ \mathsf{app}$$

  where $\Gamma \coloneqq \Gamma_1 \uplus \Gamma_2$ and $M \coloneqq M_1 \uplus M_2$ and $(m, e) \coloneqq (m_1 + m_2 + 1, e_1 + e_2)$. By *i.h.*, there exist multi type M', two type contexts Γ_1' and Π_1, and two derivations $\Psi_1 \triangleright_{\mathrm{CbV}} \Pi_1, x : M_1 \uplus M' \vdash^{(m_1'', e_1'')} V'\langle\!\langle x\rangle\!\rangle : [O \multimap N]$ and $\Phi' \triangleright_{\mathrm{CbV}} \Gamma' \vdash^{(m', e')} v : M'$ such that $\Gamma_1 = \Gamma' \uplus \Pi_1$ ad $(m_1, e_1) = (m' + m_1'', e' + e_1'' - 1)$. We can then construct the following derivation Ψ

$$\frac{\Psi_1 \triangleright_{\mathrm{CbV}} \Pi_1, x : M_1 \uplus M' \vdash^{(m_1'', e_1'')} V'\langle\!\langle x\rangle\!\rangle : [O \multimap N] \qquad \Phi_2 \triangleright_{\mathrm{CbV}} \Gamma_2, x : M_2 \vdash^{(m_2,e_2)} t : O}{\Pi_1 \uplus \Gamma_2, x : M_1 \uplus M' \uplus M_2 \vdash^{(m_1'' + m_2 + 1, e_1'' + e_2)} V'\langle\!\langle x\rangle\!\rangle t : N}\ \mathsf{app}$$

  where $M_1 \uplus M' \uplus M_2 = M \uplus M'$. If we set $\Pi \coloneqq \Pi_1 \uplus \Gamma_2$ and $(m'', e'') \coloneqq (m_1'' + m_2 + 1, e_1'' + e_2)$, then we have $\Gamma' \uplus \Pi = \Gamma' \uplus \Pi_1 \uplus \Gamma_2 = \Gamma_1 \uplus \Gamma_2 = \Gamma$ and $(m' + m'', e' + e'' - 1) = (m' + m_1'' + m_2 + 1, e' + e_1'' + e_2 - 1) = (m_1 + m_2 + 1, e_1 + e_2) = (m, e)$, as required.
- *Right application, i.e.* $V = tV'$: then, $V\langle\!\langle x\rangle\!\rangle = tV'\langle\!\langle x\rangle\!\rangle$ and $V\langle\!\langle v\rangle\!\rangle = tV'\langle\!\langle v\rangle\!\rangle$. So, Φ has the form

$$\frac{\Phi_1 \triangleright_{\mathrm{CbV}} \Gamma_1, x : M_1 \vdash^{(m_1,e_1)} t : [O \multimap N] \qquad \Phi_2 \triangleright_{\mathrm{CbV}} \Gamma_2, x : M_2 \vdash^{(m_2,e_2)} V'\langle\!\langle v\rangle\!\rangle : O}{\Gamma, x : M \vdash^{(m,e)} tV'\langle\!\langle v\rangle\!\rangle : N}\ \mathsf{app}$$

  where $\Gamma \coloneqq \Gamma_1 \uplus \Gamma_2$ and $M \coloneqq M_1 \uplus M_2$ and $(m, e) \coloneqq (m_1 + m_2 + 1, e_1 + e_2)$. By *i.h.*, there exist a multi type M', two type contexts Γ_2' and Π_2, and two derivations $\Psi_2 \triangleright_{\mathrm{CbV}} \Pi_2, x : M_2 \uplus M' \vdash^{(m_2'', e_2'')} V'\langle\!\langle x\rangle\!\rangle : O$ and $\Phi' \triangleright_{\mathrm{CbV}} \Gamma' \vdash^{(m', e')} v : M'$ such that $\Gamma_2 = \Gamma' \uplus \Pi_2$ ad $(m_2, e_2) = (m' + m_2'', e' + e_2'' - 1)$. We can then construct the following derivation Ψ

$$\frac{\Phi_1 \triangleright_{\mathrm{CbV}} \Gamma_1, x : M_1 \vdash^{(m_1,e_1)} t : [O \multimap N] \qquad \Psi_2 \triangleright_{\mathrm{CbV}} \Pi_2, x : M_2 \uplus M' \vdash^{(m_2'', e_2'')} V'\langle\!\langle x\rangle\!\rangle : O}{\Pi_2 \uplus \Gamma_1, x : M_2 \uplus M' \uplus M_1 \vdash^{(m_2'' + m_1 + 1, e_2'' + e_1)} V'\langle\!\langle x\rangle\!\rangle t : N}\ \mathsf{app}$$

  where $M_1 \uplus M' \uplus M_2 = M \uplus M'$. If we set $\Pi \coloneqq \Pi_2 \uplus \Gamma_1$ and $(m'', e'') \coloneqq (m_2'' + m_1 + 1, e_2'' + e_1)$, then we have $\Gamma' \uplus \Pi = \Gamma' \uplus \Pi_2 \uplus \Gamma_1 = \Gamma_1 \uplus \Gamma_2 = \Gamma$ and $(m' + m'', e' + e'' - 1) = (m' + m_2'' + m_1 + 1, e' + e_2'' + e_1 - 1) = (m_1 + m_2 + 1, e_1 + e_2) = (m, e)$, as required.
- *Left explicit substitution, i.e.* $V = V'[y \leftarrow t]$: then, $V\langle\!\langle x\rangle\!\rangle = V'\langle\!\langle x\rangle\!\rangle[y \leftarrow t]$ and $V\langle\!\langle v\rangle\!\rangle = V'\langle\!\langle v\rangle\!\rangle[y \leftarrow t]$ where $y \notin \mathsf{fv}(v) \cup \{x\}$. We can suppose without loss of generality that $y \notin \mathsf{fv}(t)$. So, Φ has the form

$$\frac{\Phi_1 \triangleright_{\mathrm{CbV}} \Gamma_1, x : M_1, y : O \vdash^{(m_1,e_1)} V'\langle\!\langle v\rangle\!\rangle : N \qquad \Phi_2 \triangleright_{\mathrm{CbV}} \Gamma_2, x : M_2 \vdash^{(m_2,e_2)} t : O}{\Gamma, x : M \vdash^{(m,e)} V'\langle\!\langle v\rangle\!\rangle[y \leftarrow t] : N}\ \mathsf{ES}$$

  where $\Gamma \coloneqq \Gamma_1 \uplus \Gamma_2$ and $M \coloneqq M_1 \uplus M_2$ and $(m, e) \coloneqq (m_1 + m_2, e_1 + e_2)$. By *i.h.*, there exist a multi type M', two type contexts Γ_1' and Π_1, and two derivations $\Psi_1 \triangleright_{\mathrm{CbV}} \Pi_1, x : M_1 \uplus M', y : O \vdash^{(m_1'', e_1'')} V'\langle\!\langle x\rangle\!\rangle : N$ and $\Phi' \triangleright_{\mathrm{CbV}} \Gamma' \vdash^{(m', e')} v : M'$ such that $\Gamma_1 = \Gamma' \uplus \Pi_1$ ad $(m_1, e_1) = (m' + m_1'', e' + e_1'' - 1)$ (note that $y \notin \mathsf{dom}(\Gamma')$ because of Lemma 5.3.1 (Relevance of the CbV type system), since $y \notin \mathsf{fv}(v)$). We can then construct the following derivation Ψ

$$\frac{\Psi_1 \triangleright_{\mathrm{CbV}} \Pi_1, x : M_1 \uplus M', y : O \vdash^{(m_1'', e_1'')} V'\langle\!\langle x\rangle\!\rangle : N \qquad \Phi_2 \triangleright_{\mathrm{CbV}} \Gamma_2, x : M_2 \vdash^{(m_2,e_2)} t : O}{\Pi_1 \uplus \Gamma_2, x : M_1 \uplus M' \uplus M_2 \vdash^{(m_1'' + m_2, e_1'' + e_2)} V'\langle\!\langle x\rangle\!\rangle t : N}\ \mathsf{ES}$$

where $M_1 \uplus M' \uplus M_2 = M \uplus M'$. If we set $\Pi := \Pi_1 \uplus \Gamma_2$ and $(m'', e'') := (m''_1 + m_2, e''_1 + e_2)$, then we have $\Gamma' \uplus \Pi = \Gamma' \uplus \Pi_1 \uplus \Gamma_2 = \Gamma_1 \uplus \Gamma_2 = \Gamma$ and $(m' + m'', e' + e'' - 1) = (m' + m''_1 + m_2, e' + e''_1 + e_2 - 1) = (m_1 + m_2, e_1 + e_2) = (m, e)$, as required.

- *Right explicit substitution, i.e.* $V = t[y \leftarrow V']$: then, $V\langle\langle x \rangle\rangle = t[y \leftarrow V'\langle\langle x \rangle\rangle]$ and $V\langle\langle v \rangle\rangle = t[y \leftarrow V'\langle\langle v \rangle\rangle]$. So, Φ has the form

$$\frac{\Phi_1 \rhd_{\mathrm{CbV}} \Gamma_1, x : M_1, y : O \vdash^{(m_1, e_1)} V''\langle\langle y \rangle\rangle : N \qquad \Phi_2 \rhd_{\mathrm{CbV}} \Gamma_2, x : M_2 \vdash^{(m_2, e_2)} V'\langle\langle v \rangle\rangle : O}{\Gamma, x : M \vdash^{(m,e)} t[y \leftarrow V'\langle\langle v \rangle\rangle] : N}\ \mathsf{ES}$$

where $\Gamma := \Gamma_1 \uplus \Gamma_2$ and $M := M_1 \uplus M_2$ and $(m, e) := (m_1 + m_2, e_1 + e_2)$. By *i.h.*, there exist a multi type M', two type contexts Γ'_2 and Π_2, and two derivations $\Psi_2 \rhd_{\mathrm{CbV}} \Pi_2, x : M_2 \uplus M', y : O \vdash^{(m''_2, e''_2)} V'\langle\langle x \rangle\rangle : O$ and $\Phi' \rhd_{\mathrm{CbV}} \Gamma' \vdash^{(m', e')} v : M'$ such that $\Gamma_2 = \Gamma' \uplus \Pi_2$ ad $(m_2, e_2) = (m' + m''_2, e' + e''_2 - 1)$ (note that $y \notin \mathsf{dom}(\Gamma')$ because of Lemma 5.3.1 (Relevance of the CbV type system), since $y \notin \mathsf{fv}(v)$). We can then construct the following derivation Ψ

$$\frac{\Phi_1 \rhd_{\mathrm{CbV}} \Gamma_1, x : M_1, y : O \vdash^{(m_1, e_1)} t : N \qquad \Psi_2 \rhd_{\mathrm{CbV}} \Pi_2, x : M_2 \uplus M' \vdash^{(m''_2, e''_2)} V'\langle\langle x \rangle\rangle : O}{\Pi_2 \uplus \Gamma_1, x : M_2 \uplus M' \uplus M_1 \vdash^{(m''_2 + m_1 + 1, e''_2 + e_1)} t[y \leftarrow V'\langle\langle x \rangle\rangle] : N}\ \mathsf{ES}$$

where $M_1 \uplus M' \uplus M_2 = M \uplus M'$. If we set $\Pi := \Pi_2 \uplus \Gamma_1$ and $(m'', e'') := (m''_2 + m_1, e''_2 + e_1)$, then we have $\Gamma' \uplus \Pi = \Gamma' \uplus \Pi_2 \uplus \Gamma_1 = \Gamma_1 \uplus \Gamma_2 = \Gamma$ and $(m' + m'', e' + e'' - 1) = (m' + m''_2 + m_1, e' + e''_2 + e_1 - 1) = (m_1 + m_2, e_1 + e_2) = (m, e)$, as required. □

□

(Click here to go back to main chapter.)

The following is required to apply Lemma 5.3.7 (Linear Removal for CbV) in the proof of Proposition 5.3.8.2 (Quantitative Subject Expansion for CbV - exponential case) to obtain the right indices.

Lemma 13.2.20 (Merging multi types of CbV type derivations)**.**

Let $v \in \mathsf{Val}$*. For any two type derivations*

$$\Phi_N \rhd_{\mathrm{CbV}} \Gamma_N \vdash^{(m_N, e_N)} v : N$$
$$\Phi_O \rhd_{\mathrm{CbV}} \Gamma_O \vdash^{(m_O, e_O)} v : O$$

there exsits type derivation

$$\Phi_{N \uplus O} \rhd_{\mathrm{CbV}} \Gamma_N \biguplus \Gamma_O \vdash^{(m_N + m_{m''}, e_N + e_O)} v : N$$

Proof.

Trivial, given that the many typing rule is the only one that can derive a multi type on the right—*i.e.*, the derived type of the subject, in this case M—for values.

□

Proposition 13.2.21 (Quantitative Subject Expansion for CbV)**.**

Let $\Phi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} u : M$ *be a type derivation.*

1. Multiplicative*: If* $t \to_{\mathsf{m}_{\mathrm{CbV}}} u$*, then there exists a derivation*

$$\Psi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m+1, e)} t : M$$

2. Exponential*: If $t \to_{\mathsf{e}_{\mathrm{CbV}}} u$, then there exists a derivation*

$$\Psi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m,e+1)} t : M$$

Proof. (Click here to go back to main chapter.)

By induction on the reduction relation $\to_{\mathrm{CbV}}$, with the root rules $\mapsto_{\mathsf{m}}$ and $\mapsto_{\mathsf{e}_{\mathrm{CbV}}}$ as the base case, and the closure by CbV contexts of $\mapsto_{\mathrm{CbV}} := \mapsto_{\mathsf{m}} \cup \mapsto_{\mathsf{e}_{\mathrm{CbV}}}$ as the inductive one.

- *Root step for $\to_{\mathsf{m}_{\mathrm{CbV}}}$ i.e. $t := S\langle\lambda x.s\rangle m \mapsto_{\mathsf{m}} S\langle s[x\leftarrow m]\rangle =: t'$ where $S := [y_1\leftarrow u_1]\dots[y_n\leftarrow u_n]$* for some $n \geq 0$. We proceed by induction on $n \in \mathbb{N}$.
  If $n = 0$ then $S = \langle\cdot\rangle$ and so $t = S\langle\lambda x.s\rangle m = (\lambda x.s)m$ and $t' = S\langle s[x\leftarrow m]\rangle = s[x\leftarrow m]$. Hence, Φ' has the form

$$\frac{\Psi \rhd_{\mathrm{CbV}} \Pi, x:O \vdash^{(m',e')} s : M \qquad \Theta \rhd_{\mathrm{CbV}} \Delta \vdash^{(m'',e'')} m : O}{\Gamma \vdash^{(m'+m'',e'+e'')} s[x\leftarrow m] : M}\ \mathsf{ES}$$

  where $\Gamma := \Pi \uplus \Delta$ and $m := m' + m''$ and $e := e' + e''$. We can then construct the following type derivation Φ:

$$\dfrac{\dfrac{\dfrac{\Psi \rhd_{\mathrm{CbV}} \Pi, x:O \vdash^{(m',e')} s : M}{\Pi \vdash^{(m',e')} \lambda x.s : O \multimap M}\ \mathsf{fun}}{\Pi \vdash^{(m',e')} \lambda x.s : [O \multimap M]}\ \mathsf{many} \qquad \Theta \rhd_{\mathrm{CbV}} \Delta \vdash^{(m'',e'')} m : O}{\Gamma \vdash^{(1+m'+m'',e'+e'')} (\lambda x.s)m : M}\ \mathsf{app}$$

  where $(1 + m' + m'', e' + e'') = (m + 1, e)$.
  Suppose now $n > 0$. Let $S' := [y_1\leftarrow u_1]\dots[y_{n-1}\leftarrow u_{n-1}]$: then, $t = S\langle\lambda x.s\rangle m = S'\langle\lambda x.s\rangle[y_n\leftarrow u_n]m$ and $t' = S\langle s[x\leftarrow m]\rangle = S'\langle s[x\leftarrow m]\rangle[y_n\leftarrow u_n]$. Hence, Φ' has the form

$$\frac{\Psi' \rhd_{\mathrm{CbV}} \Gamma', y_n : N_n \vdash^{(m',e')} S'\langle s[x\leftarrow m]\rangle : M \qquad \Psi_n \rhd_{\mathrm{CbV}} \Gamma_n \vdash^{(m_n,e_n)} u_n : N_n}{\Gamma \vdash^{(m,e)} S'\langle s[x\leftarrow m]\rangle[y_n\leftarrow u_n] : M}\ \mathsf{ES}$$

  where $\Gamma := \Gamma' \uplus \Gamma_n$ and $(m, e) := (m' + m_n, e' + e_n)$. By *i.h.* applied to Ψ' (since $S'\langle\lambda x.s\rangle m \mapsto_{\mathsf{m}} S'\langle s[x\leftarrow m]\rangle$), there exists a derivation with conclusion $\Gamma', y_n : N_n \vdash^{(m'+1,e')} S'\langle\lambda x.s\rangle m : M$, which necessarily has the form (as $y_n \notin \mathsf{dom}(\Gamma'_0)$ by Lemma 5.3.1 (Relevance of the CbV type system), since $y_n \notin \mathsf{fv}(m)$)

$$\frac{\Psi \rhd_{\mathrm{CbV}} \Pi, x:O, y_n : N_n \vdash^{(m'',e'')} S'\langle\lambda x.s\rangle : M \qquad \Theta \rhd_{\mathrm{CbV}} \Gamma'_0 \vdash^{(m'_0,e'_0)} m : O}{\Gamma', y_n : N_n \vdash^{(m',e')} S'\langle\lambda x.s\rangle m : M}\ \mathsf{ES}$$

  where $\Gamma' := \Pi \uplus \Gamma'_0$ and $(m', e') = (m'' + m'_0, e'' + e'_0)$. Therefore, we can construct the following derivation Φ:

$$\dfrac{\dfrac{\Psi \rhd_{\mathrm{CbV}} \Pi, x:O, y_n : N_n \vdash^{(m'',e'')} S'\langle\lambda x.s\rangle : M \qquad \Psi_n \rhd_{\mathrm{CbV}} \Gamma_n \vdash^{(m_n,e_n)} u_n : N_n}{\Pi \uplus \Gamma_n, x:O \vdash^{(m''+m_n,e''+e_n)} S\langle\lambda x.s\rangle : M}\ \mathsf{ES} \qquad \Theta \rhd_{\mathrm{CbV}} \Gamma'_0 \vdash^{(m'_0,e'_0)} m : O}{\Pi \uplus \Gamma_n \uplus \Gamma'_0 \vdash^{(m''+m_n+m'_0+1,e''+e_n+e'_0)} S\langle\lambda x.s\rangle m : M}\ \mathsf{app}$$

  where $\Pi \uplus \Gamma_n \uplus \Gamma'_0 = \Gamma' \uplus \Gamma_n = \Gamma$ and $(m'' + m_n + m'_0 + 1, e'' + e_n + e'_0) = (m' + m_n + 1, e' + e_n + 1) = (m + 1, e)$.

- *Root step for* $\to_{\mathsf{e}_{\mathrm{CbV}}}$ *i.e.* $t \coloneqq V\langle\!\langle x\rangle\!\rangle[x{\leftarrow}S\langle v\rangle] \mapsto_{\mathsf{e}_{\mathrm{CbV}}} S\langle V\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle =: t'$ with $S \coloneqq [y_1{\leftarrow}u_1]\dots[y_n{\leftarrow}u_n]$ for some $n \geq 0$. We proceed by induction on $n \in \mathbb{N}$.
If $n = 0$ then $S = \langle\cdot\rangle$ and so $t = S\langle v\rangle = v$ and $t' = S\langle V\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle = V\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]$. Hence, Φ' has the form

$$\frac{\Psi_0 \rhd_{\mathrm{CbV}} \Gamma_0, x : N \vdash^{(m_0,e_0)} V\langle\!\langle v\rangle\!\rangle : M \qquad \Theta_1 \rhd_{\mathrm{CbV}} \Gamma_1 \vdash^{(m_1,e_1)} v : N}{\Gamma \vdash^{(m,e)} V\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v] : M}\ \mathsf{ES}$$

where $\Gamma \coloneqq \Gamma_0 \uplus \Gamma_1$, and $m \coloneqq m_0 + m_1$ and $e \coloneqq e_0 + e_1$. By Lemma 5.3.7 (Linear Removal for CbV), there are a multi type N', two type contexts Γ'_0 and Π and two derivations $\Theta' \rhd_{\mathrm{CbV}} \Gamma'_0 \vdash^{(m'_0,m'_0)} v : N'$ and $\Psi \rhd_{\mathrm{CbV}} \Pi, x : N \uplus N' \vdash^{(m'',e'')} V\langle\!\langle v\rangle\!\rangle : M$ such that $\Gamma_0 = \Gamma'_0 \uplus \Pi$ and $(m_0, e_0) = (m'_0 + m'', e'_0 + e'' - 1)$. Note that $x \notin \mathsf{dom}(\Gamma'_0)$ by Lemma 5.3.1 (Relevance of the CbV type system), since $x \notin \mathsf{fv}(v)$. By Lemma 13.2.20 (Merging multi type of CbV type derivations), there is a derivation $\Theta \rhd_{\mathrm{CbV}} \Gamma'_0 \uplus \Gamma_1 \vdash^{(m'_0+m_1,e'_0+e_1)} v : N \uplus N'$. We can construct the following derivation Φ:

$$\frac{\Psi \rhd_{\mathrm{CbV}} \Pi, x : N \uplus N' \vdash^{(m'',e'')} V\langle\!\langle x\rangle\!\rangle : M \qquad \Theta \rhd_{\mathrm{CbV}} \Gamma'_0 \uplus \Gamma_1 \vdash^{(m'_0+m_1,e'_0+e_1)} v : N \uplus N'}{\Pi \uplus \Gamma'_0 \uplus \Gamma_1 \vdash^{(m''+m'_0+m_1,e''+e'_0+e_1)} V\langle\!\langle x\rangle\!\rangle[x{\leftarrow}S\langle v\rangle] : M}\ \mathsf{ES}$$

where $\Pi \uplus \Gamma'_0 \uplus \Gamma_1 = \Gamma_0 \uplus \Gamma_1 = \Gamma$ and $(m'' + m'_0 + m_1, e'' + e'_0 + e_1) = (m_0 + m_1, e_0 + 1 + e_1) = (m, e + 1)$.
Suppose now $n > 0$. Let $S' \coloneqq [y_1{\leftarrow}u_1]\dots[y_{n-1}{\leftarrow}u_{n-1}]$: then, $t = S\langle v\rangle = S'\langle v\rangle[y_n{\leftarrow}u_n]$ and $t' = S\langle V\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle = S'\langle V\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle[y_n{\leftarrow}u_n]$. Hence, Φ' has the form

$$\frac{\Psi' \rhd_{\mathrm{CbV}} \Gamma_0, y_n : N_n \vdash^{(m_0,e_0)} S'\langle V\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle : M \qquad \Theta_n \rhd_{\mathrm{CbV}} \Gamma_n \vdash^{(m_n,e_n)} u_n : N_n}{\Gamma \vdash^{(m,e)} S\langle V\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle : M}\ \mathsf{ES}$$

where $\Gamma \coloneqq \Gamma_0 \uplus \Gamma_n$ and $(m, e) = (m_0 + m_n, e_0 + e_n)$. By *i.h.* applied to Ψ' (since $V\langle\!\langle x\rangle\!\rangle[x{\leftarrow}S'\langle v\rangle] \mapsto_{\mathsf{e}_{\mathrm{CbV}}} S'\langle V\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle$), there exists a derivation with conclusion $\Gamma_0, y_n : N_n \vdash^{(m_0,e_0+1)} V\langle\!\langle x\rangle\!\rangle[x{\leftarrow}S'\langle v\rangle] : M$, which necessarily has the form (as $y_n \notin \mathsf{dom}(\Gamma'_0)$ by Lemma 5.3.1 (Relevance of the CbV type system), since $y_n \notin \mathsf{fv}(V\langle\!\langle x\rangle\!\rangle)$)

$$\frac{\Psi \rhd_{\mathrm{CbV}} \Gamma'_0, x : N \vdash^{(m'_0,e'_0)} V\langle\!\langle x\rangle\!\rangle : M \qquad \Psi'' \rhd_{\mathrm{CbV}} y_n : N_n, \Gamma''_0 \vdash^{(m''_0,e''_0)} S'\langle v\rangle : N}{\Gamma_0, y_n : N_n \vdash^{(m_0,e_0+1)} V\langle\!\langle x\rangle\!\rangle[x{\leftarrow}S'\langle v\rangle] : M}\ \mathsf{ES}$$

where $\Gamma_0 = \Gamma'_0 \uplus \Gamma''_0$ and $(m_0, e_0 + 1) = (m'_0 + m''_0, e'_0 + e''_0)$. Therefore, we can construct the following derivation Φ:

$$\dfrac{\Psi \rhd_{\mathrm{CbV}} \Gamma'_0, x : N \vdash^{(m'_0,e'_0)} V\langle\!\langle x\rangle\!\rangle : M \qquad \dfrac{\Psi'' \rhd_{\mathrm{CbV}} y_n : N_n, \Gamma''_0 \vdash^{(m''_0,e''_0)} S'\langle v\rangle : N \qquad \Theta_n \rhd_{\mathrm{CbV}} \Gamma_n \vdash^{(m_n,e_n)} u_n : N_n}{\Gamma''_0 \uplus \Gamma_n \vdash^{(m''_0+m_n,e''_0+e_n)} S\langle v\rangle : N}\ \mathsf{ES}}{\Gamma'_0 \uplus \Gamma''_0 \uplus \Gamma_n \vdash^{(m'_0+m''+m_n,e'_0+e''+e_n)} V\langle\!\langle x\rangle\!\rangle[x{\leftarrow}S\langle v\rangle] : M}\ \mathsf{ES}$$

where $\Gamma'_0 \uplus \Gamma''_0 \uplus \Gamma_n = \Gamma_0 \uplus \Gamma_n = \Gamma$ and $(m'_0 + m''_0 + m_n, e'_0 + e''_0 + e_n) = (m_0 + m_n, e_0 + 1 + e_n) = (m, e + 1)$.
- *Application left, i.e.* $t \coloneqq us \to_{\mathsf{r}} u's =: t'$ with $u \to_{\mathsf{r}} u'$ and $\mathsf{r} \in \{\mathsf{m}_{\mathrm{CbV}}, \mathsf{e}_{\mathrm{CbV}}\}$. So, Φ' has the form

$$\frac{\Psi_1 \rhd_{\mathrm{CbV}} \Gamma_1 \vdash^{(m',e')} u' : [N \multimap M] \qquad \Psi_2 \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m'+m_2+1,e'+e_2)} t' : M}\ \mathsf{app}$$

where $\Gamma \coloneqq \Gamma_1 \uplus \Gamma_2$ and $m \coloneqq m' + m_2 + 1$ and $e \coloneqq e' + e_2$. By *i.h.* applied to Ψ_1, there is a derivation $\Psi \rhd_{\mathrm{CbV}} \Gamma_1 \vdash^{(m_1,e_1)} u : [N \multimap M]$ where

– $m_1 \coloneqq m' + 1$ and $e_1 \coloneqq e'$ if $\mathsf{r} = \mathsf{m}_{\mathrm{CbV}}$,
– $m_1 \coloneqq m_1$ and $e_1 \coloneqq e' + 1$ if $\mathsf{r} = \mathsf{e}_{\mathrm{CbV}}$.

Thus, we have the derivation Φ

$$\dfrac{\Psi \rhd_{\mathrm{CbV}} \Gamma_1 \vdash^{(m_1,e_1)} u : [N \multimap M] \qquad \Psi_2 \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2+1,e_1+e_2)} t : M}\ \mathsf{app}$$

where
– $m_1 + m_2 + 1 = m' + 1 + m_2 + 1 = m + 1$ and $e_1 + e_2 = e' + e_2 = e$ if $\mathsf{r} = \mathsf{m}_{\mathrm{CbV}}$,
– $m_1 + m_2 + 1 = m' + m_2 + 1 = m$ and $e_1 + e_2 = e' + 1 + e_2 = e + 1$ if $\mathsf{r} = \mathsf{e}_{\mathrm{CbV}}$.

- *Application right, i.e.* $t \coloneqq su \to_{\mathsf{r}} su' =: t'$ with $u \to_{\mathsf{r}} u'$ and $\mathsf{r} \in \{\mathsf{m}_{\mathrm{CbV}}, \mathsf{e}_{\mathrm{CbV}}\}$. So, Φ' has the form

$$\dfrac{\Psi_1 \rhd_{\mathrm{CbV}} \Gamma_1 \vdash^{(m_1,e_1)} s : [N \multimap M] \qquad \Psi_2 \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m',e')} u' : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m'+1,e_1+e')} t' : M}\ \mathsf{app}$$

where $\Gamma \coloneqq \Gamma_1 \uplus \Gamma_2$ and $m \coloneqq m_1 + m' + 1$ and $e \coloneqq e_1 + e'$. By *i.h.* applied to d'_2, there is a derivation $\Psi \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} u : N$ where
– $m_2 \coloneqq m' + 1$ and $e_2 \coloneqq e'$ if $\mathsf{r} = \mathsf{m}_{\mathrm{CbV}}$,
– $m_2 \coloneqq m'$ and $e_2 \coloneqq e' + 1$ if $\mathsf{r} = \mathsf{e}_{\mathrm{CbV}}$.

Thus, we have the derivation Φ

$$\dfrac{\Psi_1 \rhd_{\mathrm{CbV}} \Gamma_1 \vdash^{(m_1,e_1)} s : [N \multimap M] \qquad \Psi \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} u : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2+1,e_1+e_2)} t : M}\ \mathsf{app}$$

where
– $m_1 + m_2 + 1 = m_1 + m' + 1 + 1 = m + 1$ and $e_1 + e_2 = e_1 + e' = e$ if $\mathsf{r} = \mathsf{m}_{\mathrm{CbV}}$,
– $m_1 + m_2 + 1 = m_1 + m' + 1 = m$ and $e_1 + e_2 = e_1 + e' + 1 = e + 11$ if $\mathsf{r} = \mathsf{e}_{\mathrm{CbV}}$.

- *Left explicit substitution, i.e.* $t \coloneqq u[x \leftarrow s] \to_{\mathsf{r}} u'[x \leftarrow s] =: t'$ with $u \to_{\mathsf{r}} u'$ and $\mathsf{r} \in \{\mathsf{m}_{\mathrm{CbV}}, \mathsf{e}_{\mathrm{CbV}}\}$. So, Φ' has the form

$$\dfrac{\Psi_1 \rhd_{\mathrm{CbV}} \Gamma_1, x : N \vdash^{(m',e')} u' : M \qquad \Psi_2 \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m'+m_2,e'+e_2)} t' : M}\ \mathsf{ES}$$

where $\Gamma \coloneqq \Gamma_1 \uplus \Gamma_2$ and $m \coloneqq m' + m_2$ and $e \coloneqq e' + e_2$. By *i.h.* applied to Ψ_1, there is a derivation $\Psi \rhd_{\mathrm{CbV}} \Gamma_1, x : N \vdash^{(m_1,e_1)} u : M$ where
– $m_1 \coloneqq m' + 1$ and $e_1 \coloneqq e'$ if $\mathsf{r} = \mathsf{m}_{\mathrm{CbV}}$,
– $m_1 \coloneqq m'$ and $e_1 \coloneqq e' + 1$ if $\mathsf{r} = \mathsf{e}_{\mathrm{CbV}}$.

Thus, we have the derivation Φ

$$\dfrac{\Psi \rhd_{\mathrm{CbV}} \Gamma_1, x : N \vdash^{(m_1,e_1)} u : M \qquad \Psi_2 \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,e_1+e_2)} t : M}\ \mathsf{ES}$$

where
– $m_1 + m_2 = m' + 1 + m_2 = m + 1$ and $e_1 + e_2 = e' + e_2 = e$ if $\mathsf{r} = \mathsf{m}_{\mathrm{CbV}}$,
– $m_1 + m_2 = m' + m_2 = m$ and $e_1 + e_2 = e' + 1 + e_2 = e + 1$ if $\mathsf{r} = \mathsf{e}_{\mathrm{CbV}}$.

- *Right explicit substitution, i.e.* $t \coloneqq u[x \leftarrow s] \to_{\mathsf{r}} u[x \leftarrow s'] =: t'$ with $s \to_{\mathsf{r}} s'$ and $\mathsf{r} \in \{\mathsf{m}_{\mathrm{CbV}}, \mathsf{e}_{\mathrm{CbV}}\}$. So, Φ' has the form

$$\dfrac{\Psi_1 \rhd_{\mathrm{CbV}} \Gamma_1, x : N \vdash^{(m_1,e_1)} u : M \qquad \Psi_2 \rhd_{\mathrm{CbV}} \Gamma_2 \vdash^{(m',e')} s' : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m',e_1+e')} t' : M}\ \mathsf{ES}$$

where $\Gamma \coloneqq \Gamma_1 \uplus \Gamma_2$ and $m \coloneqq m_1 + m'$ and $e \coloneqq e_1 + e'$. By *i.h.* applied to Ψ_2, there is a derivation $\Psi \triangleright_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} s : M$ where
- $m_2 \coloneqq m' + 1$ and $e_2 \coloneqq e'$ if $\mathsf{r} = \mathsf{m}_{\mathrm{CbV}}$,
- $m_2 \coloneqq m'$ and $e_2 \coloneqq e' + 1$ if $\mathsf{r} = \mathsf{e}_{\mathrm{CbV}}$.

Thus, we have the derivation Φ

$$\frac{\Psi_1 \triangleright_{\mathrm{CbV}} \Gamma_1, x : N \vdash^{(m_1,e_1)} u : M \qquad \Psi \triangleright_{\mathrm{CbV}} \Gamma_2 \vdash^{(m_2,e_2)} s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,e_1+e_2)} t : M}\ \mathsf{ES}$$

where
- $m_1 + m_2 = m_1 + m' + 1 = m + 1$ and $e_1 + e_2 = e_1 + e' = e$ if $\mathsf{r} = \mathsf{m}_{\mathrm{CbV}}$,
- $m_1 + m_2 = m_1 + m' = m$ and $e_1 + e_2 = e_1 + e' + 1 = e + 1$ if $\mathsf{r} = \mathsf{e}_{\mathrm{CbV}}$.

□

(Click here to go back to main chapter.)

Theorem 13.2.22 (Tight Completeness for CbV)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be closed. If there exists $d : t \to^*_{\mathrm{CbV}} u$ for some u in $\to_{\mathrm{CbV}}$-normal form, then there exists a type derivation $\Phi \triangleright_{\mathrm{CbV}} \emptyset \vdash^{(|d|_{\mathsf{m}},|d|_{\mathsf{e}})} t : \mathbf{0}$.

Proof. *(Click here to go back to main chapter.)*
By induction on the length $|d|$ of the reduction sequence $d\colon t \to^*_{\mathrm{CbV}} u$:
- *Base case*: Let $k \coloneqq 0$. Then $t = u$ and t is in $\to_{\mathrm{CbV}}$-normal form. By Proposition 4.6.1.1 (Syntactic characaterization of closed normal forms - CbV), we have that $\mathsf{norm}_{\mathrm{CbV}}(t)$. In addition, Proposition 5.3.6 (Tight typability of CbV-normal forms) yields tight type derivation $\Phi \triangleright_{\mathrm{CbV}} \emptyset \vdash^{(0,0)} t : \mathbf{0}$, which satisfies the statement—in particular, because $|d|_{\mathsf{m}} = |d|_{\mathsf{e}} = 0$.
- *Inductive case*: Let $k > 0$; *i.e.*, $t \to_{\mathrm{CbV}} s \to^{k-1}_{\mathrm{CbV}} u$. Let d' be the evaluation $s \to^{k-1}_{\mathrm{CbV}} u$. By *i.h.*, there exists tight type derivation

$$\Psi \triangleright_{\mathrm{CbV}} \emptyset \vdash^{(|d'|_{\mathsf{m}},|d'|_{\mathsf{e}})} s : \mathbf{0}$$

Case analysis on the kind of reduction step in $t \to_{\mathrm{CbV}} s$:
- *Multiplicative step*: Let $t \to_{\mathsf{m}_{\mathrm{CbV}}} s$. By Proposition 5.3.8.1 (Quantitative Subject Expansion for CbV - Multiplicative), there exists (tight) type derivation

$$\Psi' \triangleright_{\mathrm{CbV}} \emptyset \vdash^{(|d'|_{\mathsf{m}}+1,|d'|_{\mathsf{e}})} s : \mathbf{0}$$

  which satisfies the statement—in particular, because $|d|_{\mathsf{m}} = |d'|_{\mathsf{m}} + 1$ and $|d|_{\mathsf{e}} = |d'|_{\mathsf{e}}$.
- *Exponential step*: Let $t \to_{\mathsf{e}_{\mathrm{CbV}}} s$. By Proposition 5.3.8.2 (Quantitative Subject Expansion for CbV - Exponential), there exists (tight) type derivation

$$\Psi' \triangleright_{\mathrm{CbV}} \emptyset \vdash^{(|d'|_{\mathsf{m}},|d'|_{\mathsf{e}}+1)} s : \mathbf{0}$$

  which satisfies the statement—in particular, because $|d|_{\mathsf{m}} = |d'|_{\mathsf{m}}$ and $|d|_{\mathsf{e}} = |d'|_{\mathsf{e}} + 1$.

□

(Click here to go back to main chapter.)

13.2.5 CbNeed correctness

As for the CbN and CbV cases, the CbNeed multi type system satisfying the following

Lemma 13.2.23 (Relevance of the CbNeed type system)**.**
Let $t \in \Lambda_{\mathsf{L}}$ and let $\Phi \triangleright_{CbNeed} \Gamma \vdash^{(m,e)} t : L$ be a type derivation. If $x \notin \mathsf{fv}(t)$ then $x \notin \mathsf{dom}(\Gamma)$.

Proof. (Click here to go back to main chapter.)
By structural induction on $\Phi \triangleright_{\text{CbNeed}} \Gamma \vdash^{(m,e)} t : M$ and proceeding by case analysis on the last derivation rule of Φ. The cases of rules ax, norm, many and app are all trivial; the case of rule app_{gc} follows easily from the *i.h.*. Let us consider only the ones appending ESs:

- Let Φ be derived as follows:

$$\frac{\Psi \triangleright_{\text{CbNeed}} \Gamma \vdash^{(m,e)} u : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e)} u[x \leftarrow s] : L}\ \text{ES}_{\text{gc}}$$

  where $t = u[x \leftarrow s]$. By *i.h.* on Ψ, $\mathsf{dom}(\Gamma) \subseteq \mathsf{fv}(u)$. Then, if $x \notin \mathsf{fv}(u)$ we have that $\mathsf{dom}(\Gamma) = \mathsf{fv}(u) = \mathsf{fv}(u[x \leftarrow s])$, and if $x \in \mathsf{fv}(u)$ then $\mathsf{dom}(\Gamma) = \mathsf{dom}(\Gamma) \setminus \{x\} = \mathsf{fv}(u) \setminus \{x\} \subseteq (\mathsf{fv}(u) \setminus \{x\}) \cup \mathsf{fv}(s) = \mathsf{fv}(u[x \leftarrow s])$.
- Let Φ be derived as follows:

$$\frac{\Psi \triangleright_{\text{CbNeed}} \Pi; x : N \vdash^{(m',e')} u : M \quad \Theta \triangleright_{\text{CbNeed}} \Delta \vdash^{(m'',e'')} s : N}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'')} u[x \leftarrow s] : M}\ \text{ES}$$

  where $t = u[x \leftarrow s]$ and $\Gamma = \Pi \biguplus \Delta$. By *i.h.* on Ψ, $\mathsf{dom}(\Pi; x : M) = \mathsf{dom}(\Gamma) \cup \{x\} \subseteq \mathsf{fv}(u)$, and by *i.h.* on Θ, $\mathsf{dom}(\Delta) \subseteq \mathsf{fv}(s)$. Hence, $\mathsf{dom}(\Gamma) = \mathsf{dom}(\Pi \biguplus \Delta) \subseteq (\mathsf{fv}(u) \setminus \{x\}) \cup \mathsf{fv}(s) = \mathsf{fv}(u[x \leftarrow s])$.

□

(Click here to go back to main chapter.)
Besides Lemma 5.4.1 (Relevance of the CbNeed type system), the CbNeed type system also requires the following property ensuring that the last typing rule applied in a type derivation of a *hereditary* reduction step—*i.e.*, Λ_{L}-terms of the form $E\langle\!\langle x \rangle\!\rangle[x \leftarrow E'\langle t \rangle] \in \Lambda_{\mathsf{L}}$—can only be ES.

Lemma 13.2.24 (Plugged variables and domain of type contexts)**.**
Let $x \in \mathsf{Var}$, E be a CbNeed evaluation context, and let $\Phi \triangleright_{CbNeed} \Gamma \vdash^{(m,e)} E\langle\!\langle x \rangle\!\rangle : M$ be a type derivation such that $M \neq \mathbf{0}$.
Then $\Gamma(x) \neq \mathbf{0}$.

Proof.
By induction on the construction of E:

- Let $E = \langle\cdot\rangle$. Then $t = x$ and so Φ is of the form

$$\frac{}{x : M \vdash^{(m,e)} x : M}\ \mathsf{ax}$$

  Clearly, if $M \neq \mathbf{0}$ then $x \in \mathsf{dom}(\Gamma)$.
- If $E = E_1 u$, then $t = E_1\langle\!\langle x \rangle\!\rangle u$. Then Φ can only have either app or app_{gc} as the last typing rule.

– Let Φ be of the form (with $\Gamma = \Pi \uplus \Delta$ and $N \neq \mathbf{0}$)

$$\dfrac{\begin{array}{cc}\vdots\ \Phi_{E_1\langle\langle x\rangle\rangle} & \\ \Pi \vdash^{(m',e')} E_1\langle\langle x\rangle\rangle : [N \multimap M] & \Delta \vdash^{(m'',e'')} u : N \end{array}}{\Pi \uplus \Delta \vdash^{(m'+m''+1,e'+e'')} E_1\langle\langle x\rangle\rangle u : M}\ \mathsf{app}$$

Then by application of *i.h.* on $\Phi_{E_1\langle\langle x\rangle\rangle}$ we obtain that $x \in \mathsf{dom}(\Pi)$, and hence $x \in \mathsf{dom}(\Gamma)$.

– If Φ has $\mathsf{app}_{\mathrm{gc}}$ as its last rule instead, then $x \in \mathsf{dom}(\Gamma)$ simply by *i.h.*.

- If $E = E_1[y{\leftarrow}u]$, then $t = E\langle\langle x\rangle\rangle = E_1\langle\langle x\rangle\rangle[y{\leftarrow}u]$, with $x \neq y$. Then Φ can only have either ES or $\mathsf{ES}_{\mathrm{gc}}$ as the last typing rule.

– Let Φ be of the form (with $\Gamma = \Pi \uplus \Delta$ and $N \neq \mathbf{0}$)

$$\dfrac{\begin{array}{cc}\vdots\ \Phi_{E_1\langle\langle x\rangle\rangle} & \\ \Pi, y : N \vdash^{(m',e')} E_1\langle\langle x\rangle\rangle : M & \Delta \vdash^{(m'',e'')} u : N \end{array}}{\Pi \uplus \Delta \vdash^{(m'+e'',e'+e'')} E_1\langle\langle x\rangle\rangle[y{\leftarrow}u] : M}\ \mathsf{ES}$$

Then by application of *i.h.* on $\Phi_{E_1\langle\langle x\rangle\rangle}$ we obtain that $x \in \mathsf{dom}(\Pi, y : N)$, which implies that $x \in \mathsf{dom}(\Pi)$ (since $y \neq x$) and so $x \in \mathsf{dom}(\Gamma)$.

– If Φ has $\mathsf{ES}_{\mathrm{gc}}$ as its last rule instead, then $x \in \mathsf{dom}(\Gamma)$ simply by *i.h.*.

- Let $E = E_1\langle\langle y\rangle\rangle[y{\leftarrow}E_2]$, and so $t = E\langle\langle x\rangle\rangle = E_1\langle\langle y\rangle\rangle[y{\leftarrow}E_2\langle\langle x\rangle\rangle]$, with $x \neq y$ because x is a free variable of t while y is a bound variable of t, and we are working up to α-equivalence. Suppose now that $\mathsf{ES}_{\mathrm{gc}}$ was the last typing rule of Φ. This means that Φ is of the form (with $\Gamma(y) = \mathbf{0}$)

$$\dfrac{\begin{array}{c}\vdots\ \Phi_{E_1\langle\langle y\rangle\rangle} \\ \Gamma \vdash^{(m,e)} E_1\langle\langle y\rangle\rangle : M \end{array}}{\Gamma \vdash^{(m,e)} E_1\langle\langle y\rangle\rangle[y{\leftarrow}E_2\langle\langle x\rangle\rangle] : M}\ \mathsf{ES}_{\mathrm{gc}}$$

However, by applying *i.h.* on $\Phi_{E_1\langle\langle y\rangle\rangle}$ we obtain that $y \in \mathsf{dom}(\Gamma)$, which is in contradiction with the constraint $\Gamma(y) = \mathbf{0}$ of rule $\mathsf{ES}_{\mathrm{gc}}$.

Hence, Φ can only have ES as the last typing rule. Thus, Φ is of the form

$$\dfrac{\begin{array}{cc} \vdots & \vdots\ \Phi_{E_2\langle\langle x\rangle\rangle} \\ \Pi, y : N \vdash^{(m',e')} E_1\langle\langle y\rangle\rangle : M & \Delta \vdash^{(m'',e'')} E_2\langle\langle x\rangle\rangle : N \end{array}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'')} E_1\langle\langle y\rangle\rangle[y{\leftarrow}E_2\langle\langle x\rangle\rangle] : M}\ \mathsf{ES}$$

with $\Gamma = \Pi \uplus \Delta$ and $N \neq \mathbf{0}$. We can then apply *i.h.* on $\Phi_{E_2\langle\langle x\rangle\rangle}$ to obtain that $x \in \mathsf{dom}(\Delta)$ and so $x \in \mathsf{dom}(\Gamma)$.

□

Thus, Lemma 13.2.24 (Plugged variables and domain of type contexts) ensures that the following application of typing rule $\mathrm{ES}_{\mathrm{gc}}$ is impossible—provided that $M \neq \mathbf{0}$:

$$\dfrac{\Phi \rhd_{\mathrm{CbNeed}} \Gamma \vdash^{(m,e)} E\langle\langle x\rangle\rangle : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e)} E\langle\langle x\rangle\rangle[x{\leftarrow}E'\langle t\rangle] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

Hence, typing $E\langle\!\langle x\rangle\!\rangle[x{\leftarrow}E'\langle t\rangle]$ requires an application of typing rule ES, as follows:

$$\frac{\Phi \rhd_{\text{CbNeed}} \Gamma; x : N \vdash^{(m,e)} E\langle\!\langle x\rangle\!\rangle : M \quad \Psi \rhd_{\text{CbNeed}} \Pi \vdash^{(m',e')} E'\langle t\rangle : N \quad N = \mathbf{0}}{\Gamma \biguplus \Pi \vdash^{(m+m',e+e')} E\langle\!\langle x\rangle\!\rangle[x{\leftarrow}E'\langle t\rangle] : M}\ \text{ES}$$

Thus, this property helps us in ensuring that the reduction quantitities corresponding to $E'\langle t\rangle$ are included in a type derivation for $E''\langle E\langle\!\langle x\rangle\!\rangle[x{\leftarrow}E'\langle t\rangle]\rangle$[1].

Proposition 13.2.25 (Typing properties of CbNeed-normal forms)**.**

Let $t \in \Lambda_{\mathsf{L}}$ be such that $\mathsf{norm}(t)$, and $\Phi \rhd_{CbNeed} \Gamma \vdash^{(m,e)} t : [\mathsf{norm}]$ be a type derivation. Then $\Gamma = \emptyset$ and $(m, e) = (0, 0)$.

Proof. (Click here to go back to main chapter.)

By Proposition 4.6.1.1 (Syntactic characterization of closed normal forms - CbNeed), $\mathsf{norm}(t)$. We proceed by case analysis on the derivation of $\mathsf{norm}(t)$:

- *Base case*: Let $t = \lambda x.u$, with $\mathsf{norm}(\lambda x.u)$. Thus, Φ can only be derived as follows

$$\frac{\dfrac{}{\emptyset \vdash^{(0,0)} \lambda x.u : \mathsf{norm}}\ \mathsf{norm}}{\emptyset \vdash^{(0,0)} \lambda x.u : [\mathsf{norm}]}\ \mathsf{many}$$

  which satisfies the statement.
- *Inductive case*: Let $\mathsf{norm}(t)$ be derived as follows

$$\frac{\mathsf{norm}(u)}{\mathsf{norm}(u[x{\leftarrow}s])}$$

  with $t = u[x{\leftarrow}s]$. We proceed by case analysis on the last typing rule in Φ:
  - Suppose Φ were of the following form

$$\frac{\Psi \rhd_{\text{CbNeed}} \Pi; x : M \vdash^{(m',e')} u : \mathsf{norm} \quad \Theta \rhd_{\text{CbNeed}} \Delta \vdash^{(m'',e'')} s : M \quad M \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m+m',e+e')} u[x{\leftarrow}s] : \mathsf{norm}}\ \text{ES}$$

    with $\Gamma = \Pi \uplus \Delta$ and $(m, e) = (m + m', e + e')$. However, application of the *i.h.* on Ψ would give that $M = \mathbf{0}$; absurd.
  - Let Φ be of the following form

$$\frac{\Psi \rhd_{\text{CbNeed}} \Gamma \vdash^{(m',e')} u : \mathsf{norm} \quad \Gamma(x) = \mathbf{0}}{\Pi \vdash^{(m,e)} u[x{\leftarrow}s] : \mathsf{norm}}\ \text{ES}_{\text{gc}}$$

    By *i.h.* on Ψ, $\Gamma = \mathbf{0}$ and $(m', e') = (0, 0)$.

□

(Click here to go back to main chapter.)

[1] In Chapter 7 (Multi types for Open CbNeed) and Chapter 9 (Multi types for Useful Open CbNeed), where we study multi type systems for different extensions to (weak and closed) CbNeed, we extend Lemma 13.2.24 (Plugged variables and domain of type contexts) to encompass the typing properties required for said extensions to CbNeed.

Lemma 13.2.26 (Linear Substitution for CbNeed)**.**

Let $\Phi \rhd_{CbNeed} \Gamma; x : M \vdash^{(m,e)} E\langle\langle x\rangle\rangle : N$ be a type derivation and let $v \in \mathsf{Val}$. Then $e \geq 1$ and there exists a splitting $M = O \uplus P$ such that for every derivation $\Psi \rhd_{\mathrm{CbV}} \Pi \vdash^{(m',e')} v : O$ there is a derivation

$$\Theta \rhd_{\mathrm{CbV}} \left(\Gamma \biguplus \Pi\right); x : P \vdash^{(m+m',e+e'-1)} V\langle\langle v\rangle\rangle : N$$

Proof. (Click here to go back to main chapter.)

We prove this by induction on E:

- *Empty context,* i.e. $E = \langle\cdot\rangle$. Then $\Gamma = \emptyset$, $O = M$, and $\Phi_{E\langle\langle x\rangle\rangle}$ is of the form

$$\frac{}{x\colon M \vdash^{(0,1)} x : M}\ \mathsf{ax}$$

  Therefore, by defining $M_1 \coloneqq M$ and $M_2 \coloneqq \mathbf{0}$, the statement holds for every $\Psi \rhd_{\mathrm{CbNeed}} \Pi \vdash^{(m',e')} v : M_1$ by taking $\Phi_{E\langle\langle v\rangle\rangle} \coloneqq \Psi$, since $E\langle\langle v\rangle\rangle = v$. In particular, note that $(m + m', e + e' - 1) = (0 + m', 1 + e' - 1) = (m', e')$.
- *Left of an application,* i.e. $E = E_1 u$. There are two possible last rules in $\Phi_{E\langle\langle x\rangle\rangle}$, namely app or $\mathsf{app}_{\mathrm{gc}}$.
  - Let $\Phi_{E\langle\langle x\rangle\rangle}$ be of the form

$$\frac{x\colon M_\Delta; \Delta \vdash^{(m_\Delta,e_\Delta)} N_1\langle\langle x\rangle\rangle : [O' \multimap O] \quad x\colon M_\Sigma; \Sigma \vdash^{(m_\Sigma,e_\Sigma)} u : O'}{x\colon (M_\Delta \uplus M_\Sigma); (\Delta \uplus \Sigma) \vdash^{(m_\Delta+m_\Sigma+1,e_\Delta+e_\Sigma)} E_1\langle\langle x\rangle\rangle u : O}\ \mathsf{app}$$

    where $\Gamma = \Delta \uplus \Sigma$, $\Delta(x) = \Sigma(x) = \mathbf{0}$ and $M = M_\Delta \uplus M_\Sigma$.
    By applying the *i.h.* on the left-hand side premise we obtain that there exists a splitting $M_\Delta = M_{\Delta,1} \uplus M_{\Delta,2}$ such that for every derivation $\Psi \rhd_{\mathrm{CbNeed}} \Pi \vdash^{(m',e')} v : M_{\Delta,1}$ there exists a derivation $\Phi_{E_1\langle\langle v\rangle\rangle} \rhd_{\mathrm{CbNeed}} x\colon M_{\Delta,2}; \Delta \uplus \Pi \vdash^{(m_\Delta+m',e_\Delta+e'-1)} E_1\langle\langle v\rangle\rangle : [O' \multimap O]$. We can then construct $\Phi_{E\langle\langle v\rangle\rangle}$ for such a Ψ as follows

$$\frac{x\colon M_{\Delta,2}; \Delta \uplus \Pi \vdash^{(m_\Delta+m',e_\Delta+e'-1)} E_1\langle\langle v\rangle\rangle : [O' \multimap O] \quad x\colon M_\Sigma; \Sigma \vdash^{(m_\Sigma,e_\Sigma)} u : O'}{x\colon M_{\Delta,2} \uplus M_\Sigma; \Delta \uplus \Pi \uplus \Sigma \vdash^{(m_\Delta+m'+m_\Sigma+1,e_\Delta+e'-1+e_\Sigma)} E_1\langle\langle v\rangle\rangle u : O}\ \mathsf{app}$$

    Note that $\Phi_{E\langle\langle v\rangle\rangle}$ is as desired by splitting M into $M_1 \coloneqq M_{\Delta,1}$ and $M_2 \coloneqq M_{\Delta,2} \uplus M_\Sigma$.
  - Let $\Phi_{E\langle\langle x\rangle\rangle}$ be of the form

$$\frac{\Phi_{E_1\langle\langle x\rangle\rangle} \rhd_{\mathrm{CbNeed}} x\colon M; \Gamma \vdash^{(m-1,e)} E_1\langle\langle x\rangle\rangle : [\mathbf{0} \multimap O]}{x\colon M; \Gamma \vdash^{(m,e)} E_1\langle\langle x\rangle\rangle u : O}\ \mathsf{app}_{\mathrm{gc}}$$

    By applying the *i.h.* on $\Phi_{E_1\langle\langle x\rangle\rangle}$ we obtain that there exists a splitting $M = M_1 \uplus M_2$ such that for every derivation $\Psi \rhd_{\mathrm{CbNeed}} \Pi \vdash^{(m',e')} v : M_1$ there is a derivation $\Phi_{E_1\langle\langle v\rangle\rangle} \rhd_{\mathrm{CbNeed}} x\colon M_2; \Gamma \uplus \Pi \vdash^{(m-1+m',e+e'-1)} E_1\langle\langle v\rangle\rangle : [\mathbf{0} \multimap O]$. We can then construct $\Phi_{E\langle\langle v\rangle\rangle}$ for such a Ψ as follows

$$\frac{x\colon M_2; \Gamma \uplus \Pi \vdash^{(m-1+m',e+e'-1)} E_1\langle\langle v\rangle\rangle : [\mathbf{0} \multimap O]}{x\colon M_2; \Gamma \uplus \Pi \vdash^{(m+m',e+e'-1)} E_1\langle\langle v\rangle\rangle u : O}\ \mathsf{app}_{\mathrm{gc}}$$

- *Left of a substitution;* i.e. $E = E_1[y \leftarrow u]$. Note that $x \neq y$, because the hypothesis $E\langle\langle x\rangle\rangle$ implies that E does not capture x. There are two possible last rules in $\Phi_{E\langle\langle x\rangle\rangle}$, namely ES and $\mathsf{ES}_{\mathrm{gc}}$.

– Let $\Phi_{E\langle\!\langle x\rangle\!\rangle}$ be of the form

$$\frac{x\colon M_\Delta;\Delta \vdash^{(m_\Delta,e_\Delta)} E_1\langle\!\langle x\rangle\!\rangle : O \quad x\colon M_\Sigma;\Sigma \vdash^{(m_\Sigma,e_\Sigma)} u : \Delta(y) \quad \Delta(y) \neq \mathbf{0}}{x\colon (M_\Delta \uplus M_\Sigma);(\Delta \setminus\!\setminus y) \uplus \Sigma \vdash^{(m_\Delta+m_\Sigma,e_\Delta+e_\Sigma)} E_1\langle\!\langle x\rangle\!\rangle[y{\leftarrow}u] : O}\ \mathsf{ES}$$

where $M = M_\Delta \uplus M_\Sigma$ and $\Gamma = (\Delta \setminus\!\setminus y) \uplus \Sigma$.
By applying the *i.h.* on the leftmost premise we obtain a splitting $M_\Delta = M_{\Delta,1} \uplus M_{\Delta,2}$ such that for every derivation $\Psi \rhd_{\mathrm{CbNeed}} \Pi \vdash^{(m',e')} v : M_{\Delta,1}$ there exists a derivation $\Phi_{E_1\langle\!\langle v\rangle\!\rangle} \rhd_{\mathrm{CbNeed}} x\colon M_{\Delta,2};\Delta \uplus \Pi \vdash^{(m_\Delta+m',e_\Delta+e'-1)} E_1\langle\!\langle v\rangle\!\rangle : O$. Note however that if $y \in \mathsf{dom}(\Pi)$, then Lemma 5.4.1 (Relevance of the CbNeed type system) applied on Ψ would imply that $y \in \mathsf{fv}(v)$, which contradicts the hypothesis that E does not capture the free variables of v; *i.e.*, $y \notin \mathsf{dom}(\Pi)$, and so $(\Pi \uplus \Delta)(y) = \Delta(y)$. We can then construct $\Phi_{E\langle\!\langle v\rangle\!\rangle}$ for such a Ψ as follows

$$\frac{x\colon M_{\Delta,2};\Delta \uplus \Pi \vdash^{(m_\Delta+m',e_\Delta+e'-1)} E_1\langle\!\langle v\rangle\!\rangle : O \quad x\colon M_\Sigma;\Sigma \vdash^{(m_\Sigma,e_\Sigma)} u : \Delta(y) \quad \Delta(y) \neq \mathbf{0}}{x\colon M_{\Delta,2} \uplus M_\Sigma;((\Delta \uplus \Pi) \setminus\!\setminus y) \uplus \Sigma \vdash^{(m_\Delta+m'+m_\Sigma,e_\Delta+e'-1+e_\Sigma)} E_1\langle\!\langle v\rangle\!\rangle[y{\leftarrow}u] : O}\ \mathsf{ES}$$

by splitting M into $M_1 \coloneqq M_{\Delta,1}$ and $M_2 \coloneqq M_{\Delta,2} \uplus M_\Sigma$. Since $y \notin \mathsf{dom}(\Pi)$, then $((\Delta \uplus \Pi) \setminus\!\setminus y) \uplus \Sigma = (\Delta \setminus\!\setminus y) \uplus \Pi \uplus \Sigma = \Gamma \uplus \Pi$.
– Let $\Phi_{E\langle\!\langle x\rangle\!\rangle}$ be of the form

$$\frac{x\colon M;\Gamma \vdash^{(m,e)} E_1\langle\!\langle x\rangle\!\rangle : O \quad \Gamma(y) = \mathbf{0}}{x\colon M;\Gamma \vdash^{(m,e)} E_1\langle\!\langle x\rangle\!\rangle[y{\leftarrow}u] : O}\ \mathrm{ES}_{\mathrm{gc}}$$

By applying *i.h.* on the premise we obtain a splitting $M = M_1 \uplus M_2$ such that for every derivation $\Psi \rhd_{\mathrm{CbNeed}} \Pi \vdash^{(m',e')} v : M_1$ there exists a derivation

$$\Phi_{E_1\langle\!\langle v\rangle\!\rangle} \rhd_{\mathrm{CbNeed}} x\colon M_2;\Gamma \uplus \Pi \vdash^{(m+m',e+e'-1)} E_1\langle\!\langle v\rangle\!\rangle : O$$

Note that $y \notin \mathsf{dom}(\Pi)$, because applying Lemma 5.4.1 (Relevance of the CbNeed type system) on Ψ would otherwise imply that $y \in \mathsf{fv}(v)$, which contradicts the hypothesis that E does not capture the free variables of v. Hence, we can then construct $\Phi_{E\langle\!\langle v\rangle\!\rangle}$ for such a Ψ as follows

$$\frac{x\colon M_2;\Gamma \uplus \Pi \vdash^{(m+m',e+e'-1)} E\langle\!\langle v\rangle\!\rangle : O \quad (\Gamma \uplus \Pi)(y) = \Gamma(y) = \mathbf{0}}{x\colon M_2;\Gamma \uplus \Pi \vdash^{(m+m',e+e'-1)} E_1\langle\!\langle v\rangle\!\rangle[y{\leftarrow}u] : O}\ \mathsf{ES}_{\mathrm{gc}}$$

- Let $E = E_1\langle\!\langle y\rangle\!\rangle[y{\leftarrow}E_2]$. We can safely assume that $x \neq y$, since we are working up to α-equivalence. By Lemma 13.2.24 (Plugged variables and domain of type contexts), $y \in \mathsf{dom}(x\colon M;\Gamma)$, and so $\Phi_{E\langle\!\langle x\rangle\!\rangle}$ can only have ES as last rule and be of the form

$$\frac{x\colon M_\Delta;\Delta \vdash^{(m_\Delta,e_\Delta)} E_1\langle\!\langle y\rangle\!\rangle : O \quad x\colon M_\Sigma;\Sigma \vdash^{(m_\Sigma,e_\Sigma)} E_2\langle\!\langle x\rangle\!\rangle : \Delta(y) \quad \Delta(y) \neq \mathbf{0}}{x\colon (M_\Delta \uplus M_\Sigma);(\Delta \setminus\!\setminus y) \uplus \Sigma \vdash^{(m_\Delta+m_\Sigma,e_\Delta+e_\Sigma)} E_1\langle\!\langle y\rangle\!\rangle[y{\leftarrow}E_2\langle\!\langle x\rangle\!\rangle] : O}\ \mathsf{ES}$$

where $M = M_\Delta \uplus M_\Sigma$, $\Gamma = (\Delta \setminus\!\setminus y) \uplus \Sigma$, and $(m,e) = (m_\Delta + m_\Sigma, e_\Delta + e_\Sigma)$.
We can then apply the *i.h.* on the premise in the middle to obtain a splitting $M_\Sigma = M_{\Sigma,1} \uplus M_{\Sigma,2}$ such that for any derivation $\Psi \rhd_{\mathrm{CbNeed}} \Pi \vdash^{(m',e')} v : M_{\Sigma,1}$ there is a derivation $\Phi_{E_2\langle\!\langle v\rangle\!\rangle} \rhd_{\mathrm{CbNeed}}$

$x : M_{\Sigma,2}; \Sigma \uplus \Pi \vdash^{(m_\Sigma+m', e_\Sigma+e'-1)} E_2\langle\langle v\rangle\rangle : \Delta(y)$. We can then construct $\Phi_{E\langle\langle v\rangle\rangle}$ for such Ψ as follows

$$\dfrac{x\colon M_\Delta; \Delta \vdash^{(m_\Delta, e_\Delta)} E_1\langle\langle y\rangle\rangle : O \quad x\colon M_{\Sigma,2}; \Sigma \uplus \Pi \vdash^{(m_\Sigma+m', e_\Sigma+e'-1)} E_2\langle\langle v\rangle\rangle : \Delta(y) \quad \Delta(y) \neq \mathbf{0}}{x\colon (M_\Delta \uplus M_{\Sigma,2}); (\Delta \setminus\setminus y) \uplus \Sigma \uplus \Pi \vdash^{(m_\Delta+m_\Sigma+m', e_\Delta+e_\Sigma+e'-1)} E_1\langle\langle y\rangle\rangle[y \leftarrow E_2\langle\langle v\rangle\rangle] : O}\ \mathsf{ES}$$

where we take $M_1 \coloneqq M_{\Sigma,1}$ and $M_2 \coloneqq M_\Delta \uplus M_{\Sigma,2}$.

□

(Click here to go back to main chapter.)

The following is required to apply Lemma 5.4.3 (Linear Substitution for CbNeed) in the proof of Proposition 5.4.4.2 (Quantitative Subject Reduction for CbNeed - exponential case) to obtain the right indices.

Lemma 13.2.27 (Splitting multi types of CbNeed type derivations)**.**
Let $t \in \Lambda$, $M \coloneqq N \biguplus O$, and let $\Phi \rhd_{CbNeed} \Gamma \vdash^{(m,e)} v : M$ be a type derivation. Then there exist type derivations

$$\Psi \rhd_{CbNeed} \Pi \vdash^{(m',e')} v : N$$
$$\Theta \rhd_{CbNeed} \Delta \vdash^{(m'',e'')} v : O$$

such that $\Gamma = \Pi \biguplus \Delta$ and $(m, e) = (m' + m'', e' + e'')$.

Proof.
Trivial, given that the many typing rule is the only one that can derive a multi type on the right—*i.e.*, the derived type of the subject, in this case M—for values.

□

Proposition 13.2.28 (Quantitative Subject Reduction for CbNeed)**.**
Let $\Phi \rhd_{CbNeed} \Gamma \vdash^{(m,e)} t : M$ be a type derivation such that $M \neq \mathbf{0}$.

1. Multiplicative*: If $t \to_{\mathsf{m}_{\mathrm{CbNeed}}} u$, then $m \geq 1$ and there exists a derivation*

$$\Psi \rhd_{CbNeed} \Gamma \vdash^{(m-1,e)} u : M$$

2. Exponential*: If $t \to_{\mathsf{e}_{\mathrm{CbNeed}}} u$, then $e \geq 1$ and there exists a derivation*

$$\Psi \rhd_{CbNeed} \Gamma \vdash^{(m,e-1)} u : M$$

Proof. *(Click here to go back to main chapter.)*
By induction on the reduction relation $\to_{\mathrm{CbNeed}}$, with $\mapsto_{\mathsf{m}_{\mathrm{CbNeed}}}$ and $r \to_{\mathsf{e}_{\mathrm{CbNeed}}}$ as the base cases, and the closure by CbNeed contexts of $\mapsto_{\mathsf{m}_{\mathrm{CbNeed}}} \cup \mapsto_{\mathsf{e}_{\mathrm{CbNeed}}}$ as the inductive one.

- *Root step for $\to_{\mathsf{m}_{\mathrm{CbNeed}}}$.* Let us assume that $t = S\langle\lambda x.s\rangle m \mapsto_{\mathsf{m}} S\langle s[x \leftarrow m]\rangle = u$, and proceed by induction on S:
  - Let $S = \langle\cdot\rangle$. Then $t = (\lambda x.s)m$ and so the last rule of Φ is either app or $\mathsf{app}_{\mathrm{gc}}$, because they are the only rules whose term in the conclusion type judgement is an application.
    * If app is the last rule of Φ, then the latter is of the form

$$\dfrac{\dfrac{\dfrac{\Pi \vdash^{(m',e')} s : M}{\Pi \setminus\setminus x \vdash^{(m',e')} \lambda x.s : \Pi(x) \multimap M}\ \mathsf{fun}}{\Pi \setminus\setminus x \vdash^{(m',e')} \lambda x.s : [\Pi(x) \multimap M]}\ \mathsf{many} \qquad \Delta \vdash^{(m'',e'')} m : \Pi(x) \quad \Pi(x) \neq \mathbf{0}}{(\Pi \setminus\setminus x) \biguplus \Delta \vdash^{(m'+m''+1, e'+e'')} (\lambda x.s)m : M}\ \mathsf{app}$$

Therefore, $m \geq 1$. Since $\Pi(x) \neq \mathbf{0}$, then we can construct Φ' as follows:

$$\frac{\Pi \vdash^{(m',e')} s : M \quad \Delta \vdash^{(m'',e'')} m : \Pi(x) \quad \Pi(x) \neq \mathbf{0}}{(\Pi \setminus\setminus x) \uplus \Delta \vdash^{(m'+m'',e'+e'')} s[x \leftarrow m] : M}\ \mathrm{ES}$$

Note that $(m' + m'', e' + e') = (m - 1, e)$.

* If $\mathsf{app}_{\mathrm{gc}}$ is the las typing rule of Φ, then the latter is of the form

$$\frac{\dfrac{\dfrac{\Pi \vdash^{(m',e')} s : M}{\Pi \vdash^{(m',e')} \lambda x.s : \mathbf{0} \multimap M}\ \mathsf{fun}}{\Pi \vdash^{(m',e')} \lambda x.s : [\mathbf{0} \multimap M]}\ \mathsf{many}}{\Pi \vdash^{(m'+1,e')} (\lambda x.s)m : M}\ \mathsf{app}_{\mathrm{gc}}$$

with $(m, e) = (m' + 1, e')$ and $\Gamma = \Pi \setminus\setminus x$. Note that $x \notin \mathsf{dom}(\Pi)$, because s is typed with M and $\lambda x.s$ is typed with $\mathbf{0} \multimap M$, so we can construct Φ' as follows:

$$\frac{\Pi \vdash^{(m',e')} s : M \quad \Pi(x) = \mathbf{0}}{\Pi \vdash^{(m',e')} s[x \leftarrow m] : M}\ \mathsf{ES}_{\mathrm{gc}}$$

– Let $S = S'[y \leftarrow \tilde{t}]$. Then $t = S\langle \lambda x.s \rangle m = ((S'\langle \lambda x.s \rangle)[y \leftarrow \tilde{t}])m \mapsto_{\mathsf{m}} (S'\langle s[x \leftarrow m] \rangle)[y \leftarrow \tilde{t}] = S\langle s[x \leftarrow m] \rangle = u$. Since we are working up to α-equivalence, it is safe to assume that $y \notin \mathsf{fv}(\tilde{t})$ and $y \notin \mathsf{fv}(m)$. There are several possible forms of Φ, namely:

* If the last rule is $\mathsf{app}_{\mathrm{gc}}$ and $[y \leftarrow \tilde{t}]$ is appended through rule $\mathsf{ES}_{\mathrm{gc}}$, then Φ is of the form

$$\frac{\dfrac{\Gamma \vdash^{(m',e')} S'\langle \lambda x.s \rangle : [\mathbf{0} \multimap N] \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m',e')} S\langle \lambda x.s \rangle : [\mathbf{0} \multimap N]}\ \mathsf{ES}_{\mathrm{gc}}}{\Gamma \vdash^{(m'+1,e')} S\langle \lambda x.s \rangle m : N}\ \mathsf{app}_{\mathrm{gc}}$$

We then construct the following derivation

$$\frac{\Gamma \vdash^{(m',e')} S'\langle \lambda x.s \rangle : [\mathbf{0} \multimap N]}{\Gamma \vdash^{(m'+1,e')} S'\langle \lambda x.s \rangle m : N}\ \mathsf{app}_{\mathrm{gc}}$$

and apply *i.h.* on it to obtain $m = m' + 1 \geq 1$. Moreover, the *i.h.* also yields a derivation $\Phi'' \rhd_{\mathrm{CbNeed}} \Gamma \vdash^{(m',e')} S'\langle s[x \leftarrow m] \rangle : N$, with which we can then construct Φ' as follows:

$$\frac{\Phi'' \rhd_{\mathrm{CbNeed}} \Gamma \vdash^{(m',e')} S'\langle s[x \leftarrow m] \rangle : N \quad \Gamma(y) \neq \mathbf{0}}{\Gamma \vdash^{(m',e')} S\langle s[x \leftarrow m] \rangle : N}\ \mathsf{ES}_{\mathrm{gc}}$$

Finally, note that $(m', e') = (m - 1, e)$.

* If the last rule is $\mathsf{app}_{\mathrm{gc}}$ and $[y \leftarrow \tilde{t}]$ is appended through rule ES, then Φ is

$$\frac{\Pi \vdash^{(m',e')} S'\langle \lambda x.s \rangle : [\mathbf{0} \multimap M] \quad \Delta \vdash^{(m'',e'')} m : N \quad \Pi(y) = N \neq \mathbf{0}}{\dfrac{(\Pi \setminus\setminus y) \uplus \Delta \vdash^{(m'+m'',e'+e'')} S\langle \lambda x.s \rangle : [\mathbf{0} \multimap M]}{(\Pi \setminus\setminus y) \uplus \Delta \vdash^{(m'+m''+1,e'+e'')} S\langle \lambda x.s \rangle m : M}\ \mathsf{app}_{\mathrm{gc}}}\ \mathsf{ES}$$

We can then construct the following derivation

$$\frac{\Pi \vdash^{(m',e')} S'\langle \lambda x.s \rangle : [\mathbf{0} \multimap M]}{\Pi \vdash^{(m'+1,e')} S'\langle \lambda x.s \rangle m : M}\ \mathsf{app}_{\mathrm{gc}}$$

Applying *i.h.* on it yields a derivation $\Phi'' \triangleright_{\mathrm{CbNeed}} \Pi \vdash^{(m',e')} S'\langle s[x{\leftarrow}m]\rangle : M$ and implies the fact that $m = m' + m'' + 1 \geq m' + 1 \geq 1$. Finally, we construct Φ' as follows:

$$\frac{\Phi'' \triangleright_{\mathrm{CbNeed}} \Pi \vdash^{(m',e')} S'\langle s[x{\leftarrow}m]\rangle : M \quad \Delta \vdash^{(m'',e'')} m : N \quad \Pi(y) = N \neq \mathbf{0}}{(\Pi \setminus\setminus y) \uplus \Delta \vdash^{(m'+m'',e'+e'')} S\langle s[x{\leftarrow}m]\rangle : M}\ \mathsf{ES}$$

Note that $(m' + m'', e' + e'') = (m - 1, e)$.

* If the last rule is app and $[y{\leftarrow}\tilde{t}]$ is appended through rule $\mathsf{ES}_{\mathrm{gc}}$, then Φ is

$$\frac{\dfrac{\Pi \vdash^{(m',e')} S'\langle \lambda x.s\rangle : [M \multimap N] \quad \Pi(y) = \mathbf{0}}{\Pi \vdash^{(m',e')} S\langle \lambda x.s\rangle : [M \multimap N]}\ \mathsf{ES}_{\mathrm{gc}} \qquad \Delta \vdash^{(m'',e'')} m : M}{\Pi \uplus \Delta \vdash^{(m'+m''+1,e'+e'')} S\langle \lambda x.s\rangle m : N}\ \mathsf{app} M \neq \mathbf{0}$$

We are now able to give the following derivation

$$\frac{\Pi \vdash^{(m',e')} S'\langle \lambda x.s\rangle : [M \multimap N] \quad \Delta \vdash^{(m'',e'')} m : M}{\Pi \uplus \Delta \vdash^{(m'+m''+1,e'+e'')} S'\langle \lambda x.s\rangle m : N}\ \mathsf{app}$$

on which application of *i.h.* gives that $m = m' + m'' + 1 \geq 1$, and yields a derivation $\Phi'' \triangleright_{\mathrm{CbNeed}} \Pi \uplus \Delta \vdash^{(m'+m'',e'+e'')} S'\langle s[x{\leftarrow}m]\rangle : N$, thus allowing us to construct Φ' as follows:

$$\frac{\Phi'' \triangleright_{\mathrm{CbNeed}} \Pi \uplus \Delta \vdash^{(m'+m'',e'+e'')} S'\langle s[x{\leftarrow}m]\rangle : N \quad (\Pi \uplus \Delta)(y) = \mathbf{0}}{\Phi'' \triangleright_{\mathrm{CbNeed}} \Pi \uplus \Delta \vdash^{(m'+m'',e'+e'')} S\langle s[x{\leftarrow}m]\rangle : N}\ \mathsf{ES}_{\mathrm{gc}}$$

Note that $y \notin \mathsf{fv}(\tilde{t})$ and so via Lemma 5.4.1 (Relevance of the CbNeed type system) we know that $\Delta(y) = \mathbf{0}$; hence, the use of rule $\mathsf{ES}_{\mathrm{gc}}$ in Φ' is correct. Moreover, note that $(m' + m'', e' + e'') = (m, e)$.

* If the last rule is app and $[y{\leftarrow}\tilde{t}]$ is appended through rule ES, then Φ is

$$\frac{\dfrac{\Pi \vdash^{(m',e')} S'\langle \lambda x.s\rangle : [M \multimap N] \quad \Delta \vdash^{(m'',e'')} \tilde{t} : \Pi(y) \quad \Pi(y) \neq \mathbf{0}}{(\Pi \setminus\setminus y) \uplus \Delta \vdash^{(m'+m'',e'+e'')} S\langle \lambda x.s\rangle : [M \multimap N]}\ \mathsf{ES} \qquad \Sigma \vdash^{(m''',e''')} m : M}{((\Pi \setminus\setminus y) \uplus \Delta) \uplus \Sigma \vdash^{((m'+m'')+m'''+1,(e'+e'')+e''')} S\langle \lambda x.s\rangle m : N}\ \mathsf{app}$$

Since $y \notin \mathsf{fv}(m) \cup \mathsf{fv}(\tilde{t})$ then we know through Lemma 5.4.1 (Relevance of the CbNeed type system) that $y \notin \mathsf{dom}(\Delta)$ and $y \notin \mathsf{dom}(\Sigma)$. Now, applying *i.h.* on the following derivation

$$\frac{\Pi \vdash^{(m',e')} S'\langle \lambda x.s\rangle : [M \multimap N] \quad \Sigma \vdash^{(m''',e''')} m : M}{\Pi \uplus \Sigma \vdash^{(m'+m'''+1,e'+e''')} S'\langle \lambda x.s\rangle m : N}\ \mathsf{app}$$

yields a derivation $\Phi'' \triangleright_{\mathrm{CbNeed}} \Pi \uplus \Sigma \vdash^{(m'+m''',e'+e''')} S'\langle s[x{\leftarrow}m]\rangle : N$ and implies $m = (m' + m'') + m''' + 1 \geq m' + m''' + 1 \geq 1$. Finally, given that $(\Pi \uplus \Sigma)(y) = \Pi(y)$, we can finally construct Φ' as follows:

$$\frac{\Phi'' \triangleright_{\mathrm{CbNeed}} \Pi \uplus \Sigma \vdash^{(m'+m''',e'+e''')} S'\langle s[x{\leftarrow}m]\rangle : N \quad \Delta \vdash^{(m'',e'')} \tilde{t} : \Pi(y) \quad \Pi(y) \neq \mathbf{0}}{((\Pi \uplus \Sigma) \setminus\setminus y) \uplus \Delta \vdash^{((m'+m''')+m'',(e'+e''')+e'')} S\langle s[x{\leftarrow}m]\rangle : N}\ \mathsf{ES}$$

Note that $((m' + m''') + m'', (e' + e''') + e'') = (m - 1, e)$, and that since $y \notin \mathsf{dom}(\Sigma)$ then $((\Pi \uplus \Sigma) \setminus\setminus y) \uplus \Delta = ((\Pi \setminus\setminus y) \uplus \Delta) \uplus \Sigma$.

All other typing rules are not possible as the final rule in Φ. In particular, rule many is not possible because the term in its final judgement has to be an abstraction, not an application term.

- *Root step for* $\to_{\mathsf{e}_{\mathrm{CbNeed}}}$. Let $t = E\langle\!\langle x \rangle\!\rangle[x{\leftarrow}S\langle v\rangle] \mapsto_{\mathsf{e}_{\mathrm{CbNeed}}} S\langle E\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle$. We can infer from Lemma 5.4.1 (Relevance of the CbNeed type system) that Φ can only have ES as its last typing rule, and so can only be of the form

$$\frac{\Phi_{E\langle\!\langle x\rangle\!\rangle} \triangleright_{\mathrm{CbNeed}} x\colon O; \Pi \vdash^{(m',e')} E\langle\!\langle x\rangle\!\rangle : M \quad \Phi_{S\langle v\rangle} \triangleright_{\mathrm{CbNeed}} \Delta \vdash^{(m'',e'')} S\langle v\rangle : O \quad O \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'')} E\langle\!\langle x\rangle\!\rangle[x{\leftarrow}S\langle v\rangle] : M}\ \mathsf{ES}$$

Note that $x \notin \mathsf{dom}(\Delta)$, because otherwise Lemma 5.4.1 (Relevance of the CbNeed type system) would imply $x \in \mathsf{fv}(S\langle v\rangle)$ and this cannot be the case, given that we are working up to α-equivalence.
We now proceed to prove by induction on S that whenever we have $\Phi_{E\langle\!\langle x\rangle\!\rangle}$ and $\Phi_{S\langle v\rangle}$ we can derive $\Phi' \triangleright_{\mathrm{CbNeed}} \Pi \uplus \Delta \vdash^{(m'+m'',e'+e''-1)} S\langle E\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle : M$.

- Let $S \coloneqq \langle\cdot\rangle$. First of all, applying Lemma 5.4.3 (Linear Substitution for CbNeed) on $\Phi_{E\langle\!\langle x\rangle\!\rangle}$ yields a splitting $O = O_1 \uplus O_2$ such that for every derivation $\Psi \triangleright_{\mathrm{CbNeed}} \Sigma \vdash^{(m''',e''')} v : O_1$ there exists a derivation $x\colon O_2; \Pi \uplus \Sigma \vdash^{(m'+m''',e'+e'''-1)} E\langle\!\langle v\rangle\!\rangle : M$. In particular, if $O_2 = \mathbf{0}$, then we can construct the desired derivation Φ' as follows

$$\cfrac{\cfrac{x\colon O; \Pi \vdash^{(m',e')} E\langle\!\langle x\rangle\!\rangle : M \qquad \Delta \vdash^{(m'',e'')} v : O}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e''-1)} E\langle\!\langle v\rangle\!\rangle : M}\ \textit{Lemma } 5.4.3}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e''-1)} E\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v] : M}\ \mathsf{ES}_{\mathrm{gc}}$$

On the other hand, if $O_2 \neq \mathbf{0}$, we can then apply Lemma 13.2.27 (Splitting multi types of CbNeed type derivations) on $\Phi_{S\langle v\rangle}$ to yield derivations $\Phi_{O_1} \triangleright_{\mathrm{CbNeed}} \Delta_{O_1} \vdash^{(m''_{O_1},e''_{O_1})} v : O_1$ and $\Phi_{O_2} \triangleright_{\mathrm{CbNeed}} \Delta_{O_2} \vdash^{(m''_{O_2},e''_{O_2})} v : O_2$ such that $\Delta = \Delta_{O_1} \uplus \Delta_{O_2}$ and $(m'', e'') = (m''_{O_1} + m''_{O_2}, e''_1 + e''_{O_2})$. Thus, we are now able to combine all these derivations to construct Φ' as follows

$$\cfrac{\cfrac{x\colon O; \Pi \vdash^{(m',e')} E\langle\!\langle x\rangle\!\rangle : M \quad \Delta_{O_1} \vdash^{(m''_{O_1},e''_{O_1})} v : O_1}{x\colon O_2; \Pi \uplus \Delta_{O_1} \vdash^{(m'+m''_{O_1},e'+e''_{O_1}-1)} E\langle\!\langle v\rangle\!\rangle : M}\ L.5.4.3 \qquad \Delta_{O_2} \vdash^{(m''_{O_2},e''_{O_2})} S\langle v\rangle : O_2}{\Pi \uplus \Delta_{O_1} \uplus \Delta_{O_2} \vdash^{(m'+m''_{O_1}+m''_{O_2},e'+e''_{O_1}-1+e''_{O_2})} E\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v] : M}\ \mathsf{ES}$$

- Let $S \coloneqq S'[y{\leftarrow}t]$. There are two possible final typing rules in $\Phi_{S\langle v\rangle}$, namely ES and $\mathsf{ES}_{\mathrm{gc}}$.
  - Let $\Phi_{S\langle v\rangle}$ be of the form

$$\frac{\Delta \vdash^{(m'',e'')} S'\langle v\rangle : O \quad \Delta(y) = \mathbf{0}}{\Delta \vdash^{(m'',e'')} S\langle v\rangle : O}\ \mathsf{ES}_{\mathrm{gc}}$$

Note that since we are working up to α-equivalence we can safely assume that $y \notin \mathsf{fv}(E\langle\!\langle x\rangle\!\rangle)$, and so via Lemma 5.4.1 (Relevance of the CbNeed type system) we have that $y \notin \mathsf{dom}(\Pi)$. We can then construct Φ' by application of the *i.h.* as follows

$$\cfrac{\cfrac{x : O; \Pi \vdash^{(m',e')} E\langle\!\langle x\rangle\!\rangle : M \quad \Delta \vdash^{(m'',e'')} S'\langle v\rangle : O}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'')} S'\langle E\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle : M}\ \textit{i.h.}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'')} S\langle E\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle : M}\ \mathsf{ES}_{\mathrm{gc}}$$

∗ Let $\Phi_{S\langle v\rangle}$ be of the form

$$\frac{y\colon P; \Delta_1 \vdash^{(m_1'', e_1'')} S'\langle v\rangle : O \quad \Delta_2 \vdash^{(m_2'', e_2'')} t : P \quad P \neq \mathbf{0}}{\Delta_1 \uplus \Delta_2 \vdash^{(m_1''+m_2'', e_1''+e_2'')} S\langle v\rangle : O}\ \mathsf{ES}$$

where $\Delta = \Delta_1 \uplus \Delta_2$ and $(m'', e'') = (m_1'' + m_2'', e_1'' + e_2'')$. Note that $y \notin \mathsf{fv}(E\langle\!\langle x\rangle\!\rangle)$, and so via Lemma 5.4.1 (Relevance of the CbNeed type system) we have that $y \notin \mathsf{dom}(\Pi)$. We can then construct Φ' by application of the *i.h.* and a rearranging of Φ as follows

$$\frac{\dfrac{x : O; \Pi \vdash^{(m', e')} E\langle\!\langle x\rangle\!\rangle : M \quad y : P; \Delta_1 \vdash^{(m_1'', e_1'')} S'\langle v\rangle : O}{y : P; \Pi \uplus \Delta_1 \vdash^{(m'+m_1'', e'+e_1''-1)} S'\langle E\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle : M}\ \textit{i.h.} \quad \Delta_2 \vdash^{(m_2'', e_2'')} t : P}{\Pi \uplus \Delta_1 \uplus \Delta_2 \vdash^{(m'+m_1''+m_2'', e'+e_1''-1+e_2'')} S\langle E\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle : M}\ \mathsf{ES}$$

- *Contextual closure.* We proceed by induction on the derivation of $t = E\langle t_1\rangle \to_{\text{CbNeed}} E\langle t_2\rangle = u$:
  - If $E = \langle\cdot\rangle$, then $t \mapsto_{\mathsf{m}} u$ or $t \mapsto_{\mathsf{e}_{\text{CbNeed}}} u$, and the statement holds as we have just proved.
  - Let $E = E_1 s$. This implies that the last typing rule in Φ is either app_{gc} or app. We only cover the case where $E\langle t_1\rangle \to_{\mathsf{m}_{\text{CbNeed}}} E\langle t_2\rangle$ and Φ ends in rule app, leaving the rest of the (analogous) cases to the reader.
    Now, Φ is of the form

$$\frac{\Pi \vdash^{(m', e')} E_1\langle t_1\rangle : [N \multimap M] \quad \Delta \vdash^{(m'', e'')} s : N}{\Pi \uplus \Delta \vdash^{(m'+m''+1, e'+e'')} E_1\langle t_1\rangle s : M}\ \mathsf{app}$$

    Then we apply *i.h.* on the left premise of the last rule, obtaining a type derivation whose final judgement is $\Pi \vdash^{(m'-1, e')} E_1\langle t_2\rangle : [N \multimap M]$, thus allowing us to construct Φ' as folows:

$$\frac{\Pi \vdash^{(m'-1, e')} E_1\langle t_2\rangle : [N \multimap M] \quad \Delta \vdash^{(m'', e'')} s : N}{\Pi \uplus \Delta \vdash^{(m'+m'', e'+e'')} E_1\langle t_2\rangle s : M}\ \mathsf{app}_b$$

    Note that $(m' + m'', e' + e'') = (m - 1, e)$.
  - Let $E = E_1[x{\leftarrow}s]$. This implies that the last typing rule in Φ is either ES_{gc} or ES. We only cover the case where $E\langle t_1\rangle \to_{\mathsf{m}_{\text{CbNeed}}} E\langle t_2\rangle$ and Φ ends in rule ES, leaving the rest of the (analogous) cases to the reader.
    Now, Φ is of the form

$$\frac{\Pi \vdash^{(m', e')} E_1\langle t_1\rangle : M \quad \Delta \vdash^{(m'', e'')} s : \Pi(x) \quad \Pi(x) \neq \mathbf{0}}{(\Pi \setminus\setminus x) \uplus \Delta \vdash^{(m'+m'', e'+e'')} E_1\langle t_1\rangle[x{\leftarrow}s] : M}\ \mathsf{ES}$$

    Applying *i.h.* on the left premise of the last rule yields a derivation whose final judgement is $\Pi \vdash^{(m'-1, e')} E_1\langle t_2\rangle : M$, thus allowing us to construct Φ' as follows:

$$\frac{\Pi \vdash^{(m'-1, e')} E_1\langle t_2\rangle : M \quad \Delta \vdash^{(m'', e'')} s : \Pi(x) \quad \Pi(x) \neq \mathbf{0}}{(\Pi \setminus\setminus x) \uplus \Delta \vdash^{(m'-1+m'', e'+e'')} E_1\langle t_2\rangle[x{\leftarrow}s] : M}\ \mathsf{ES}$$

    Note that $(m' - 1 + m'', e' + e'') = (m - 1, e)$

– Let $E = E_1\langle\langle x\rangle\rangle[x{\leftarrow}E_2]$. We only consider the case where

$$E_1\langle\langle x\rangle\rangle[x{\leftarrow}E_2\langle t_1\rangle] \to_{\mathsf{m}_{\mathrm{CbNeed}}} E_1\langle\langle x\rangle\rangle[x{\leftarrow}E_2\langle t_2\rangle]$$

leaving the other (analogous) case to the reader.
First of all, Lemma 5.4.1 (Relevance of the CbNeed type system) implies that the last rule in Φ is ES; *i.e.*, Φ is of the form

$$\frac{\Pi \vdash^{(m',e')} E_1\langle x\rangle : M \quad \Delta \vdash^{(m'',e'')} E_2\langle t_1\rangle : \Pi(x) \quad \Pi(x) \neq \mathbf{0}}{(\Pi \setminus\setminus x) \uplus \Delta \vdash^{(m'+m'',e'+e'')} E_1\langle\langle x\rangle\rangle[x{\leftarrow}E_2\langle t_1\rangle] : M}\ \mathsf{ES}$$

Applying now the *i.h.* on the premise in the middle of the last rule yields a derivation with conclusion $\Delta \vdash^{(m''-1,e'')} E_2\langle t_2\rangle : \Pi(x)$, thus allowing us to construct Φ' as follows

$$\frac{\Pi \vdash^{(m',e')} E_1\langle\langle x\rangle\rangle : M \quad \Delta \vdash^{(m''-1,e'')} E_2\langle t_2\rangle : \Pi(x) \quad \Pi(x) \neq \mathbf{0}}{(\Pi \setminus\setminus x) \uplus \Delta \vdash^{(m'+m''-1,e'+e'')} E_1\langle\langle x\rangle\rangle[x{\leftarrow}E_2\langle t_2\rangle] : M}\ \mathsf{ES}$$

verifying that $(m' + m'' - 1, e' + e'') = (m - 1, e)$.

□

(Click here to go back to main chapter.)

Theorem 13.2.29 (Tight Correctness for CbNeed)**.**

Let $t \in \Lambda_{\mathsf{L}}$ be closed and $\Phi \triangleright_{CbNeed} \Gamma \vdash^{(m,e)} t : M$ be a type derivation such that $M \neq \mathbf{0}$. Then there exists $u \in \Lambda_{\mathsf{L}}$ such that

1. $\mathsf{norm}(u)$,
2. *there exists a reduction sequence $d : t \to^*_{\mathrm{CbNeed}} u$, and*
3. *$|d|_{\mathsf{m}} \leq m$ and $|d|_{\mathsf{e}} \leq e$.*

Moreover, if Φ is tight then $(m, e) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}})$.

Proof. *(Click here to go back to main chapter.)*

By induction on $m + e$, and proceeding by case analysis on whether $t \to_{\mathrm{CbNeed}}$-reduces or not. Note that if t is in $\to_{\mathrm{CbNeed}}$-normal form, then we only have to prove the *moreover* part, that states that if Φ is tight then $(m, e) = (0, 0)$, which follows from Proposition 5.4.2 (Typing properties of CbNeed-normal forms).

Otherwise, if $t \to_{\mathrm{CbNeed}} s$ for some $s \in \Lambda_{\mathsf{L}}$, then there are two cases:

1. *Multiplicative steps*: Let $t \to_{\mathsf{m}_{\mathrm{CbNeed}}} s$. By Proposition 5.4.4.1 (Quantitative Subject Reduction for CbNeed - multiplicative steps), there exists $\Psi \triangleright_{\mathrm{CbN}} \Gamma \vdash^{(m-1,e)} s : M$. By *i.h.* on Ψ, there exist u and d' such that $\mathsf{norm}(u)$ and $d' : s \to^*_{\mathrm{CbNeed}} u$, $|d'|_{\mathsf{m}} \leq m - 1$ and $|d'|_{\mathsf{e}} \leq e$. Just note that $t \to_{\mathsf{m}_{\mathrm{CbNeed}}} s$ and so, if $d : t \to^*_{\mathrm{CbNeed}} u$ is d' preceeded by such a step, we have $|d|_{\mathsf{m}} \leq m$ and $|d|_{\mathsf{e}} \leq e$.
   If Φ is tight, then so is Ψ. Then $|d'|_{\mathsf{m}} = m - 1$ and $|d'|_{\mathsf{e}} = e$ by *i.h.*, that give $|d|_{\mathsf{m}} = m$ and $|d|_{\mathsf{e}} = e$.
2. *Exponential steps*: Let $t \to_{\mathsf{e}_{\mathrm{CbNeed}}} s$. By Proposition 5.4.4.2 (Quantitative Subject Reduction for CbNeed - exponential steps), there exists $\Psi \triangleright_{\mathrm{CbNeed}} \Gamma \vdash^{(m,e-1)} s : M$. By *i.h.*, there exist u and d' such that $\mathsf{norm}(u)$ and $d' : s \to^*_{\mathrm{CbNeed}} u$, $|d'|_{\mathsf{m}} \leq m$ and $|d'|_{\mathsf{e}} \leq e - 1$. Just note that $t \to_{\mathsf{e}} s$ and so, if $d : t \to^*_{\mathrm{CbNeed}} u$ is d' preceeded by such a step, we have $|d|_{\mathsf{m}} \leq m$ and $|d|_{\mathsf{e}} \leq e$. Finally, if Φ is tight, then so is Ψ. Then $|d'|_{\mathsf{m}} = m$ and $|d'|_{\mathsf{e}} = e - 1$ by *i.h.*, that give $|d|_{\mathsf{m}} = m$ and $|d|_{\mathsf{e}} = e$.

□

(Click here to go back to main chapter.)

13.2.6 CbNeed completeness

Proposition 13.2.30 (Tight typability of CbNeed-normal forms)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be such that $\mathsf{norm}(t)$. Then there exists tight type derivation

$$\Phi \rhd_{CbNeed} \emptyset \vdash^{(m,e)} t : [\mathsf{norm}]$$

Proof. (Click here to go back to main chapter.)
By Proposition 4.6.1 (Syntactic characterization of closed normal forms - CbNeed), We can easily prove by induction on predicate $\mathsf{norm}(.)$ that if $\mathsf{norm}(t)$ then $t = S\langle \lambda x.u \rangle$, for some abstraction $\lambda x.u$ and substitution context $S = \langle\cdot\rangle[x_1 \leftarrow t_1]...[x_n \leftarrow t_n]$, with $n \geq 0$.
Therefore, we can derive Φ as follows

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash^{(0,0)} \lambda x.u : \mathsf{norm}}\,\mathsf{norm}}{\vdash^{(0,0)} \lambda x.u : [\mathsf{norm}]}\,\mathsf{many}}{\vdash^{(0,0)} \lambda x.u[x_1 \leftarrow t_1] : [\mathsf{norm}]}\,\mathsf{ES_{gc}}}{\vdots}\,\mathsf{ES_{gc}}}{\vdash^{(0,0)} \lambda x.u[x_1 \leftarrow t_1]\ldots[x_n \leftarrow t_n] : [\mathsf{norm}]}\,\mathsf{ES_{gc}}$$

□

(Click here to go back to main chapter.)

Lemma 13.2.31 (Linear Removal for CbNeed)**.**
Let $\Phi \rhd_{CbNeed} \Gamma; x : M \vdash^{(m,e)} E\langle\langle v \rangle\rangle : N$ be a type derivation, where $v \in \mathsf{Val}$ and $x \notin \mathsf{fv}(v)$. Then there exist type derivations

$$\begin{array}{lll} \Psi & \rhd_{CbNeed} & \Pi \vdash^{(m',e')} v : O \\ \Theta & \rhd_{CbNeed} & \Delta; x : (M \uplus O) \vdash^{(m'',e'')} E\langle\langle x \rangle\rangle : N \end{array}$$

such that $\Gamma = \Pi \biguplus \Delta$ and $(m, e) = (m' + m'', e' + e'' - 1)$.

Proof. (Click here to go back to main chapter.)
We prove this by induction on the context E:

- Let $E = \langle\cdot\rangle$. By taking $\Gamma_v := \Gamma$, $\Gamma' := \emptyset$, $M := O$, $(m_v, e_v) := (m, e)$, and $(m', e') := (0, 1)$, we can then take $\Phi_v := \Phi$ and construct $\Phi_{E\langle\langle x \rangle\rangle}$ as follows:

$$\dfrac{}{x : M \vdash^{(0,1)} x : M}\,\mathsf{ax}$$

  verifying that $(m, e) = (m_v, e_v) = (0 + m_v, 1 + e_v - 1) = (m' + m_v, e' + e_v - 1)$ and $\Gamma = \emptyset \uplus \Gamma = \Gamma' \uplus \Gamma_v$.
- Let $E = E_1 t$, and so $E\langle\langle v \rangle\rangle = E_1\langle\langle v \rangle\rangle t$. There are two possible last rules in Φ, namely app or $\mathsf{app_{gc}}$.
  Let us assume Φ is of the form

$$\dfrac{\Phi_{E_1\langle\langle v \rangle\rangle} \rhd_{\mathrm{CbNeed}} \Pi \vdash^{(m_\Pi, e_\Pi)} E_1\langle\langle v \rangle\rangle : [O' \multimap O] \quad \Delta \vdash^{(m_\Delta, e_\Delta)} t : O' \quad O' \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m_\Pi + m_\Delta + 1, e_\Pi + e_\Delta)} E_1\langle\langle v \rangle\rangle t : O}\,\mathsf{app}$$

Then we can apply the *i.h.* on $\Phi_{E_1\langle\langle v\rangle\rangle}$ to obtain a type M and type derivations

$$\Phi_v \triangleright_{\mathrm{CbNeed}} \Pi_v \vdash^{(m_v,e_v)} v : M$$

and

$$\Phi_{E_1\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Pi' \uplus \{x\colon M\} \vdash^{(m'',e'')} E_1\langle\langle x\rangle\rangle : [O' \multimap O]$$

such that $\Pi = \Pi' \uplus \Pi_v$ and $(m_\Pi, e_\Pi) = (m'' + m_v, e'' + e_v - 1)$.
Thus, we are able to construct $\Phi_{E\langle\langle x\rangle\rangle}$ as follows

$$\frac{\Phi_{E_1\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Pi' \uplus \{x\colon M\} \vdash^{(m'',e'')} E_1\langle\langle x\rangle\rangle : [O' \multimap O] \quad \Delta \vdash^{(m_\Delta,e_\Delta)} t : O' \quad O' \neq \mathbf{0}}{\Pi' \uplus \{x\colon M\} \uplus \Delta \vdash^{(m''+m_\Delta+1,e''+e_\Delta)} E_1\langle\langle x\rangle\rangle t : O}\ \mathsf{app}$$

and, by taking $\Gamma' \coloneqq \Pi' \uplus \Delta$, $\Gamma_v \coloneqq \Pi_v$, and $(m', e') \coloneqq (m'' + m_\Delta + 1, e'' + e_\Delta)$, then we verify that

$$\Gamma = \Pi \uplus \Delta = \Pi' \uplus \Pi_v \uplus \Delta = \Gamma' \uplus \Gamma_v$$

and

$$(m, e) = (m_\Pi + m_\Delta + 1, e_\Pi + e_\Delta) = (m'' + m_v + m_\Delta + 1, e'' + e_v - 1 + e_\Delta) = (m' + m_v, e' + e_v - 1)$$

Now, let us assume Φ is of the form

$$\frac{\Phi_{E_1\langle\langle v\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Gamma \vdash^{(m-1,e)} E_1\langle\langle v\rangle\rangle : [\mathbf{0} \multimap O]}{\Gamma \vdash^{(m,e)} E_1\langle\langle v\rangle\rangle t : O}\ \mathsf{app}_{\mathrm{gc}}$$

We then apply the *i.h.* on $\Phi_{E_1\langle\langle v\rangle\rangle}$ to obtain type M and type derivations

$$\Phi_v \triangleright_{\mathrm{CbNeed}} \Gamma_v \vdash^{(m_v,e_v)} v : M$$

and

$$\Phi_{N_1\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Gamma' \uplus \{x : M\} \vdash^{(m'',e'')} N_1\langle\langle x\rangle\rangle : [O' \multimap O]$$

such that $\Gamma = \Gamma' \uplus \Gamma_v$ and $(m - 1, e) = (m'' + m_v, e'' + e_v - 1)$.
Thus, we are able to construct $\Phi_{E\langle\langle x\rangle\rangle}$ as follows

$$\frac{\Phi_{E_1\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Gamma' \uplus \{x\colon M\} \vdash^{(m'',e'')} E_1\langle\langle x\rangle\rangle : [O' \multimap O]}{\Gamma' \uplus \{x : M\} \vdash^{(m''+1,e'')} E_1\langle\langle x\rangle\rangle t : O}\ \mathsf{app}_{\mathrm{gc}}$$

and, by taking $(m', e') = (m'' + 1, e'')$, then verify that

$$(m, e) = (m'' + m_v + 1, e'' + e_v - 1) = (m' + m_v, e' + e_v - 1)$$

- Let $E = E_1[y{\leftarrow}t]$, and so $E\langle\langle v\rangle\rangle = E_1\langle\langle v\rangle\rangle[y{\leftarrow}t]$. Note that we can safely assume that $x \neq y$, since we are working up to α-equivalence and y has a binding occurrence in E while x represents a free variable. Moreover, note that $y \notin \mathsf{fv}(v)$, since otherwise $E\langle\langle v\rangle\rangle$ would not be well-defined.
  There are two possible last rules in Φ, namely ES or $\mathsf{ES}_{\mathrm{gc}}$.
  Let Φ be of the form

$$\frac{\Phi_{E_1\langle\langle v\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Pi \vdash^{(m_\Pi,e_\Pi)} E_1\langle\langle v\rangle\rangle : O \quad \Delta \vdash^{(m_\Delta,e_\Delta)} t : \Pi(y) \quad \Pi(y) \neq \mathbf{0}}{(\Pi \setminus\setminus y) \uplus \Delta \vdash^{(m_\Pi+m_\Delta,e_\Pi+e_\Delta)} E_1\langle\langle v\rangle\rangle[y{\leftarrow}t] : O}\ \mathsf{ES}$$

where $\Gamma = (\Pi \setminus\setminus y) \uplus \Delta$, $(m, e) = (m_\Pi + m_\Delta, e_\Pi + e_\Delta)$.
Then we can apply the *i.h.* on $\Phi_{E_1\langle\langle v\rangle\rangle}$ to obtain type M and type derivations

$$\Phi_v \triangleright_{\mathrm{CbNeed}} \Pi_v \vdash^{(m_v, e_v)} v : M$$

and

$$\Phi_{E_1\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Pi' \uplus \{x\colon M\} \vdash^{(m'', e'')} E_1\langle\langle x\rangle\rangle : O$$

such that $\Pi = \Pi' \uplus \Pi_v$ and $(m_\Pi, e_\Pi) = (m'' + m_v, e'' + e_v - 1)$.
Moreover, since $y \neq x$ and $y \notin \mathsf{dom}(\Pi_v)$—otherwise Lemma 5.4.1 (Relevance of the CbNeed type system) would imply that $y \in \mathsf{fv}(v)$, which we already know not to be the case—then $(\Pi' \uplus \{x\colon M\})(y) = \Pi(y)$ and so we are able to construct $\Phi_{E\langle\langle x\rangle\rangle}$ as follows

$$\frac{\Phi_{E_1\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Pi' \uplus \{x\colon M\} \vdash^{(m'', e'')} E_1\langle\langle x\rangle\rangle : O \quad \Delta \vdash^{(m_\Delta, e_\Delta)} t : \Pi(y) \quad \Pi(y) \neq \mathbf{0}}{((\Pi' \uplus \{x : M\}) \setminus\setminus y) \uplus \Delta \vdash^{(m'' + m_\Delta, e'' + e_\Delta)} E_1\langle\langle x\rangle\rangle[y{\leftarrow}t] : O}\ \mathsf{ES}$$

Now, by taking $\Gamma' \coloneqq (\Pi' \setminus\setminus y) \uplus \Delta$, $\Gamma_v \coloneqq \Pi_v$, and $(m', e') \coloneqq (m'' + m_\Delta, e'' + e_\Delta)$, we can verify that

$$\Gamma = (\Pi \setminus\setminus y) \uplus \Delta = ((\Pi' \uplus \Pi_v) \setminus\setminus y) \uplus \Delta = (\Pi' \setminus\setminus y) \uplus \Pi_v \uplus \Delta = \Gamma' \uplus \Gamma$$

and

$$(m, e) = (m_\Pi + m_\Delta, e_\Pi + e_\Delta) = ((m'' + m_v) + m_\Delta, (e'' + e_v - 1) + e_\Delta) = (m' + m_v, e' + e_v)$$

If Φ is instead of the form

$$\frac{\Phi_{E_1\langle\langle v\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Gamma \vdash^{(m,e)} E_1\langle\langle v\rangle\rangle : O \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e)} E_1\langle\langle v\rangle\rangle[y{\leftarrow}t] : O}\ \mathsf{ES}_{\mathrm{gc}}$$

then we can apply the *i.h.* on $\Phi_{E_1\langle\langle v\rangle\rangle}$ to obtain a type M and type derivations

$$\Phi_v \triangleright_{\mathrm{CbNeed}} \Pi_v \vdash^{(m_v, e_v)} v : M$$

and

$$\Phi_{E_1\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Gamma' \uplus \{x\colon M\} \vdash^{(m', e')} E_1\langle\langle x\rangle\rangle : O$$

such that $\Gamma = \Gamma' \uplus \Gamma_v$ and $(m, e) = (m' + m_v, e' + e_v - 1)$. Note that this type context and these indices are exactly as desired, and so we can finally construct $\Phi_{E\langle\langle x\rangle\rangle}$ as follows:

$$\frac{\Phi_{E_1\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Gamma' \uplus \{x\colon M\} \vdash^{(m', e')} E_1\langle\langle x\rangle\rangle : O}{\Gamma' \uplus \{x : M\} \vdash^{(m', e')} E_1\langle\langle x\rangle\rangle[y{\leftarrow}t] : O}\ \mathsf{ES}_{\mathrm{gc}}$$

- Let $E = E_1\langle\langle y\rangle\rangle[y{\leftarrow}E_2]$, and so $E\langle\langle v\rangle\rangle = E_1\langle\langle y\rangle\rangle[y{\leftarrow}E_2\langle\langle v\rangle\rangle]$. Once again, we assume $x \neq y$. By Lemma 13.2.24 (Plugged variables and domain of type contexts), there is only one possible form of Φ, namely:

$$\frac{\Pi \vdash^{(m_\Pi, e_\Pi)} E_1\langle\langle y\rangle\rangle : O \quad \Phi_{E_2\langle\langle v\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Delta \vdash^{(m_\Delta, e_\Delta)} E_2\langle\langle v\rangle\rangle : \Pi(y) \quad \Pi(y) \neq \mathbf{0}}{(\Pi \setminus\setminus y) \uplus \Delta \vdash^{(m_\Pi + m_\Delta, e_\Pi + e_\Delta)} E_1\langle\langle y\rangle\rangle[y{\leftarrow}E_2\langle\langle v\rangle\rangle] : O}\ \mathsf{ES}$$

where $\Gamma = (\Pi \setminus\setminus y) \uplus \Delta$, $(m, e) = (m_\Pi + m_\Delta, e_\Pi + e_\Delta)$.

Then we can apply the *i.h.* on $\Phi_{E_2\langle\langle v\rangle\rangle}$ to obtain type M and type derivations

$$\Phi_v \triangleright_{\mathrm{CbNeed}} \Delta_v \vdash^{(m_v,e_v)} v : M$$

and

$$\Phi_{E_2\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Delta' \uplus \{x\colon M\} \vdash^{(m'',e'')} E_2\langle\langle x\rangle\rangle : \Pi(y)$$

such that $\Delta = \Delta' \uplus \Delta_v$ and $(m_\Delta, e_\Delta) = (m'' + m_v, e'' + e_v - 1)$.
We can then construct $\Phi_{E\langle\langle x\rangle\rangle}$ as follows

$$\frac{\Pi \vdash^{(m_\Pi,e_\Pi)} E_1\langle\langle y\rangle\rangle : O \quad \Phi_{E_2\langle\langle x\rangle\rangle} \triangleright_{\mathrm{CbNeed}} \Delta' \uplus \{x\colon M\} \vdash^{(m'',e'')} E_2\langle\langle x\rangle\rangle : \Pi(y) \quad \Pi(y) \neq \mathbf{0}}{(\Pi \setminus\setminus y) \uplus \Delta' \vdash^{(m_\Pi+m'',e_\Pi+e'')} E_1\langle\langle y\rangle\rangle[y{\leftarrow}E_2\langle\langle x\rangle\rangle] : O}\ \mathsf{ES}$$

and, by taking $\Gamma' \coloneqq (\Pi \setminus\setminus y) \uplus \Delta'$, $\Gamma_v \coloneqq \Delta_v$, $(m', e') = (m_\Pi + m'', e_\Pi + e'')$, then verify that

$$\Gamma = (\Pi \setminus\setminus y) \uplus \Delta = (\Pi \setminus\setminus y) \uplus (\Delta' \uplus \Delta_v) = \Gamma' \uplus \Gamma_v$$

and

$$(m, e) = (m_\Pi + m_\Delta, e_\Pi + e_\Delta) = (m_\Pi + (m'' + m_v), e_\Pi + (e'' + e_v - 1)) = (m' + m_v, e' + e_v - 1)$$

□

(Click here to go back to main chapter.)
The following is required to apply Lemma 5.4.7 (Linear Removal for CbNeed) in the proof of Proposition 5.4.8.2 (Quantitative Subject Expansion for CbNeed - exponential case) to obtain the right indices.

Lemma 13.2.32 (Merging multi types of CbNeed type derivations)**.**
Let $v \in \mathsf{Val}$. For any two type derivations

$$\Phi_N \triangleright_{CbNeed} \Gamma_N \vdash^{(m_N,e_N)} v : N$$
$$\Phi_O \triangleright_{\mathrm{CbV}} \Gamma_O \vdash^{(m_O,e_O)} v : O$$

there exsits type derivation

$$\Phi_{N \uplus O} \triangleright_{CbNeed} \Gamma_N \biguplus \Gamma_O \vdash^{(m_N+m_{m''},e_N+e_O)} v : N$$

Proof.
Trivial, given that the many typing rule is the only one that can derive a multi type on the right—*i.e.*, the derived type of the subject, in this case M—for values.

□

Proposition 13.2.33 (Quantitative Subject Expansion for CbNeed)**.**
Let $\Phi \triangleright_{CbNeed} \Gamma \vdash^{(m,e)} u : L$ be a type derivation such that $M \neq \mathbf{0}$.
1. Multiplicative*: If $t \to_{\mathsf{m}_{\mathrm{CbNeed}}} u$, then there exists a derivation*

$$\Psi \triangleright_{CbNeed} \Gamma \vdash^{(m+1,e)} t : M$$

2. Exponential*: If* $t \to_{\mathsf{e}_{\mathrm{CbNeed}}} u$*, then there exists a derivation*

$$\Psi \triangleright_{CbNeed} \Gamma \vdash^{(m,e+1)} t : M$$

Proof. (Click here to go back to main chapter.)

By induction on the derivation $t \to_{\mathsf{nd}} u$, with the root rules $\mapsto_{\mathsf{m}}$ and $\mapsto_{\mathsf{e}}$ as the base case, and the closure by CbNeed contexts of $\mapsto_{nd}$ as the inductive one.

- *Root step for* $\to_{\mathsf{m}}$: Let $t = S\langle\lambda x.s\rangle m \mapsto_{\mathsf{m}} S\langle s[x{\leftarrow}m]\rangle = u$, and proceed by induction on S:
  - Let $S \coloneqq \langle\cdot\rangle$. Then $t = (\lambda x.s)m$ and $u = s[x{\leftarrow}m]$, and so Φ has either ES or $\mathrm{ES}_{\mathrm{gc}}$ as its last typing rule.
    - If Φ is of the form
    $$\frac{\Gamma \vdash^{(m,e)} s : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e)} s[x{\leftarrow}m] : M}\ \mathrm{ES}_{\mathrm{gc}}$$
    then we can construct Φ' as follows
    $$\dfrac{\dfrac{\dfrac{\Gamma \vdash^{(m,e)} s : M}{\Gamma \vdash^{(m,e)} \lambda x.s : \mathbf{0} \multimap M}\ \mathsf{fun}}{\Gamma \vdash^{(m,e)} \lambda x.s : [\mathbf{0} \multimap M]}\ !}{\Gamma \vdash^{(m+1,e)} (\lambda x.s)m : M}\ \mathsf{app}_{\mathrm{gc}}$$
    - Let Φ be of the form
    $$\frac{x\colon O; \Pi \vdash^{(m',e')} s : M \quad \Delta \vdash^{(m'',e'')} m : O \quad O \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'')} s[x{\leftarrow}m] : M}\ \mathrm{ES}$$
    where $\Gamma = \Pi \biguplus \Delta$ and $(m,e) = (m'+m'', e'+e'')$.
    We can then construct Φ' as follows
    $$\dfrac{\dfrac{\dfrac{x\colon O; \Pi \vdash^{(m',e')} s : M}{\Pi \vdash^{(m',e')} \lambda x.s : O \multimap M}\ \mathsf{fun}}{\Pi \vdash^{(m',e')} \lambda x.s : [O \multimap M]}\ ! \qquad \Delta \vdash^{(m'',e'')} m : O}{\Pi \biguplus \Delta \vdash^{(m'+m''+1,e'+e'')} (\lambda x.s)m : M}\ \mathsf{app}_b$$
  - Let $S \coloneqq S'[y{\leftarrow}\tilde{t}]$. Then $t = (S'\langle\lambda x.s\rangle[y{\leftarrow}\tilde{t}])m$ and $u = S'\langle s[x{\leftarrow}m]\rangle[y{\leftarrow}\tilde{t}]$. Note that since we are working up to α-equivalence, $y \notin \mathsf{fv}(m)$. There are two possible last typing rules of Φ, namely ES and $\mathrm{ES}_{\mathrm{gc}}$.
    - Let Φ be of the form
    $$\frac{y\colon O; \Pi \vdash^{(m',e')} S'\langle s[x{\leftarrow}m]\rangle : M \quad \Delta \vdash^{(m'',e'')} \tilde{t} : O \quad O \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'')} S'\langle s[x{\leftarrow}m]\rangle[y{\leftarrow}\tilde{t}] : M}\ \mathrm{ES}$$
    where $\Gamma = \Pi \biguplus \Delta$ and $(m,e) = (m'+m'', e'+e'')$. We can then apply the *i.h.* on the leftmost premise to obtain a type derivation $\Phi'_{i.h.} \triangleright_{\mathrm{CbNeed}} y\colon O; \Pi \vdash^{(m'+1,e')} S'\langle\lambda x.s\rangle m : M$. We then analyze the two possibilities of the last typing rule in $\Phi'_{i.h.}$, namely app_b or $\mathsf{app}_{\mathrm{gc}}$

· Let $\Phi'_{i.h.}$ be of the form

$$\frac{y\colon O; \Pi_1 \vdash^{(m'_1,e'_1)} S'\langle\lambda x.s\rangle : [P \multimap M] \quad \Pi_2 \vdash^{(m'_2,e'_2)} m : P \quad P \neq \mathbf{0}}{y\colon O; \Pi \vdash^{(m'+1,e')} S'\langle\lambda x.s\rangle m : M}\ \mathsf{app}_b$$

where $(m', e') = (m'_1 + m'_2, e'_1 + e'_2)$, $\Pi = \Pi_1 \uplus \Pi_2$, and $y \notin \mathsf{dom}(\Pi_2)$, since otherwise Lemma 5.4.1 (Relevance of the CbNeed type system) would imply $y \in \mathsf{fv}(m)$.
We can then construct Φ' as follows

$$\frac{\dfrac{y\colon O; \Pi_1 \vdash^{(m'_1,e'_1)} S'\langle\lambda x.s\rangle : [P \multimap M] \quad \Delta \vdash^{(m'',e'')} \tilde{t} : O}{\Delta \uplus \Pi_1 \vdash^{(m'_1+m'',e'_1+e'')} S\langle\lambda x.s\rangle : [P \multimap M]}\ \mathrm{ES} \qquad \Pi_2 \vdash^{(m'_2,e'_2)} m : P}{\Delta \uplus \Pi_1 \uplus \Pi_2 \vdash^{(m'_1+m''+m'_2+1,e'_1+e''+e'_2)} S\langle\lambda x.s\rangle m : M}\ \mathsf{app}_b$$

· Let $\Phi'_{i.h.}$ be of the form

$$\frac{y\colon O; \Pi \vdash^{(m',e')} S'\langle\lambda x.s\rangle : [\mathbf{0} \multimap M]}{y\colon O; \Pi \vdash^{(m'+1,e')} S'\langle\lambda x.s\rangle m : M}\ \mathsf{app}_{\mathrm{gc}}$$

We can then construct Φ' as follows

$$\frac{\dfrac{y\colon O; \Pi \vdash^{(m',e')} S'\langle\lambda x.s\rangle : [\mathbf{0} \multimap M] \quad \Delta \vdash^{(m'',e'')} \tilde{t} : O}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'')} S\langle\lambda x.s\rangle : [\mathbf{0} \multimap M]}\ \mathsf{app}_{\mathrm{gc}}}{\Pi \uplus \Delta \vdash^{(m'+m''+1,e'+e'')} S\langle\lambda x.s\rangle m : M}\ \mathrm{ES}$$

∗ Let Φ be of the form

$$\frac{\Gamma \vdash^{(m,e)} S'\langle s[x{\leftarrow}m]\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e)} S'\langle s[x{\leftarrow}m]\rangle[y{\leftarrow}\tilde{t}] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

We then apply the *i.h.* on the leftmost premise to obtain a type derivation $\Phi'_{i.h.} \triangleright_{\mathrm{CbNeed}} \Gamma \vdash^{(m+1,e)} S'\langle\lambda x.s\rangle m : M$ for which there are two possible last typing rules, namely app_b and $\mathsf{app}_{\mathrm{gc}}$.

· Let $\Phi'_{i.h.}$ be of the form

$$\frac{\Pi \vdash^{(m',e')} S'\langle\lambda x.s\rangle : [O \multimap M] \quad \Delta \vdash^{(m'',e'')} m : O \quad O \neq \mathbf{0}}{\Gamma \vdash^{(m+1,e)} S'\langle\lambda x.s\rangle m : M}\ \mathsf{app}_b$$

where $\Gamma = \Pi \uplus \Delta$ and $(m, e) = (m' + m'', e' + e'')$.
We can then construct Φ' as follows

$$\frac{\dfrac{\Pi \vdash^{(m',e')} S'\langle\lambda x.s\rangle : [O \multimap M]}{\Pi \vdash^{(m',e')} S\langle\lambda x.s\rangle : [O \multimap M]}\ \mathrm{ES}_{\mathrm{gc}} \qquad \Delta \vdash^{(m'',e'')} m : O}{\Pi \uplus \Delta \vdash^{(m'+m''+1,e'+e'')} S\langle\lambda x.s\rangle m : M}\ \mathsf{app}_b$$

· Let $\Phi'_{i.h.}$ be of the form

$$\frac{\Gamma \vdash^{(m,e)} S'\langle\lambda x.s\rangle : [\mathbf{0} \multimap M]}{\Gamma \vdash^{(m+1,e)} S'\langle\lambda x.s\rangle m : M}\ \mathsf{app}_{\mathrm{gc}}$$

We can then construct Φ' as follows

$$\dfrac{\dfrac{\Gamma \vdash^{(m,e)} S'\langle \lambda x.s\rangle : [\mathbf{0} \multimap M]}{\Gamma \vdash^{(m,e)} S\langle \lambda x.s\rangle : [\mathbf{0} \multimap M]}\ \mathrm{ES_{gc}}}{\Gamma \vdash^{(m+1,e)} S\langle \lambda x.s\rangle m : M}\ \mathsf{app}_{\mathrm{gc}}$$

- *Root step for* $\to_{\mathsf{e}}$: Let $t = E\langle\!\langle x \rangle\!\rangle[x \leftarrow S\langle v\rangle] \mapsto_{\mathsf{e}} S\langle E\langle\!\langle v\rangle\!\rangle[x \leftarrow v]\rangle = u$, and proceed by induction on S:
  - Let $S := \langle\cdot\rangle$.Then $t = E\langle\!\langle x \rangle\!\rangle[x \leftarrow v] \mapsto_{\mathsf{e}} E\langle\!\langle v\rangle\!\rangle[x \leftarrow v] = u$, and so Φ has either ES or $\mathrm{ES_{gc}}$ as its last typing rule.
    * Let Φ be of the form

$$\dfrac{\Phi_{E\langle\!\langle v\rangle\!\rangle} \triangleright_{\mathrm{CbNeed}} \Gamma \vdash^{(m,e)} E\langle\!\langle v\rangle\!\rangle : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e)} E\langle\!\langle v\rangle\!\rangle[x \leftarrow v] : M}\ \mathrm{ES_{gc}}$$

    We apply Lemma 5.4.7 (Linear Removal for CbNeed) on $\Phi_{E\langle\!\langle v\rangle\!\rangle}$ to obtain type derivations $\Phi_v \triangleright_{\mathrm{CbNeed}} \Gamma_v \vdash^{(m_v,e_v)} v : O$ and $\Phi_{E\langle\!\langle x\rangle\!\rangle} \triangleright_{\mathrm{CbNeed}} \Gamma' \uplus \{x : O\} \vdash^{(m',e')} E\langle\!\langle x\rangle\!\rangle : M$ such that $\Gamma = \Gamma' \uplus \Gamma_v$ and $(m, e) = (m' + m_v, e' + e_v - 1)$. We can then construct Φ' with such derivations as follows

$$\dfrac{\Gamma' \uplus \{x : O\} \vdash^{(m',e')} E\langle\!\langle x\rangle\!\rangle : M \quad \Gamma_v \vdash^{(m_v,e_v)} v : O}{\Gamma' \uplus \Gamma_v \vdash^{(m'+m_v,e'+e_v)} E\langle\!\langle x\rangle\!\rangle[x \leftarrow v] : M}\ \mathrm{ES}$$

    In particular, note that $(m' + m_v, e' + e_v) = (m, e + 1)$.
    * Let Φ be of the form

$$\dfrac{x : O; \Pi \vdash^{(m',e')} E\langle\!\langle v\rangle\!\rangle : M \quad \Delta \vdash^{(m'',e'')} v : O \quad O \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'')} E\langle\!\langle v\rangle\!\rangle[x \leftarrow v] : M}\ \mathrm{ES}$$

    We can then apply Lemma 5.4.7 (Linear Removal for CbNeed) on the leftmost premise with respect to x to obtain a multi type P and type derivations $\Phi_v \triangleright_{\mathrm{CbNeed}} \Pi_v \vdash^{(m_v,e_v)} v : P$ and $\Phi_{E\langle\!\langle x\rangle\!\rangle} \triangleright_{\mathrm{CbNeed}} \Pi_{E\langle\!\langle x\rangle\!\rangle} \uplus \{x : P\} \vdash^{(m_{E\langle\!\langle x\rangle\!\rangle}, e_{E\langle\!\langle x\rangle\!\rangle})} E\langle\!\langle x\rangle\!\rangle : M$ such that $x : O; \Pi = \Pi_v \uplus \Pi_{E\langle\!\langle x\rangle\!\rangle}$ and $(m', e') = (m_{E\langle\!\langle x\rangle\!\rangle} + m_v, e_{E\langle\!\langle x\rangle\!\rangle} + e_v - 1)$. Note how Lemma 5.4.1 (Relevance of the CbNeed type system) implies that $x \notin \mathsf{dom}(\Pi_v)$—given that $x \notin \mathsf{fv}(v)$—and so $\Phi_{E\langle\!\langle x\rangle\!\rangle}$ can be rewritten as

$$\Phi_{E\langle\!\langle x\rangle\!\rangle} \triangleright_{\mathrm{CbNeed}} x : O \biguplus P; \Pi'_{E\langle\!\langle x\rangle\!\rangle} \vdash^{(m_{E\langle\!\langle x\rangle\!\rangle}, e_{E\langle\!\langle x\rangle\!\rangle})} E\langle\!\langle x\rangle\!\rangle : M$$

    where $\Pi_{E\langle\!\langle x\rangle\!\rangle} = x : O; \Pi'_{E\langle\!\langle x\rangle\!\rangle}$ and so $\Pi = \Pi'_{E\langle\!\langle x\rangle\!\rangle} \uplus \Pi_v$.
    Furthermore, we can apply Lemma 13.2.32 (Merging multi types of CbNeed type derivations) on $\Delta \vdash^{(m'',e'')} v : O$ and Φ_v to obtain a type derivation $\Phi_{O \uplus P} \triangleright_{\mathrm{CbNeed}} \Delta \uplus \Pi_v \vdash^{(m''+m_v,e''+e_v)} v : O \uplus P$.
    Finally, we can construct Φ' as follows

$$\dfrac{x : O \uplus P; \Pi'_{E\langle\!\langle x\rangle\!\rangle} \vdash^{(m_{E\langle\!\langle x\rangle\!\rangle}, e_{E\langle\!\langle x\rangle\!\rangle})} E\langle\!\langle x\rangle\!\rangle : M \quad \Delta \uplus \Pi_v \vdash^{(m''+m_v,e''+e_v)} v : O \uplus P}{\Pi'_{E\langle\!\langle x\rangle\!\rangle} \uplus \Delta \uplus \Pi_v \vdash^{(m_{E\langle\!\langle x\rangle\!\rangle}+m''+m_v, e_{E\langle\!\langle x\rangle\!\rangle}+e''+e_v)} E\langle\!\langle x\rangle\!\rangle[x \leftarrow v] : M}\ \mathrm{ES}$$

    Note that $(m_{E\langle\!\langle x\rangle\!\rangle} + m'' + m_v, e_{E\langle\!\langle x\rangle\!\rangle} + e'' + e_v) = (m' + m'', e' + 1 + e'') = (m, e + 1)$.

– Let $S := S[y{\leftarrow}m]$. Then $t = E\langle\!\langle x \rangle\!\rangle[x{\leftarrow}S'\langle v\rangle[y{\leftarrow}m]] \mapsto_{\mathsf{e}} S'\langle E\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle[y{\leftarrow}m] = u$. Note that $y \notin \mathsf{fv}(E\langle\!\langle x\rangle\!\rangle)$, since y is bound in $S\langle v\rangle$ and we are working up to α-equivalence. Then, Φ has either ES or $\mathrm{ES}_{\mathrm{gc}}$ as its last typing rule.

∗ Let Φ be of the form

$$\frac{\Gamma \vdash^{(m,e)} S'\langle E\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e)} S'\langle E\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle[y{\leftarrow}m] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

We can then apply the *i.h.* on the premise to obtain a type derivation $\Phi'_{i.h.} \triangleright_{\mathrm{CbNeed}} \Gamma \vdash^{(m,e+1)} E\langle\!\langle x\rangle\!\rangle[x{\leftarrow}S'\langle v\rangle] : M$. Moreover, note that $\Phi'_{i.h.}$ can only have ES as its last typing rule, by application of Lemma 5.4.1 (Relevance of the CbNeed type system). $\Phi'_{i.h.}$ is hence of the form

$$\frac{x\colon O; \Pi \vdash^{(m_{E\langle\!\langle x\rangle\!\rangle}, e_{E\langle\!\langle x\rangle\!\rangle})} E\langle\!\langle x\rangle\!\rangle : M \quad \Phi_{S'\langle v\rangle} \triangleright_{\mathrm{CbNeed}} \Delta \vdash^{(m_{S'\langle v\rangle}, e_{S'\langle v\rangle})} S'\langle v\rangle : O \quad O \neq \mathbf{0}}{\Gamma \vdash^{(m,e+1)} E\langle\!\langle x\rangle\!\rangle[x{\leftarrow}S'\langle v\rangle] : M}\ \mathrm{ES}$$

where $\Pi \biguplus \Delta = \Gamma$ and $(m, e+1) = (m_{E\langle\!\langle x\rangle\!\rangle} + m_{S'\langle v\rangle}, e_{E\langle\!\langle x\rangle\!\rangle} + e_{S'\langle v\rangle})$. Since $\Gamma(y) = \mathbf{0}$ then $\Delta(y) = \mathbf{0}$ and we can construct Φ' as follows

$$x\colon \frac{O; \Pi \vdash^{(m_{E\langle\!\langle x\rangle\!\rangle}, e_{E\langle\!\langle x\rangle\!\rangle})} E\langle\!\langle x\rangle\!\rangle : M \qquad \dfrac{\Delta \vdash^{(m_{S'\langle v\rangle}, e_{S'\langle v\rangle})} S'\langle v\rangle : O \quad \Delta(y) = \mathbf{0}}{\Delta \vdash^{(m_{S'\langle v\rangle}, e_{S'\langle v\rangle})} S'\langle v\rangle[y{\leftarrow}m] : O}\ \mathrm{ES}_{\mathrm{gc}}}{\Pi \biguplus \Delta \vdash^{(m_{E\langle\!\langle x\rangle\!\rangle}+m_{S'\langle v\rangle}, e_{E\langle\!\langle x\rangle\!\rangle}+e_{S'\langle v\rangle})} E\langle\!\langle x\rangle\!\rangle[x{\leftarrow}S'\langle v\rangle[y{\leftarrow}m]] : M}\ \mathrm{ES}$$

∗ Let Φ be of the form

$$\frac{y\colon O; \Pi \vdash^{(m_1,e_1)} S'\langle E\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle : M \quad \Delta \vdash^{(m_2,e_2)} m : O}{\Pi \biguplus \Delta \vdash^{(m_1+m_2, e_1+e_2)} S'\langle E\langle\!\langle v\rangle\!\rangle[x{\leftarrow}v]\rangle[y{\leftarrow}m] : M}\ \mathrm{ES}$$

where $\Pi \biguplus \Delta = \Gamma$ and $(m_1 + m_2, e_1 + e_2) = (m, e)$. We can then apply the *i.h.* on the leftmost premise to obtain a type derivation $\Phi'_{i.h.} \triangleright_{\mathrm{CbNeed}} y\colon O; \Pi \vdash^{(m_1,e_1+1)} E\langle\!\langle x\rangle\!\rangle[x{\leftarrow}S'\langle v\rangle] : M$ which has to have ES as its last typing rule—via Lemma 5.4.1 (Relevance of the CbNeed type system)—as follows

$$\frac{x\colon P; \Pi_1 \vdash^{(m_{1,1},e_{1,1})} E\langle\!\langle x\rangle\!\rangle : M \quad y\colon O; \Pi_2 \vdash^{(m_{1,2},e_{1,2})} S'\langle v\rangle : P}{y\colon O; \Pi_1 \biguplus \Pi_2 \vdash^{(m_{1,1}+m_{1,2}, e_{1,1}+e_{1,2})} E\langle\!\langle x\rangle\!\rangle[x{\leftarrow}S'\langle v\rangle] : M}\ \mathrm{ES}$$

where $\Pi_1 \biguplus \Pi_2 = \Pi$, $(m_{1,1} + m_{1,2}, e_{1,1} + e_{1,2}) = (m_1, e_1 + 1)$. We then construct Φ' as follows

$$x\colon \frac{P; \Pi_1 \vdash^{(m_{1,1},e_{1,1})} E\langle\!\langle x\rangle\!\rangle : M \qquad \dfrac{y\colon O; \Pi_2 \vdash^{(m_{1,2},e_{1,2})} S'\langle v\rangle : P \quad \Delta \vdash^{(m_2,e_2)} m : O}{\Pi_2 \biguplus \Delta \vdash^{(m_{1,2}+m_2, e_{1,2}+e_2)} S'\langle v\rangle[y{\leftarrow}m] : P}\ \mathrm{ES}}{\Pi_1 \biguplus \Pi_2 \biguplus \Delta \vdash^{(m_{1,1}+m_{1,2}+m_2, e_{1,1}+e_{1,2}+e_2)} E\langle\!\langle x\rangle\!\rangle[x{\leftarrow}S'\langle v\rangle[y{\leftarrow}m]] : M}\ \mathrm{ES}$$

Note that $\Pi_1 \biguplus \Pi_2 \biguplus \Delta = \Pi \biguplus \Delta = \Gamma$ and $(m_{1,1} + m_{1,2} + m_2, e_{1,1} + e_{1,2} + e_2) = (m_1 + m_2, e_1 + e_2) = (m, e+1)$.

- *Contextual closure*: We proceed by induction on the derivation of $t = E\langle t'\rangle \to_{\mathsf{nd}} E\langle u'\rangle = u$:
  – Let $E = \langle\cdot\rangle$. Then $t \mapsto_{\mathsf{m}} u$ or $t \mapsto_{\mathsf{e}} u$ and in either case the statement holds, as we have just proved.

– Let $E = E_1 s$. Then $\Phi \rhd_{\text{CbNeed}} \Gamma \vdash^{(m,e)} E_1\langle u'\rangle s : M$ and its last typing rule is either app_b or app_{gc}. We only cover the case where $E\langle t'\rangle \to_{\mathsf{m}} E\langle u'\rangle$ and Φ ends in rule app_b, leaving the rest of the (analogous) cases to the reader.
Let Φ be of the form

$$\frac{\Phi_{E_1\langle u\rangle} \rhd_{\text{CbNeed}} \Pi \vdash^{(m',e')} E_1\langle u'\rangle : [O \multimap M] \quad \Delta \vdash^{(m'',e'')} s : O}{\Pi \uplus \Delta \vdash^{(m'+m''+1,e'+e'')} E_1\langle u'\rangle s : M}\ \mathsf{app}_b$$

where $\Gamma = \Pi \uplus \Delta$, $(m' + m'' + 1, e' + e'') = (m, e)$, and $O \neq \mathbf{0}$.
We can then apply the *i.h.* on $\Phi_{E_1\langle u\rangle}$ to obtain a type derivation $\Phi'_{i.h.} \rhd_{\text{CbNeed}} \Pi \vdash^{(m+1,e)} E_1\langle t'\rangle : [O \multimap M]$ with which we construct Φ' as follows

$$\frac{\Pi \vdash^{(m+1,e)} E_1\langle t'\rangle : [O \multimap M] \quad \Delta \vdash^{(m'',e'')} s : O}{\Pi \uplus \Delta \vdash^{(m'+1+m''+1,e'+e'')} E_1\langle t'\rangle s : M}\ \mathsf{app}_b$$

Note that $(m' + 1 + m'' + 1, e' + 1 + e'') = (m + 1, e)$.
– Let $E = E_1[x \leftarrow s]$. Then $\Phi \rhd_{\text{CbNeed}} \Gamma \vdash^{(m,e)} E_1\langle u'\rangle[x \leftarrow s] : M$ and its last typing rule is either ES or ES_{gc}. We only cover the case where $E\langle t_1\rangle \to_{\mathsf{m}} E\langle u_1\rangle$ and Φ ends in rule ES, leaving the rest of the (analogous) cases to the reader.
Let Φ be of the form

$$\frac{\Phi_{E_1\langle u'\rangle} \rhd_{\text{CbNeed}} x : O; \Pi \vdash^{(m',e')} E_1\langle u'\rangle : M \quad \Delta \vdash^{(m'',e'')} s : O}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'')} E_1\langle u'\rangle[x \leftarrow s] : M}\ \text{ES}$$

where $\Pi \uplus \Delta = \Gamma$, $(m' + m'', e' + e'') = (m, e)$, and $O \neq \mathbf{0}$.
We can then apply the *i.h.* on $\Phi_{E_1\langle u'\rangle}$ to obtain a type derivation

$$\Phi'_{i.h.} \rhd_{\text{CbNeed}} x : O; \Pi \vdash^{(m'+1,e')} E_1\langle t'\rangle : M$$

with which Φ' goes as follows

$$\frac{x : O; \Pi \vdash^{(m'+1,e')} E_1\langle t'\rangle : M \quad \Delta \vdash^{(m'',e'')} s : O}{\Pi \uplus \Delta \vdash^{(m'+1+m'',e'+e'')} E_1\langle t'\rangle[x \leftarrow s] : M}\ \text{ES}$$

Note that $(m' + 1 + m'', e' + e'') = (m + 1, e)$.
– Let $E = E_1\langle\!\langle x\rangle\!\rangle[x \leftarrow E_2]$. We only consider the case where

$$t = E\langle t'\rangle = E_1\langle\!\langle x\rangle\!\rangle[x \leftarrow E_2\langle t'\rangle] \to_{\mathsf{m}} E_1\langle\!\langle x\rangle\!\rangle[x \leftarrow E_2\langle u'\rangle] = E\langle u'\rangle = u$$

leaving the (analogous) case when $t \to_{\mathsf{e}} u$ to the reader.
Therefore, Φ is of the form

$$\frac{\Pi \vdash^{(m',e')} E_1\langle\!\langle x\rangle\!\rangle : M \quad \Phi_{E_2\langle u'\rangle} \rhd_{\text{CbNeed}} \Delta \vdash^{(m'',e'')} E_2\langle u'\rangle : O}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'')} E_1\langle\!\langle x\rangle\!\rangle[x \leftarrow E_2\langle u'\rangle] : M}\ \text{ES}$$

where $\Pi \uplus \Delta = \Gamma$, $(m' + m'', e' + e'') = (m, e)$, and $O \neq \mathbf{0}$.
We can then apply the *i.h.* on $\Phi_{E_2\langle u'\rangle}$ to obtain a type derivation $\Phi'_{i.h.} \rhd_{\text{CbNeed}} \Delta \vdash^{(m''+1,e'')} E_2\langle t'\rangle : O$, with which we can finally construct Φ' as follows

$$\frac{\Pi \vdash^{(m',e')} E_1\langle\!\langle x\rangle\!\rangle : M \quad \Delta \vdash^{(m''+1,e'')} E_2\langle t'\rangle : O}{\Pi \uplus \Delta \vdash^{(m'+m''+1,e'+e'')} E_1\langle\!\langle x\rangle\!\rangle[x \leftarrow E_2\langle t'\rangle] : M}\ \text{ES}$$

Note that $(m' + m'' + 1, e' + e'') = (m + 1, e)$.

□

(Click here to go back to main chapter.)

Theorem 13.2.34 (Tight Completeness for CbNeed)**.**

Let $t \in \Lambda_{\mathsf{L}}$ be closed. If there exists $d : t \to^*_{\mathrm{CbNeed}} u$ for some u in $\to_{\mathrm{CbNeed}}$-normal form, then there exists a type derivation $\Phi \rhd_{CbNeed} \emptyset \vdash^{(|d|_{\mathsf{m}},|d|_{\mathsf{e}})} t : [\mathsf{norm}]$.

Proof. (Click here to go back to main chapter.)

By induction on the length $|d|$ of the reduction sequence $d\colon t \to_{\mathrm{CbNeed}} u$:

- *Base case*: Let $k := 0$. Then $t = u$ and t is in $\to_{\mathrm{CbNeed}}$-normal form. By Proposition 4.6.1.1 (Syntactic characaterization of closed normal forms - CbNeed), we have that $\mathsf{norm}(t)$. In addition, Proposition 5.4.6 (Tight typability of CbNeed-normal forms) yields tight type derivation $\Phi \rhd_{\mathrm{CbNeed}} \emptyset \vdash^{(0,0)} t : [\mathsf{norm}]$, which satisfies the statement—in particular, because $|d|_{\mathsf{m}} = |d|_{\mathsf{e}} = 0$.
- *Inductive case*: Let $k > 0$; *i.e.*, $t \to_{\mathrm{CbNeed}} s \to^{k-1}_{\mathrm{CbNeed}} u$. Let d' be the evaluation $s \to^{k-1}_{\mathrm{CbNeed}} u$. By *i.h.*, there exists tight type derivation

  $$\Psi \rhd_{\mathrm{CbNeed}} \emptyset \vdash^{(|d'|_{\mathsf{m}},|d'|_{\mathsf{e}})} s : [\mathsf{norm}]$$

  Case analysis on the kind of reduction step in $t \to_{\mathrm{CbNeed}} s$:
  - *Multiplicative step*: Let $t \to_{\mathsf{m}_{\mathrm{CbNeed}}} s$. By Proposition 5.4.8.1 (Quantitative Subject Expansion for CbNeed - Multiplicative), there exists (tight) type derivation

    $$\Psi' \rhd_{\mathrm{CbNeed}} \vdash^{(|d'|_{\mathsf{m}}+1,|d'|_{\mathsf{e}})} : [\mathsf{norm}]$$

    which satisfies the statement—in particular, because $|d|_{\mathsf{m}} = |d'|_{\mathsf{m}} + 1$ and $|d|_{\mathsf{e}} = |d'|_{\mathsf{e}}$.
  - *Exponential step*: Let $t \to_{\mathsf{e}_{\mathrm{CbNeed}}} s$. By Proposition 5.4.8.2 (Quantitative Subject Expansion for CbNeed - Exponential), there exists (tight) type derivation

    $$\Psi' \rhd_{\mathrm{CbNeed}} \vdash^{(|d'|_{\mathsf{m}},|d'|_{\mathsf{e}}+1)} : [\mathsf{norm}]$$

    which satisfies the statement—in particular, because $|d|_{\mathsf{m}} = |d'|_{\mathsf{m}}$ and $|d|_{\mathsf{e}} = |d'|_{\mathsf{e}} + 1$.

□

(Click here to go back to main chapter.)

13.2.7 CbNeed is as efficient as CbV

Corollary 13.2.35 (Correctness for CbNeed in the CbV type system)**.**

Let $t \in \Lambda_{\mathsf{L}}$ be closed and let $\Phi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m,e)} t : M$ be a type derivation. Then there exists $u \in \Lambda_{\mathsf{L}}$ such that

1. $\mathsf{norm}(u)$,
2. *there exists a reduction sequence $d : t \to^*_{\mathrm{CbNeed}} u$, and*
3. *$|d|_{\mathsf{m}} \leq m$ and $|d|_{\mathsf{e}} \leq e$.*

Proof. (Click here to go back to main chapter.)

By induction on $m+e$ and case analysis on whether t reduces or not. Note that if $\mathsf{norm}(t)$ then the statement holds with $u \coloneqq t$ and d the empty evaluation, so that $|d|_{\mathsf{m}} = 0 = |d|_{\mathsf{e}}$.

Otherwise $\neg\mathsf{norm}(t)$. Then, by Proposition 4.6.1.1 (Syntactic characterization of closed normal forms - CbNeed), there exists $s \in \Lambda_{\mathsf{L}}$ such that $t \to_{\mathrm{CbNeed}} s$.

As $t \to_{\mathrm{CbNeed}} s$ means either $t \to_{\mathsf{m}_{\mathrm{CbNeed}}} s$ or $t \to_{\mathsf{e}_{\mathrm{CbNeed}}} s$, that in turn means either $t \to_{\mathsf{m}_{\mathrm{CbV}}} s$ or $t \to_{\mathsf{e}_{\mathrm{CbV}}} s$, because, according to our presentations, the CbNeed rewriting relations are subsets of the CbV ones. Then, by Proposition 5.3.4 (Quantitative Subject Reduction for CbV), there exists $\Psi \rhd_{\mathrm{CbV}} \Gamma \vdash^{(m',e')} s : M$ such that:

- $m' \coloneqq m-1$ and $e' = e$ if $t \to_{\mathsf{m}_{\mathrm{CbNeed}}} s$,
- $m' \coloneqq m$ and $e' = e-1$ if $t \to_{\mathsf{e}_{\mathrm{CbNeed}}} s$.

By *i.h.* (since $m'+e' = m+e-1$), there is a term u such that $d' \colon s \to^*_{\mathrm{CbNeed}} u$ and $\mathsf{norm}(u)$, with $|d'|_{\mathsf{m}} \leq m'$ and $|d'|_{\mathsf{e}} \leq e'$.

The evaluation $d \colon t \to^*_{\mathrm{CbNeed}} u$ obtained by prefixing d' with the step $t \to_{\mathrm{CbNeed}} s$ verifies $|d|_{\mathsf{m}} \leq m$ and $|d|_{\mathsf{e}} \leq e$ because:

- if $t \to_{\mathsf{m}_{\mathrm{CbNeed}}} s$ then $|d|_{\mathsf{m}} = |d'|_{\mathsf{m}} + 1 \leq m' + 1 = m$ and $|d|_{\mathsf{e}} = |d'|_{\mathsf{e}} \leq e' = e$,
- if $t \to_{\mathsf{e}_{\mathrm{CbNeed}}} s$ then $|d|_{\mathsf{m}} = |d'|_{\mathsf{m}} \leq m' = m$ and $|d|_{\mathsf{e}} = |d'|_{\mathsf{e}} + 1 \leq e' + 1 = e$.

□

(Click here to go back to main chapter.)

Corollary 13.2.36 (CbNeed duplicates as wisely as CbV)**.**

Let $t, s \in \Lambda_{\mathsf{L}}$ be such that $d \colon t \to^*_{\mathrm{CbV}} s$ and $\mathsf{norm}_{\mathrm{CbV}}(s)$. Then there exist $u \in \Lambda_{\mathsf{L}}$ such that $\mathsf{norm}(u)$, and reduction sequence $d' \colon t \to^*_{\mathrm{CbNeed}} u$ such that $|d'|_{\mathsf{m}} \leq |d|_{\mathsf{m}}$ and $|d'|_{\mathsf{e}} \leq |d|_{\mathsf{e}}$.

Proof. (Click here to go back to main chapter.)

By Theorem 5.3.9 (Tight Completeness for CbV), there exists type derivation

$$\Phi \rhd_{\mathrm{CbV}} \emptyset \vdash^{(m,e)} t : \mathbf{0}$$

such that $m = |d|_{\mathsf{m}}$ and $e = |d|_{\mathsf{e}}$. Then Corollary 5.6.1 (Correctness for CbNeed in the CbV type system) yields reduction sequence $d' : t \to^*_{\mathrm{CbNeed}} u$ such that $\mathsf{norm}(u)$, $|d'|_{\mathsf{m}} \leq m = |d|_{\mathsf{m}}$ and $|d'|_{\mathsf{e}} \leq e = |d|_{\mathsf{e}}$.

□

(Click here to go back to main chapter.)

13.3 Proofs of Chapter 6 (Open CbNeed)

Lemma 13.3.1 (Unique derivation parameterization of open evaluation contexts)**.**
Let $P \in \mathcal{E}_{\mathcal{V}}$ and $P \in \mathcal{E}_{\mathcal{W}}$. Then $\mathcal{V} = \mathcal{W}$.

Proof. (Click here to go back to main chapter.)

By induction on the derivation of $P \in \mathcal{E}_{\mathcal{V}}$, proceeding by case analysis on the last applied derivation rule. The case where $P = (\mathcal{H}, \epsilon)$ for some term context $\mathcal{H}$ is trivial, since $\mathsf{nv}(.)$ is a *function*. The statement follows in the cases of derivation rules $\mathsf{O_I}$ and $\mathsf{O_{GC}}$ from the *i.h.*, and by definition of $\mathsf{nv}(.)$ in the case of $\mathsf{O_I}$. As an example, let us prove the case of $\mathsf{O_{HER}}$:

Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as follows:

$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad x \notin \mathcal{W}}{Q\langle x\rangle @[x \leftarrow \mathcal{H}] \in \mathcal{E}_{\mathcal{W} \cup \mathsf{nv}(\mathcal{H})}}\ \mathsf{O_{HER}}$$

where $P = Q\langle x\rangle @[x \leftarrow \mathcal{H}]$ and $\mathcal{V} = \mathcal{W} \cup \mathsf{nv}(\mathcal{H})$. Let us assume that $P = Q\langle x\rangle @[x \leftarrow \mathcal{H}] \in \mathcal{E}_{\mathcal{V}'}$ for some set of variables $\mathcal{V}'$. Given the shape of P, it could only be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{W}'} \quad x \notin \mathcal{W}'}{Q\langle x\rangle @[x \leftarrow \mathcal{H}] \in \mathcal{E}_{\mathcal{W}' \cup \mathsf{nv}(\mathcal{H})}}\ \mathsf{O_{HER}}$$

for some $\mathcal{W}'$ such that $\mathcal{V}' = \mathcal{W}' \cup \mathsf{nv}(\mathcal{H})$. The statement then follows by application of *i.h.* on $Q \in \mathcal{E}_{\mathcal{W}'}$, yielding that $\mathcal{W}' = \mathcal{W}$.

□

13.3.1 Characterizing Open CbNeed-normal forms.

Lemma 13.3.2 (Redex in non-normal terms)**.**
Let $t \in \Lambda$. Then, t is not a normal term if and only if there exist term context $\mathcal{H}$, and terms $\lambda x.u$ and s such that $t = \mathcal{H}\langle(\lambda x.u)s\rangle$.

Proof. (Click here to go back to main chapter.)

⇒ Let $t \in \Lambda$ be a non-normal term. We proceed by induction on the structure of t, noting that t is neither a variable nor a value.
Let $t = t_1 t_2$ such that t_1 is not inert or t_2 is not a normal term.
If t_1 is not a normal term, then by *i.h.* $t_1 = \mathcal{H}_1\langle(\lambda x_1.u_1)s_1\rangle$ and so by defining $\mathcal{H} \coloneqq \mathcal{H}_1 t_2$ we have that $t = \mathcal{H}\langle(\lambda x_1.u_1)s_1\rangle$.
Let t_1 be a normal term. If $t_1 = v_1$, then by defining $\mathcal{H} \coloneqq \langle\cdot\rangle$ we have that $t = \mathcal{H}\langle v_1 t_2\rangle$. Let $t_1 = i_1$ and t_2 be a non-normal term. Then by *i.h.* $t_2 = \mathcal{H}_2\langle(\lambda x_2.u_2)s_2\rangle$ and so by defining $\mathcal{H} \coloneqq i_1\mathcal{H}_2$ we have that $t = \mathcal{H}\langle(\lambda x_2.u_2)s_2\rangle$.

⇐ By induction on the structure of $\mathcal{H}$.
- If $\mathcal{H} = \langle\cdot\rangle$ then $t = (\lambda x.u)s$, which does not correspond to any term derivable from the grammar of inert terms.
- If $\mathcal{H} = \mathcal{J}m$, then $t = \mathcal{J}\langle(\lambda x.u)s\rangle m$. By *i.h.* $\mathcal{J}\langle(\lambda x.u)s\rangle$ is not a normal term, therefore nor is $\mathcal{J}\langle(\lambda x.u)s\rangle m = t$.
- If $\mathcal{H} = i\mathcal{J}$, then $t = i\mathcal{J}\langle(\lambda x.u)s\rangle$. By *i.h.* $\mathcal{J}\langle(\lambda x.u)s\rangle$ is not a normal term, therefore nor is $i\mathcal{J}\langle(\lambda x.u)s\rangle = t$.

□

(Click here to go back to main chapter.)

Before proving the characterization of Open CbNeed-normal forms by means of predicate $\mathsf{onorm}(.)$, we first need to give a series of properties. The first set concerns term contexts, and revolves around the notion of needed variables:

Lemma 13.3.3 (Rewriting: term contexts)**.**

1. Term contexts give needed variables*: For every $x \in \mathsf{Var}$ and term context $\mathcal{H}$, $x \in \mathsf{nv}(\mathcal{H}\langle x\rangle)$.*
2. Focusing inert terms on needed variables*: Let i be an inert term and $x \in \mathsf{nv}(i)$. Then there exists a term context $\mathcal{H}_x$ such that $x \notin \mathsf{nv}(\mathcal{H}_x) \subset \mathsf{nv}(i)$ and that $\mathcal{H}_x\langle x\rangle = i$.*
3. Focusing term contexts on needed variabless*: Let $x \in \mathsf{nv}(\mathcal{H})$. Then for every $t \in \Lambda$ there exists a term context $\mathcal{H}_t$ such that $x \notin \mathsf{nv}(\mathcal{H}_t) \subset \mathsf{nv}(\mathcal{H})$ and that $\mathcal{H}_t\langle x\rangle = \mathcal{H}\langle t\rangle$.*

Proof.

1. *Term contexts give needed variables*: By structural induction on $\mathcal{H}$:
   - *Context hole*: If $\mathcal{H} = \langle\cdot\rangle$, then it holds trivially.
   - *Left of an application*: Let $\mathcal{H} = \mathcal{J}t$. By *i.h.*, $x \in \mathsf{nv}(\mathcal{J}\langle x\rangle)$. Hence,
   $$x \in \mathsf{nv}(\mathcal{J}\langle x\rangle) = \mathsf{nv}(\mathcal{J}\langle x\rangle t) = \mathsf{nv}(\mathcal{H}\langle x\rangle)$$
   - *Right of an application*: Let $\mathcal{H} = i\mathcal{J}$. By *i.h.*, $x \in \mathsf{nv}(\mathcal{J}\langle x\rangle)$. Hence,
   $$x \in \mathsf{nv}(\mathcal{J}\langle x\rangle) \subseteq \mathsf{nv}(i) \cup \mathsf{nv}(\mathcal{J}\langle x\rangle) = \mathsf{nv}(\mathcal{H}\langle x\rangle)$$
2. *Focusing inert terms on needed variables*: By structural induction on i:
   - If $i = x$ then the statement holds by defining $\mathcal{H} \coloneqq \langle\cdot\rangle$.
   - Let $i = jf$. If $x \in \mathsf{nv}(j)$, then applying the *i.h.* on j yields term context $\mathcal{H}_j$ such that $x \notin \mathsf{nv}(\mathcal{H}_j) \subset \mathsf{nv}(j)$ and such that $\mathcal{H}_j\langle x\rangle = j$. Thus, the statement holds by defining $\mathcal{H} \coloneqq \mathcal{H}_j f$.
     If $x \notin \mathsf{nv}(j)$ then $x \in \mathsf{nv}(f)$, implying in turn that f is an inert term—since values have no needed variable. Then, applying the *i.h.* on f yields term context $\mathcal{H}_f$ such that $x \notin \mathsf{nv}(\mathcal{H}_f) \subset \mathsf{nv}(f)$ and $\mathcal{H}_f\langle x\rangle = f$. Thus, the statement holds by defining $\mathcal{H} \coloneqq j\mathcal{H}_f$.
3. *Focusing term contexts on needed variables*: Let $t \in \Lambda$. We proceed by structural induction on $\mathcal{H}$:
   - *Context hole*: This case is impossible, because $\mathsf{nv}(\langle\cdot\rangle) = \emptyset$.
   - *Left of an application*: Let $\mathcal{H} = \mathcal{J}s$. Then $x \in \mathsf{nv}(\mathcal{J}) = \mathsf{nv}(\mathcal{H})$. By *i.h.*, there exists term context $\mathcal{J}_t$ such that $x \notin \mathsf{nv}(\mathcal{J}_t) \subset \mathsf{nv}(\mathcal{J})$ and $\mathcal{J}_t\langle x\rangle = \mathcal{J}\langle t\rangle$. Therefore, the statement holds by defining $\mathcal{H}_t \coloneqq \mathcal{J}_t s$.
   - *Right of an application*: Let $\mathcal{H} = i\mathcal{J}$. Case analysis on whether $x \in \mathsf{nv}(i)$:
     - Let $x \in \mathsf{nv}(i)$. By Lemma 13.3.3.2 (Focusing inert terms on needed variables), there exists term context $\mathcal{J}_x$ such that $x \notin \mathsf{nv}(\mathcal{J}_x) \subset \mathsf{nv}(i)$ and $i = \mathcal{J}_x\langle x\rangle$. The statement follows by defining $\mathcal{H}_t \coloneqq \mathcal{J}_x\mathcal{J}\langle t\rangle$, verifying that $x \notin \mathsf{nv}(\mathcal{J}_x) = \mathsf{nv}(\mathcal{J}_x\mathcal{J}\langle t\rangle) = \mathsf{nv}(\mathcal{H}_t)$, that $\mathsf{nv}(\mathcal{H}_t) = \mathsf{nv}(\mathcal{J}_x\mathcal{J}\langle t\rangle) = \mathsf{nv}(\mathcal{J}_x) \subset \mathsf{nv}(i) \subseteq \mathsf{nv}(i) \cup \mathsf{nv}(\mathcal{J}) = \mathsf{nv}(\mathcal{H})$ and that $\mathcal{H}_t\langle x\rangle = \mathcal{J}_x\langle x\rangle\mathcal{J}\langle t\rangle = i\mathcal{J}\langle t\rangle = t$.
     - Let $x \notin \mathsf{nv}(i)$. Then $x \in \mathsf{nv}(\mathcal{J})$. By *i.h.*, there exists $\mathcal{J}_t$ such that $x \notin \mathsf{nv}(\mathcal{J}_t) \subset \mathsf{nv}(\mathcal{J})$ and that $\mathcal{J}_t\langle x\rangle = \mathcal{J}\langle t\rangle$. The statement follows by defining $\mathcal{H}_t \coloneqq i\mathcal{J}_t$, verifying that $x \notin \mathsf{nv}(i) \cup \mathsf{nv}(\mathcal{J}_t) = \mathsf{nv}(\mathcal{H}_t)$, that $\mathsf{nv}(\mathcal{H}_t) = \mathsf{nv}(i) \cup \mathsf{nv}(\mathcal{J}_t) \subset \mathsf{nv}(i) \cup \mathsf{nv}(\mathcal{J}) = \mathsf{nv}(\mathcal{H})$, and that $\mathcal{H}_t\langle x\rangle = i\mathcal{J}_t\langle x\rangle = i\mathcal{J}\langle t\rangle = \mathcal{H}\langle t\rangle$.

□

Next, we proceed to lift Lemma 13.3.3.1 and Lemma 13.3.3.3 to open evaluation contexts, as follows

Lemma 13.3.4 (Rewriting open evaluation contexts)**.**

1. Open evaluation contexts give needed variables*: Let $P \in \mathcal{E}_{\mathcal{V}}$ and $x \notin \mathsf{dom}(P)$. Then $x \in \mathsf{nv}(P\langle x\rangle)$.*
2. Focusing open evaluation contexts on needed variables*: Let $P \in \mathcal{E}_{\mathcal{V}}$ and $x \in \mathcal{V}$. Then for every $t \in \Lambda$ there exists an open evaluation context $P_t \in \mathcal{E}_{\mathcal{V}_t}$ such that $P_t\langle x\rangle = P\langle t\rangle$ and $x \notin \mathcal{V}_t \subset \mathcal{V}$.*

Proof.

1. By induction on the derivation of $P \in \mathcal{E}_{\mathcal{V}}$:
   - *Rule* $\mathsf{O_{AX}}$: Let $P = (\mathcal{H}, \epsilon) \in \mathcal{E}_{\mathsf{nv}(\mathcal{H})}$. By Lemma 13.3.3.1 (Term contexts give needed variables), $x \in \mathsf{nv}(\mathcal{H}\langle x\rangle)$. Hence, $x \in \mathsf{nv}(P\langle x\rangle) = \mathsf{nv}(\mathcal{H}\langle x\rangle)$.
   - *Rule* $\mathsf{O_{GC}}$: Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as follows
   $$\frac{Q \in \mathcal{E}_{\mathcal{V}} \quad y \notin \mathcal{V}}{Q@[y{\leftarrow}t] \in \mathcal{E}_{\mathcal{V}}}\ \mathsf{O_{GC}}$$
   where $P = Q@[y{\leftarrow}t]$. By hypothesis, $x \neq y$ and $x \notin \mathsf{dom}(Q\langle x\rangle)$. By *i.h.*, $x \in \mathsf{nv}(Q\langle x\rangle)$. Case analysis on whether $y \in \mathsf{nv}(Q\langle x\rangle)$:
     - Let $y \notin \mathsf{nv}(Q\langle x\rangle)$. The statement follows because then $\mathsf{nv}(P\langle x\rangle) = \mathsf{nv}(Q\langle x\rangle)$.
     - Let $y \in \mathsf{nv}(Q\langle x\rangle)$. Since $x \neq y$, then
     $$x \in (\mathsf{nv}(Q\langle x\rangle) \setminus \{y\}) \subseteq (\mathsf{nv}(Q\langle x\rangle) \setminus \{y\}) \cup \mathsf{nv}(t) = \mathsf{nv}(P\langle x\rangle)$$
   - *Rule* $\mathsf{O_I}$: Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as follows
   $$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \in \mathcal{W}}{Q@[y{\leftarrow}i] \in \mathcal{E}_{(\mathcal{W}\setminus\{y\})\cup\mathsf{nv}(i)}}\ \mathsf{O_I}$$
   where $P = Q@[y{\leftarrow}i]$ and $\mathcal{V} = (\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(i)$. By *i.h.*, we have that $x \in \mathsf{nv}(Q\langle x\rangle)$. Case analysis on whether $y \in \mathsf{nv}(Q\langle x\rangle)$:
     - Let $y \notin \mathsf{nv}(Q\langle x\rangle)$. The statement follows because then $x \in \mathsf{nv}(Q\langle x\rangle) = \mathsf{nv}(P\langle x\rangle)$.
     - Let $y \in \mathsf{nv}(Q\langle x\rangle)$. Since $x \neq y$, then
     $$x \in \mathsf{nv}(Q\langle x\rangle) \setminus \{y\} \subseteq (\mathsf{nv}(Q\langle x\rangle) \setminus \{y\}) \cup \mathsf{nv}(i) = \mathsf{nv}(P\langle x\rangle)$$
   - *Rule* $\mathsf{O_{HER}}$: Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as follows:
   $$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \notin \mathcal{W}}{Q\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{W}\cup\mathsf{nv}(\mathcal{H})}}\ \mathsf{O_{HER}}$$
   where $P = Q\langle y\rangle@[y{\leftarrow}\mathcal{H}]$ and $\mathcal{V} = \mathcal{W} \cup \mathsf{nv}(\mathcal{H})$. By α-conversion, we may safely assume that $y \notin \mathsf{dom}(Q)$. By *i.h.*, we have that $y \in \mathsf{nv}(Q\langle y\rangle)$. Moreover, Lemma 13.3.3.1 (Term contexts give needed variables) gives that $x \in \mathsf{nv}(\mathcal{H}\langle x\rangle)$. Hence,
   $$\begin{array}{rcl} x \in \mathsf{nv}(\mathcal{H}\langle x\rangle) & \subseteq & (\mathsf{nv}(Q\langle y\rangle) \setminus \{y\}) \cup \mathsf{nv}(\mathcal{H}\langle x\rangle) \\ & = & \mathsf{nv}((Q\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle x\rangle) \\ & = & \mathsf{nv}(P\langle x\rangle) \end{array}$$

2. Let $t \in \Lambda$. We proceed by induction on the derivation of $P \in \mathcal{E}_{\mathcal{V}}$:
   - *Rule* O_{AX}: Let $P = (\mathcal{H}, \epsilon) \in \mathcal{E}_{\mathsf{nv}(\mathcal{H})}$, with $x \in \mathsf{nv}(\mathcal{H})$. By Lemma 13.3.3.3 (Focusing term contexts on needed variables), there exists a term context $\mathcal{H}_t$ such that $x \notin \mathsf{nv}(\mathcal{H}_t) \subset \mathsf{nv}(\mathcal{H})$ and that $\mathcal{H}_t\langle x \rangle = \mathcal{H}\langle t \rangle$. Thus, the statement holds by defining $P_t \coloneqq (\mathcal{H}_t, \epsilon)$.
   - *Rule* O_{GC}: Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as follows

     $$\frac{Q \in \mathcal{E}_{\mathcal{V}} \quad y \notin \mathcal{V}}{Q@[y \leftarrow u] \in \mathcal{E}_{\mathcal{V}}}\ \mathsf{O}_{\mathsf{GC}}$$

     where $P = Q@[y \leftarrow u]$. By *i.h.* with respect to x and Q, there exists $Q_t \in \mathcal{E}_{\mathcal{V}_t}$ such that $x \notin \mathcal{V}_t \subset \mathcal{V}$ and $Q_t\langle x \rangle = Q\langle t \rangle$. We may then derive $P_t \in \mathcal{E}_{\mathcal{V}_t}$ as follows

     $$\frac{Q_t \in \mathcal{E}_{\mathcal{V}_t} \quad y \notin \mathcal{V}_t}{Q_t@[y \leftarrow u] \in \mathcal{E}_{\mathcal{V}_t}}\ \mathsf{O}_{\mathsf{GC}}$$

     where $P_t = Q_t@[y \leftarrow u]$, noting that $P_t\langle x \rangle = (Q_t\langle x \rangle)@[y \leftarrow u] = (Q\langle t \rangle)@[y \leftarrow u] = P\langle t \rangle$.
   - *Rule* O_{I}: Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as

     $$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \in \mathcal{W}}{Q@[y \leftarrow i] \in \mathcal{E}_{(\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(i)}}\ \mathsf{O}_{\mathsf{I}}$$

     where $P = Q@[y \leftarrow i]$ and $\mathcal{V} = (\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(i)$. Case analysis on whether $x \in (\mathcal{W} \setminus \{y\})$:
     - Let $x \in (\mathcal{W} \setminus \{y\})$. Note that then $x \notin y$. By application of the *i.h.* with respect to x and Q, there exists $Q_t \in \mathcal{E}_{\mathcal{W}_t}$ such that $x \notin \mathcal{W}_t \subset \mathcal{W}$ and $Q_t\langle x \rangle = Q\langle t \rangle$. Now, if $y \notin \mathcal{W}_t$, then the statement holds by deriving P_t as follows

       $$\frac{Q_t \in \mathcal{E}_{\mathcal{W}_t} \quad y \notin \mathcal{W}_t}{Q_t@[y \leftarrow i] \in \mathcal{E}_{\mathcal{W}_t}}\ \mathsf{O}_{\mathsf{GC}}$$

       where $P_t = Q_t@[y \leftarrow i]$. If $y \in \mathcal{W}_t$ instead, we proceed by case analysis on whether $x \in \mathsf{nv}(i)$:
       (a) Let $x \notin \mathsf{nv}(i)$. The statement then holds by deriving P_t as follows

       $$\frac{Q_t \in \mathcal{E}_{\mathcal{W}_t} \quad y \in \mathcal{W}_t}{Q_t@[y \leftarrow i] \in \mathcal{E}_{(\mathcal{W}_t \setminus \{x\}) \cup \mathsf{nv}(i)}}\ \mathsf{O}_{\mathsf{I}}$$

       where $P_t = Q_t@[y \leftarrow i]$. Thus, and since $x \neq y$ and $x \notin \mathcal{W}_t$, we have that

       $$x \notin \mathcal{W}_t \setminus \{y\} \subset \mathcal{W} \setminus \{y\} \subseteq (\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(i) = \mathcal{V}$$

       (b) Let $x \in \mathsf{nv}(i)$. By application of the *i.h.* with respect to y and Q_t, there exists $Q_x \in \mathcal{E}_{\mathcal{W}_x}$ such that $y \notin \mathcal{W}_x \subset \mathcal{W}_t \subset \mathcal{W}$, and that $Q_x\langle y \rangle = Q_t\langle x \rangle = Q\langle t \rangle$. Moreover, by Lemma 13.3.3.2 (Focusing inert terms on needed variables), there exists a term context $\mathcal{H}_x$ such that $x \notin \mathsf{nv}(\mathcal{H}_x) \subset i$ and that $\mathcal{H}_x\langle x \rangle = i$. Thus, we may derive $P_t \in \mathcal{E}_{\mathcal{V}_t}$

       $$\frac{Q_x \in \mathcal{E}_{\mathcal{W}_x} \quad y \notin \mathcal{W}_x}{Q_x\langle y \rangle@[y \leftarrow \mathcal{H}_x] \in \mathcal{E}_{\mathcal{W}_x \cup \mathsf{nv}(\mathcal{H}_x)}}\ \mathsf{O}_{\mathsf{HER}}$$

       where $P_t = Q_x\langle y \rangle@[y \leftarrow \mathcal{H}_x]$ and $\mathcal{V}_t = \mathcal{W}_x \cup \mathsf{nv}(\mathcal{H}_x)$.

– Let $x \notin (\mathcal{W} \setminus \{y\})$. Since $x \in \mathcal{V} = (\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(i)$, then it must be that $x \in \mathsf{nv}(i)$. By application of the *i.h.* with respect to y and Q, there exists $Q_t \in \mathcal{E}_{\mathcal{W}_t}$ such that $y \notin \mathcal{W}_t \subset \mathcal{W}$ and $Q_t\langle y\rangle = Q\langle t\rangle$. Moreover, by Lemma 13.3.3.2 (Focusing inert terms on needed variables), there exists term context $\mathcal{H}_x$ such that $x \notin \mathsf{nv}(\mathcal{H}_x) \subset \mathsf{nv}(i)$ and $\mathcal{H}_x\langle x\rangle = i$. Thus, we may derive $P_t \in \mathcal{E}_{\mathcal{V}_t}$ as follows

$$\frac{Q_y \in \mathcal{E}_{\mathcal{W}_y} \quad y \notin \mathcal{W}_y}{Q_y\langle y\rangle@[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\mathcal{W}_y \cup \mathsf{nv}(\mathcal{H}_x)}}\ \mathsf{O}_{\mathsf{HER}}$$

where $P_t = Q_y\langle y\rangle@[y{\leftarrow}\mathcal{H}_x]$, verifying in particular that

$$P_x\langle x\rangle = (Q_y\langle y\rangle)@[y{\leftarrow}\mathcal{H}_x\langle x\rangle] = Q\langle t\rangle@[y{\leftarrow}i] = P\langle t\rangle$$

- *Rule* $\mathsf{O}_{\mathsf{HER}}$: Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \notin \mathcal{W}}{Q\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{W} \cup \mathsf{nv}(\mathcal{H})}}\ \mathsf{O}_{\mathsf{HER}}$$

where $P = Q\langle y\rangle@[y{\leftarrow}\mathcal{H}]$ and $\mathcal{V} = \mathcal{W} \cup \mathsf{nv}(\mathcal{H})$. We do case analysis on whether $x \in \mathcal{W}$:

– Let $x \in \mathcal{W}$. By application of the *i.h.* with respect to x and Q, there exists $Q_y \in \mathcal{E}_{\mathcal{W}_y}$ such that $x \notin \mathcal{W}_y \subset \mathcal{W}$ and $Q_y\langle x\rangle = Q\langle y\rangle$. Since $y \notin \mathcal{W} \supset \mathcal{W}_x$, we can then derive $P_t \in \mathcal{E}_{\mathcal{V}_t}$ as follows

$$\frac{Q_y \in \mathcal{E}_{\mathcal{W}_y} \quad y \notin \mathcal{W}_y}{Q_y@[y{\leftarrow}\mathcal{H}\langle t\rangle] \in \mathcal{E}_{\mathcal{W}_y}}\ \mathsf{O}_{\mathsf{GC}}$$

where $P_t = Q_y@[y{\leftarrow}\mathcal{H}\langle t\rangle]$, verifying in particular that

$$P_t\langle x\rangle = (Q_y\langle x\rangle)@[y{\leftarrow}\mathcal{H}\langle t\rangle] = (Q\langle y\rangle)@[y{\leftarrow}\mathcal{H}\langle t\rangle] = P\langle t\rangle$$

– Let $x \notin \mathcal{W}$. Since $x \in \mathcal{V} = \mathcal{W} \cup \mathsf{nv}(\mathcal{H})$, then it must be that $x \in \mathsf{nv}(\mathcal{H})$. By Lemma 13.3.3.3 (Focusing term contexts on needed variables), there exists term context $\mathcal{H}_t$ such that $x \notin \mathsf{nv}(\mathcal{H}_t) \subset \mathsf{nv}(\mathcal{H})$ and $\mathcal{H}_t\langle x\rangle = \mathcal{H}\langle t\rangle$. Thus, we can derive $P_t \in \mathcal{E}_{\mathcal{V}_t}$ as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \notin \mathcal{W}}{Q\langle y\rangle@[y{\leftarrow}\mathcal{H}_t] \in \mathcal{E}_{\mathcal{W} \cup \mathsf{nv}(\mathcal{H}_t)}}\ \mathsf{O}_{\mathsf{HER}}$$

where $P_t = Q\langle y\rangle@[y{\leftarrow}\mathcal{H}_t]$.

□

Given that programs are structured inductively with respect to the length of their environments, we shall later see how the proof that the $\mathsf{onorm}(.)$ predicate characterizes the Open CbNeed-normal forms requires appending or removing ESs while preserving the $\to_{\mathsf{ond}}$-normality or the $\to_{\mathsf{ond}}$-reducibility of programs. More concretely, we need the following

Lemma 13.3.5 (Properties of Open CbNeed-normal forms and ESs)**.**

1. Removing ESs does not create $\to_{\mathsf{ond}}$-redexes: *if* $(t, E[x{\leftarrow}u])$ *is* $\to_{\mathsf{ond}}$*-normal, then* (t, E) *is* $\to_{\mathsf{ond}}$*-normal.*
2. Appending ESs that do not create $\to_{\mathsf{ond}}$-redexes: *let* (t, E) *be a* $\to_{\mathsf{ond}}$*-normal form such that if* $x \in \mathsf{nv}(t, E)$ *then* u *is inert. Then* $(t, E[x{\leftarrow}u])$ *is also in* $\to_{\mathsf{ond}}$*-normal form.*

Proof.

1. *Removing ESs does not create $\to_{\mathsf{ond}}$-redexes*: We prove the contrapositive statement, that is,

$$\begin{array}{c}\text{If } (t,E) \text{ is not in } \to_{\mathsf{ond}}\text{-normal form}\\ \text{then}\\ (t,E[x{\leftarrow}u]) \text{ is not in } \to_{\mathsf{ond}}\text{-normal form.}\end{array}$$

We proceed by case analysis on the kind of $\to_{\mathsf{ond}}$-reduction step in (t,E); namely, whether it is a multiplicative or exponential step:

- *Multiplicative step*: Let $(t,E) = P\langle(\lambda y.s)m\rangle \to_{\mathsf{om}} P\langle s,[y{\leftarrow}m]\rangle$, with $P \in \mathcal{E}_{\mathcal{V}}$. Case analysis on whether $x \in \mathcal{V}$:
  - Let $x \notin \mathcal{V}$. Then we may derive $P@[x{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}}$ via the application of the O_{GC} rule, getting that
  $$(t,E[x{\leftarrow}u]) = (P@[x{\leftarrow}u])\langle(\lambda y.s)m\rangle \to_{\mathsf{om}} (P@[x{\leftarrow}u])\langle s,[y{\leftarrow}m]\rangle$$
  - Let $x \in \mathcal{V}$. Case analysis on the shape of u:
    - Let u be inert. Then we can derive
    $$\frac{P \in \mathcal{E}_{\mathcal{V}} \quad x \in \mathcal{V}}{P@[x{\leftarrow}u] \in \mathcal{E}_{(\mathcal{V}\setminus\{x\})\cup\mathsf{nv}(u)}}\ \mathsf{O}_{\mathsf{I}}$$
    to obtain that
    $$(t,E[x{\leftarrow}u]) = (P@[x{\leftarrow}u])\langle(\lambda y.s)m\rangle \to_{\mathsf{om}} (P@[x{\leftarrow}u])\langle s,[y{\leftarrow}m]\rangle$$
    - Let u be a non-inert Λ-term. First, let $\tilde{s} \coloneqq (\lambda y.s)m$, and note that by an application of Lemma 13.3.4.2 (Focusing open evaluation contexts on needed variables) with respect to x and P, there exists $P_{\tilde{s}} \in \mathcal{E}_{\mathcal{V}_{\tilde{s}}}$ such that $x \notin \mathcal{V}_{\tilde{s}} \subset \mathcal{V}$, and that $P_{\tilde{s}}\langle x\rangle = P\langle\tilde{s}\rangle$. Two sub-cases:
    (a) Let $u \in \mathsf{Val}$. Then we can derive
    $$\frac{P_{\tilde{s}} \in \mathcal{E}_{\mathcal{V}_{\tilde{s}}} \quad x \notin \mathcal{V}_{\tilde{s}}}{P_{\tilde{s}}@[x{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}_{\tilde{s}}}}\ \mathsf{O}_{\mathsf{GC}}$$
    to obtain that
    $$\begin{array}{rcl}(t,E[x{\leftarrow}u]) & = & P\langle(\lambda y.s)m\rangle@[x{\leftarrow}u]\\ & = & P_{\tilde{s}}\langle x\rangle@[x{\leftarrow}u]\\ & = & (P_{\tilde{s}}@[x{\leftarrow}u])\langle x\rangle\\ & \to_{\mathsf{oe}} & (P_{\tilde{s}}@[x{\leftarrow}u])\langle u^{\alpha}\rangle\end{array}$$
    (b) Let $u \notin \mathsf{Val}$. Since u is also not an inert term, then we may conclude that u is a non-normal term. By Lemma 6.2.1 (Redex in non-normal terms), there exist term context $\mathcal{H}_u$ and $((\lambda z.\tilde{t})\tilde{u}) \in \Lambda$ such that $u = \mathcal{H}_u\langle(\lambda z.\tilde{t})\tilde{u}\rangle$. Then we can derive
    $$\frac{P_{\tilde{s}} \in \mathcal{V}_{\tilde{s}} \quad x \notin \mathcal{V}_{\tilde{s}}}{P_{\tilde{s}}\langle x\rangle@[x{\leftarrow}\mathcal{H}_u] \in \mathcal{E}_{\mathcal{V}_{\tilde{s}}\cup\mathsf{nv}(\mathcal{H}_u)}}\ \mathsf{O}_{\mathsf{HER}}$$
    to obtain that
    $$\begin{array}{rcl}(t,E[x{\leftarrow}u]) & = & P\langle(\lambda y.s)m\rangle@[x{\leftarrow}u]\\ & = & P_{\tilde{s}}\langle x\rangle@[x{\leftarrow}u]\\ & = & P_{\tilde{s}}\langle x\rangle@[x{\leftarrow}\mathcal{H}_u\langle(\lambda z.\tilde{t})\tilde{u}\rangle]\\ & = & (P_{\tilde{s}}\langle x\rangle@[x{\leftarrow}\mathcal{H}_u])\langle(\lambda z.\tilde{t})\tilde{u}\rangle\\ & \to_{\mathsf{om}} & (P_{\tilde{s}}\langle x\rangle@[x{\leftarrow}\mathcal{H}_u])\langle\tilde{t},[z{\leftarrow}\tilde{u}]\rangle\end{array}$$

- *Exponential step*: Let $(t, E) = P\langle y\rangle \to_{\mathsf{oe}} P\langle v^\alpha\rangle$, with $P \in \mathcal{E}_{\mathcal{V}}$, $y \in \mathsf{dom}(P)$, $P(y) = v$. By α-conversion, we may safely assume that $y \neq x$. We now consider two sub-cases, depending on whether $x \in \mathcal{V}$:
  - If $x \notin \mathcal{V}$, then we can derive
$$\frac{P \in \mathcal{E}_{\mathcal{V}} \quad x \notin \mathcal{V}}{P@[x{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}}}\ \mathsf{O_{GC}}$$
    to obtain that $(t, E[x{\leftarrow}u]) = (P@[x{\leftarrow}u])\langle y\rangle \to_{\mathsf{oe}} (P@[x{\leftarrow}u])\langle v^\alpha\rangle$.
  - Let $x \in \mathcal{V}$. Case analysis on the shape of u:
    * Let u be inert. Then we can derive
$$\frac{P \in \mathcal{E}_{\mathcal{V}} \quad x \in \mathcal{V}}{P@[x{\leftarrow}u] \in \mathcal{E}_{(\mathcal{V}\setminus\{x\})\cup\mathsf{nv}(u)}}\ \mathsf{O_I}$$
      to obtain that $(t, E[x{\leftarrow}u]) = (P@[x{\leftarrow}u])\langle y\rangle \to_{\mathsf{oe}} (P@[x{\leftarrow}u])\langle v^\alpha\rangle$.
    * Let u be a non-inert Λ-term. First, let $\tilde{s} := (\lambda y.s)m$, and note that by an application of Lemma 13.3.4.2 (Focusing open evaluation contexts on needed variables) with respect to x and P, there exists $P_{\tilde{s}} \in \mathcal{E}_{\mathcal{V}_{\tilde{s}}}$ such that $x \notin \mathcal{V}_{\tilde{s}} \subset \mathcal{V}$, and that $P_{\tilde{s}}\langle x\rangle = P\langle\tilde{s}\rangle$. There are two sub-cases, depending on the shape of u:
    (a) If $u \in \mathsf{Val}$, then we can derive
$$\frac{P_{\tilde{s}} \in \mathcal{E}_{\mathcal{V}_{\tilde{s}}} \quad x \notin \mathcal{V}_{\tilde{s}}}{P_{\tilde{s}}@[x{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}_{\tilde{s}}}}\ \mathsf{O_{GC}}$$
    to obtain that
$$\begin{array}{rcl}(t, E[x{\leftarrow}u]) & = & P\langle y\rangle@[x{\leftarrow}u] \\ & = & P_{\tilde{s}}\langle x\rangle@[x{\leftarrow}u] \\ & = & (P_{\tilde{s}}@[x{\leftarrow}u])\langle x\rangle \\ & \to_{\mathsf{oe}} & (P_{\tilde{s}}@[x{\leftarrow}u])\langle u^\alpha\rangle\end{array}$$
    (b) Let $u \in \mathsf{Val}$. Since u is also not an inert term, then we may conclude that u is a non-normal term. By Lemma 6.2.1 (Redex in non-normal terms), there exist term context $\mathcal{H}_u$ and $((\lambda z.s)m) \in \Lambda$ such that $\mathcal{H}_u\langle(\lambda z.s)m\rangle$. Then we can derive
$$\frac{P_{\tilde{s}} \in \mathcal{E}_{\mathcal{V}_{\tilde{s}}} \quad x \notin \mathcal{V}_{\tilde{s}}}{P_{\tilde{s}}\langle x\rangle@[x{\leftarrow}\mathcal{H}_u] \in \mathcal{E}_{\mathcal{V}_{\tilde{s}}\cup\mathsf{nv}(\mathcal{H}_u)}}\ \mathsf{O_{HER}}$$
    to obtain that
$$\begin{array}{rcl}(t, E[x{\leftarrow}u]) & = & P\langle y\rangle@[x{\leftarrow}u] \\ & = & P_{\tilde{s}}\langle x\rangle@[x{\leftarrow}u] \\ & = & P_{\tilde{s}}\langle x\rangle@[x{\leftarrow}\mathcal{H}_u\langle(\lambda z.s)m\rangle] \\ & = & (P_{\tilde{s}}\langle x\rangle@[x{\leftarrow}\mathcal{H}_u])\langle(\lambda z.s)m\rangle \\ & \to_{\mathsf{om}} & (P_{\tilde{s}}\langle x\rangle@[x{\leftarrow}\mathcal{H}_u])\langle s, [z{\leftarrow}m]\rangle\end{array}$$

2. *Adding ESs that do not create $\to_{\mathsf{ond}}$-redexes*: We prove the contrapositive statement; that is, that for all $x \in \mathsf{Var}$ and $u \in \Lambda$ satisfying that either $x \notin \mathsf{nv}(t, E)$ or that u is inert, the following holds

If $(t, E[x{\leftarrow}u])$ is not in $\to_{\mathsf{ond}}$-normal form
then
(t, E) is not in $\to_{\mathsf{ond}}$-normal form

Let us assume that $x \notin \mathsf{nv}(t, E)$ or that u is inert. We proceed by case analysis on the kind of $\to_{\mathsf{ond}}$-reduction step in $(t, E[x{\leftarrow}u])$; namely, whether it is a multiplicative or exponential step:

- Let $(t, E[x{\leftarrow}u]) = P\langle(\lambda x.s)m\rangle \to_{\mathsf{om}} P\langle s, [x{\leftarrow}m]\rangle$, with $P \in \mathcal{E}_{\mathcal{V}}$. We proceed by induction on the derivation of $P \in \mathcal{E}_{\mathcal{V}}$:
  - *Rule* $\mathsf{O_{AX}}$: This case is clearly not possible.
  - *Rule* $\mathsf{O_{GC}}$: Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as
    $$\frac{Q \in \mathcal{E}_{\mathcal{V}} \quad x \notin \mathcal{V}}{Q@[x{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}}}\ \mathsf{O_{GC}}$$
    with $P = Q@[x{\leftarrow}u]$. Then $(t, E) = Q\langle(\lambda x.s)m\rangle \to_{\mathsf{om}} Q\langle s, [x{\leftarrow}m]\rangle$.
  - *Rule* $\mathsf{O_I}$: Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as
    $$\frac{Q \in \mathcal{E}_{\mathcal{V}} \quad x \in \mathcal{V}}{Q@[x{\leftarrow}u] \in \mathcal{E}_{(\mathcal{W}\setminus\{x\})\cup\mathsf{nv}(u)}}\ \mathsf{O_I}$$
    with $P = Q@[x{\leftarrow}u]$, $\mathcal{V} = (\mathcal{W} \setminus \{x\}) \cup \mathsf{nv}(u)$ and u inert. Then $(t, E) = Q\langle(\lambda x.s)m\rangle \to_{\mathsf{om}} Q\langle s, [x{\leftarrow}m]\rangle$.
  - *Rule* $\mathsf{O_{HER}}$: Suppose $P \in \mathcal{E}_{\mathcal{V}}$ were derived as
    $$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad x \notin \mathcal{W}}{Q\langle x\rangle@[x{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{W}\cup\mathsf{nv}(\mathcal{H})}}\ \mathsf{O_{HER}}$$
    with $P = Q\langle x\rangle@[x{\leftarrow}\mathcal{H}]$, $\mathcal{V} = \mathcal{W} \cup \mathsf{nv}(\mathcal{H})$ and $u = \mathcal{H}\langle(\lambda x.s)m\rangle$. By Lemma 6.2.1 (Redex in non-normal terms), u could not be inert, thus implying by hypothesis that $x \notin \mathsf{nv}(t, E)$. However, by Lemma 13.3.4.1 (Open evaluation contexts give needed variables), we have that $x \in \mathsf{nv}(Q\langle x\rangle) = \mathsf{nv}(t, E)$. Hence, this case is absurd.
- Let $(t, E[x{\leftarrow}u]) = P\langle y\rangle \to_{\mathsf{oe}} P\langle v^{\alpha}\rangle$, with $P \in \mathcal{E}_{\mathcal{V}}$, $x \in \mathsf{dom}(P)$ and $P(x) = v$. We proceed by case analysis on the last rule applied in $P \in \mathcal{E}_{\mathcal{V}}$:
  - *Rule* $\mathsf{O_{AX}}$: This case is clearly not possible.
  - *Rule* $\mathsf{O_{GC}}$: Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as
    $$\frac{Q \in \mathcal{E}_{\mathcal{V}} \quad x \notin \mathcal{V}}{Q@[x{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}}}\ \mathsf{O_{GC}}$$
    with $P = Q@[x{\leftarrow}u]$.
    Suppose $y = x$, and so $u = v$. That is, suppose u is not inert, which would imply that $x \notin \mathsf{nv}(t, E)$. We would then have that $(t, E[x{\leftarrow}u]) = (t, E)@[x{\leftarrow}u] = Q\langle x\rangle@[x{\leftarrow}u]$; that is, $(t, E) = Q\langle x\rangle$. However, this would imply by Lemma 13.3.4.1 (Open evaluation contexts give needed variables) that $x \in \mathsf{nv}(t, E)$, which would contradict the hypothesis.
    Therefore, it must be that $y \neq x$, which implies that $y \in \mathsf{dom}(Q)$ and $Q(y) = v$, and so $(t, E) = Q\langle y\rangle \to_{\mathsf{oe}} Q\langle v^{\alpha}\rangle$.
  - *Rule* $\mathsf{O_I}$: Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as
    $$\frac{Q \in \mathcal{E}_{\mathcal{V}} \quad x \in \mathcal{V}}{Q@[x{\leftarrow}u] \in \mathcal{E}_{(\mathcal{V}\setminus\{x\})\cup\mathsf{nv}(u)}}\ \mathsf{O_I}$$
    with $P = Q@[x{\leftarrow}u]$ and u inert. This means that $u \notin \mathsf{Val}$, and so we have that $y \in \mathsf{dom}(Q)$ and $Q(y) = v$. That is, $(t, E) = Q\langle y\rangle \to_{\mathsf{oe}} Q\langle v^{\alpha}\rangle$.

– *Rule* $\mathsf{O_{HER}}$: Suppose $P \in \mathcal{E}_{\mathcal{V}}$ were derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{V}} \quad x \in \mathcal{V}}{Q\langle x\rangle @ [x \leftarrow \mathcal{H}] \in \mathcal{E}_{\mathcal{V} \cup \mathsf{nv}(\mathcal{H})}}\ \mathsf{O_{HER}}$$

with $P = Q\langle x\rangle @ [x \leftarrow \mathcal{H}]$ and $u = \mathcal{H}\langle y\rangle$. But then $y \notin \mathsf{dom}(P)$, and so this case is absurd.

□

In particular, note that the contrapositive of Lemma 13.3.5.1 and Lemma 13.3.5.2 show how $\to_{\mathsf{ond}}$-reducibility—*i.e.*, the property of not being a $\to_{\mathsf{ond}}$-normal form—is preserved when appending or removing an ES.

With Lemma 13.3.5 (Properties of Open CbNeed-normal forms), we can now lift Lemma 13.3.3.2 (Focusing inert terms on needed variables) to program contexts as follows

Lemma 13.3.6 (Open CbNeed-normal forms and needed variables)**.**

Let $p \in \mathcal{PR}$ be in $\to_{\mathsf{ond}}$-normal form and let $x \in \mathsf{nv}(p)$. Then there exists $P \in \mathcal{E}_{\mathcal{V}}$ such that $P\langle x\rangle = p$, $x \notin \mathcal{V} \subset \mathsf{nv}(p)$ and $x \notin \mathsf{dom}(P)$.

Proof.

Let $p = (t, E)$. We proceed by induction on the length of E.

- Let $E = \epsilon$. That is, $p = (t, \epsilon)$. First, let us suppose that t were a non-normal term. By Lemma 6.2.1 (Redex in non-normal terms), there would exist term context $\mathcal{H}$ and $((\lambda y.s)m) \in \Lambda$ such that $t = \mathcal{H}\langle(\lambda y.s)m\rangle$. But then $(\mathcal{H}, \epsilon) \in \mathcal{E}_{\mathsf{nv}(\mathcal{H})}$ and so $p = (\mathcal{H}, \epsilon)\langle(\lambda y.s)m\rangle \to_{\mathsf{om}} (\mathcal{H}, \epsilon)\langle s, [y \leftarrow m]\rangle$, which is absurd.
  Thus, t must be a normal term. Moreover, if $t \in \mathsf{Val}$, then $\mathsf{nv}(t)$ and the statement holds trivially.
  Finally, let t be an inert term and $x \in \mathsf{nv}(t)$. By Lemma 13.3.3.2 (Rewriting term contexts), there exists a term context $\mathcal{H}_x$ such that $x \notin \mathsf{nv}(\mathcal{H}_x) \subset \mathsf{nv}(t)$ and $\mathcal{H}_x\langle x\rangle = t$. The statement follows by taking $P_x \coloneqq (\mathcal{H}_x, \epsilon) \in \mathcal{E}_{\mathsf{nv}(\mathcal{H}_x)}$ verifies the statement, since $x \notin \mathsf{nv}(\mathcal{H}_x) \subset \mathsf{nv}(t) = \mathsf{nv}(p)$ and $(\mathcal{H}_x, \epsilon)\langle x\rangle = (\mathcal{H}_x\langle x\rangle, \epsilon) = (t, \epsilon) = p$.
- Let $E = E'[y \leftarrow u]$. Note that $x \neq y$, because $x \in \mathsf{nv}(p)$ and $\mathsf{nv}(p) \cap \mathsf{dom}(p) = \emptyset$—easily provable for every $p \in \mathcal{PR}$.
  We proceed by case analysis on whether $x \in \mathsf{nv}(p)$:
  – Let $x \in \mathsf{nv}(t, E')$. By Lemma 13.3.5.1 (Properties of Open CbNeed-normal forms and ESs), (t, E') is in $\to_{\mathsf{ond}}$-normal form, and so we can apply the *i.h.* on it to obtain $Q \in \mathcal{E}_{\mathcal{W}}$ such that $Q\langle x\rangle = (t, E')$, $x \notin \mathcal{W} \subset \mathsf{nv}(t, E')$, and $x \notin \mathsf{dom}(Q)$.
    Note that if $y \notin \mathcal{W}$ then we may derive $P \in \mathcal{E}_{\mathcal{V}}$ as

$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \notin \mathcal{W}}{Q @ [y \leftarrow u] \in \mathcal{E}_{\mathcal{W}}}\ \mathsf{O_{GC}}$$

    where $\mathcal{V} = \mathcal{W} \subset \mathsf{nv}(t, E') = \mathsf{nv}(t, E'[y \leftarrow u])$.
    Let us now consider the case where $y \in \mathcal{W}$. By Lemma 13.3.4.2 (Focusing open evaluation contexts on needed variables), there exists $Q_x \in \mathcal{E}_{\mathcal{W}_x}$ such that $y \notin \mathcal{W}_x \subset \mathcal{W}$ and $Q_x\langle y\rangle = Q\langle x\rangle = (t, E')$. We proceed by case analysis on the shape of u:
    * Let u be inert. Case analysis on whether $x \in \mathsf{nv}(u)$:

1. $x \in \mathsf{nv}(u)$: by Lemma 13.3.3.2 (Rewriting term contexts), there exists a term context $\mathcal{H}_x$ such that $x \notin \mathsf{nv}(\mathcal{H}_x) \subset \mathsf{nv}(u)$ and $\mathcal{H}_x\langle x\rangle = u$. We may then derive $P \in \mathcal{E}_{\mathcal{V}}$ as follows:

$$\frac{Q_x \in \mathcal{E}_{\mathcal{W}_x} \quad y \notin \mathcal{W}_x}{Q_x\langle y\rangle @[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\mathcal{W}_x \cup \mathsf{nv}(\mathcal{H}_x)}}\ \mathsf{O_{HER}}$$

where $\mathcal{V} = \mathcal{W}_x \cup \mathsf{nv}(\mathcal{H}_x)$. We thus verify that

$$\begin{array}{rcl} \mathcal{V} & = & \mathcal{W}_x \cup \mathsf{nv}(\mathcal{H}_x) \\ & \subset & \mathsf{nv}(t, E') \cup \mathsf{nv}(u) \\ & = & \mathsf{nv}(t, E'[y{\leftarrow}u]) \end{array}$$

Moreover, since $x \notin \mathcal{W}_x$ and $x \notin \mathsf{nv}(\mathcal{H}_x)$, then $x \notin \mathcal{V}$.

2. Let $x \notin \mathsf{nv}(u)$. Then, we may derive $P \in \mathcal{E}_{\mathcal{V}}$ as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \in \mathcal{W}}{Q@[y{\leftarrow}u] \in \mathcal{E}_{(\mathcal{W}\setminus\{y\}) \cup \mathsf{nv}(u)}}\ \mathsf{O_I}$$

where $\mathcal{V} = (\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(u)$, and verifying that

$$\begin{array}{rcl} \mathcal{V} & = & (\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(u) \\ & \subset & (\mathsf{nv}(t, E') \setminus \{y\}) \cup \mathsf{nv}(u) \\ & \subseteq & \mathsf{nv}(t, E'[y{\leftarrow}u]) \end{array}$$

Note that the last inequality holds regardlessly of whether $y \in \mathsf{nv}(t, E')$ or not. Moreover, since $x \notin \mathcal{W}$ and $x \notin \mathsf{nv}(u)$, then $x \notin \mathcal{V}$.

$*$ Suppose $u \in \mathsf{Val}$. Then we would be able to derive

$$\frac{Q_y \in \mathcal{E}_{\mathcal{W}_y} \quad y \notin \mathcal{W}_y}{Q_y@[y{\leftarrow}u] \in \mathcal{E}_{\mathcal{W}_y}}\ \mathsf{O_{GC}}$$

and thus obtain that $p = (Q@[y{\leftarrow}u])\langle y\rangle \to_{\mathsf{oe}} (Q@[y{\leftarrow}u])\langle u^{\alpha}\rangle$, which is absurd.

$*$ Suppose u were not a normal term. Then, by Lemma 6.2.1 (Redex in non-normal terms), there would exist term $\mathcal{H}$ and $((\lambda z.s)m) \in \Lambda$ such that $\mathcal{H}\langle(\lambda z.s)m\rangle = u$. But then we would be able to derive

$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \notin \mathcal{W}}{Q\langle y\rangle @[y{\leftarrow}\mathcal{H}_u] \in \mathcal{E}_{\mathcal{W} \cup \mathsf{nv}(\mathcal{H}_u)}}\ \mathsf{O_{HER}}$$

and thus obtain that

$$p = (Q\langle y\rangle @[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \to_{\mathsf{om}} (Q\langle y\rangle @[y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle$$

which is absurd.

– Let $x \notin \mathsf{nv}(t, E')$. Note that since $x \in \mathsf{nv}(t, E'[y{\leftarrow}u])$ by hypothesis, then it must be that $y \in \mathsf{nv}(t, E')$, thus having that $\mathsf{nv}(t, E'[y{\leftarrow}u]) = (\mathsf{nv}(t, E') \setminus \{y\}) \cup \mathsf{nv}(u)$ and so $x \in \mathsf{nv}(u)$.
By application of the *i.h.* on y and (t, E'), there exists $Q \in \mathcal{E}_{\mathcal{W}}$ such that $Q\langle y\rangle = (t, E')$, $y \notin \mathcal{W} \subset \mathsf{nv}(t, E')$, and $y \notin \mathsf{dom}(Q)$. We proceed by case analysis on the shape of u:

∗ Let u be inert. By Lemma 13.3.3.2 (Term contexts and needed variables), there exists term context $\mathcal{H}_x$ such that $x \notin \mathsf{nv}(\mathcal{H}_x) \subset \mathsf{nv}(u)$ and $\mathcal{H}_x\langle x\rangle = u$. We may then derive $P \in \mathcal{E}_{\mathcal{V}}$ as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \notin \mathcal{W}}{Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\mathcal{W}\cup\mathsf{nv}(\mathcal{H}_x)}}\ \mathsf{O}_{\mathsf{HER}}$$

where $P = Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}_x]$ and $\mathcal{V} = \mathcal{W} \cup \mathsf{nv}(\mathcal{H}_x)$, and verifying that

$$\begin{array}{rcl} \mathcal{V} & = & \mathcal{W} \cup \mathsf{nv}(\mathcal{H}_x) \\ & = & (\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(\mathcal{H}_x) \\ & \subset & (\mathsf{nv}(t, E') \setminus \{y\}) \cup \mathsf{nv}(u) \\ & = & \mathsf{nv}(t, E'[y{\leftarrow}u]) \end{array}$$

Moreover, since $x \notin \mathcal{W}$—because $\mathcal{W} \subset \mathsf{nv}(t, E')$ and $x \notin \mathsf{nv}(\mathcal{H}_x)$ either, then $x \notin \mathcal{V}$.

∗ Suppose $u \in \mathsf{Val}$. We would then be able to derive

$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \notin \mathcal{W}}{Q @ [y{\leftarrow}u] \in \mathcal{E}_{\mathcal{W}}}\ \mathsf{O}_{\mathsf{GC}}$$

and thus obtain that

$$p = (Q @ [y{\leftarrow}u])\langle y\rangle \to_{\mathsf{oe}} (Q @ [y{\leftarrow}u])\langle u^{\alpha}\rangle$$

which is absurd.

∗ Suppose u were not a normal term. By Lemma 6.2.1 (Redex in non-normal terms), there would exist term context $\mathcal{H}$ and $((\lambda z.s)m) \in \Lambda$ such that $\mathcal{H}\langle(\lambda z.s)m\rangle = u$. But then we would be able to derive

$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \notin \mathcal{W}}{Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{W}\cup\mathsf{nv}(\mathcal{H})}}\ \mathsf{O}_{\mathsf{HER}}$$

and thus obtain that

$$p = (Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \to_{\mathsf{om}} (Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle$$

which is absurd.

□

Proposition 13.3.7 (Syntactic characterization of Open CbNeed-normal forms)**.**
Let $p \in \mathcal{PR}$. Then p is in $\to_{\mathsf{ond}}$-normal form if and only if $\mathsf{onorm}(p)$.

Proof. (Click here to go back to main chapter.)

⇒ : Let $p = (t, E)$ be in $\to_{\mathsf{ond}}$-normal form. We prove that $\mathsf{onorm}(p)$ proceeding by induction on the length of E:

- Let $E = \epsilon$. That is, $p = (t, E) = (t, \epsilon)$. First, note that if we suppose that t is non-normal term, then an application of Lemma 6.2.1 (Redex in non-normal terms) would yield the existence of term context $\mathcal{H}$ and $(\lambda x.u)s \in \Lambda$ such that $t = \mathcal{H}\langle(\lambda x.u)s\rangle$ and so obtain that

$$\begin{array}{rcl} p & = & (t, E) \\ & = & (\mathcal{H}\langle(\lambda x.u)s\rangle, \epsilon) \\ & = & (\mathcal{H}, \epsilon)\langle(\lambda x.u)s\rangle \\ & \to_{\mathsf{ond}} & (\mathcal{H}, \epsilon)\langle u, [x{\leftarrow}s]\rangle \end{array}$$

which is absurd. Hence, t must be a normal term. Thus, if t is an inert term we may derive

$$\frac{}{\mathsf{inert}(t, \epsilon)}\ \mathsf{I_{AX}}$$

and if t is a value we may derive

$$\frac{}{\mathsf{abs}(t, \epsilon)}\ \mathsf{A_{AX}}$$

In either case, we conclude that $\mathsf{onorm}(t, \epsilon)$.

- Let $E = E'[y{\leftarrow}u]$. That is, $p = (t, E) = (t, E'[y{\leftarrow}u])$. Note that (t, E') is itself in $\rightarrow_{\mathsf{ond}}$-normal form—by Lemma 13.3.5.1 (Properties of Open CbNeed-normal forms and ESs). Hence, there exists a derivation of $\mathsf{onorm}(t, E')$—by *i.h.*
  First, note that if $\mathsf{abs}(t, E')$, then we may derive

$$\frac{\mathsf{abs}(t, E')}{\mathsf{abs}(t, E'[y{\leftarrow}u])}\ \mathsf{A_{GC}}$$

  and conclude that $\mathsf{onorm}(t, E')$.
  Let us now consider the case of $\mathsf{inert}(t, E')$, and prove the statement by case analysis on whether $y \in \mathsf{nv}(t, E')$:
  - Let $y \notin \mathsf{nv}(t, E')$. Thus, we may derive $\mathsf{inert}(t, E'[y{\leftarrow}u])$ as follows

$$\frac{\mathsf{inert}(t, E') \quad y \notin \mathsf{nv}(t, E')}{\mathsf{inert}(t, E'[y{\leftarrow}u])}\ \mathsf{I_{GC}}$$

    concluding that $\mathsf{onorm}(t, E)$.
  - Let $y \in \mathsf{nv}(t, E')$. By Lemma 13.3.6 (Open CbNeed-normal forms and needed variables), there exists an $P \in \mathcal{E}_{\mathcal{V}}$ such that $P\langle y\rangle = (t, E')$, $y \notin \mathcal{V} \subset \mathcal{W}$, and $y \notin \mathsf{dom}(P)$. We now proceed by case analysis on the shape of u:
    * If u is an inert term, then we may derive $\mathsf{inert}(t, E'[y{\leftarrow}u])$ as follows

$$\frac{\mathsf{inert}(t, E') \quad y \in \mathsf{nv}(t, E')}{\mathsf{inert}(t, E'[y{\leftarrow}u])}\ \mathsf{I_I}$$

      concluding that $\mathsf{onorm}(t, E'[y{\leftarrow}u])$.
    * Suppose u were a value. We would then be able to derive

$$\frac{P \in \mathcal{E}_{\mathcal{V}} \quad y \notin \mathcal{V}}{P@[y{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}}}\ \mathsf{O_{GC}}$$

      to obtain that

$$(t, E'[y{\leftarrow}u]) = P@[y{\leftarrow}u]\langle y\rangle \rightarrow_{\mathsf{oe}} P@[y{\leftarrow}u]\langle u^{\alpha}\rangle$$

      which is absurd.
    * Suppose u were not a normal term. Then there would exist term context $\mathcal{H}$ and $((\lambda z.s)m) \in \Lambda$ such that $\mathcal{H}\langle(\lambda z.s)m\rangle$—by Lemma 6.2.1 (Redex in non-normal terms). But then we would be able to derive

$$\frac{P \in \mathcal{E}_{\mathcal{V}} \quad y \notin \mathcal{V}}{P\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{V}\cup\mathsf{nv}(\mathcal{H})}}\ \mathsf{O_{HER}}$$

and obtain that

$$(t, E'[y{\leftarrow}u]) = (P\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \to_{\mathsf{om}} (P\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle$$

which is absurd.

$\Leftarrow$: Let $\mathsf{onorm}(p)$. Case analysis on the predicate from which we derive $\mathsf{onorm}(p)$:

- *Inert predicate*: We proceed by induction on the derivation of $\mathsf{inert}(p)$, proceeding by case analysis on the last applied derivation rule:
  - *Rule* $\mathsf{I_{AX}}$: Let $\mathsf{inert}(t, E)$ be derived as follows

    $$\frac{}{\mathsf{inert}(i, \epsilon)}\ \mathsf{I_{AX}}$$

    where $p = (i, \epsilon)$. Note that p is in $\to_{\mathsf{oe}}$-normal form because the environment is empty. By Lemma 6.2.1 (Redex in non-normal terms), we also have that p is in $\to_{\mathsf{om}}$-normal form. Therefore, p is in $\to_{\mathsf{ond}}$-normal form.
  - *Rule* $\mathsf{I_I}$: Let $\mathsf{inert}(t, E)$ be derived as follows

    $$\frac{\mathsf{inert}(t, E') \quad y \in \mathsf{nv}(t, E')}{\mathsf{inert}(t, E'[y{\leftarrow}u])}\ \mathsf{I_I}$$

    where $p = (t, E'[y{\leftarrow}u])$ and u is an inert term. By *i.h.*, (t, E') is in $\to_{\mathsf{ond}}$-normal form. Firstly, since $u \notin \mathsf{Val}$, note that there cannot exist $P@[y{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}}$—for some $\mathcal{V}$—with which we would have that

    $$p = (P@[y{\leftarrow}u])\langle y\rangle \to_{\mathsf{oe}} (P@[y{\leftarrow}u])\langle u^{\alpha}\rangle$$

    Said differently, p is in $\to_{\mathsf{oe}}$-normal form.
    Secondly, by Lemma 6.2.1 (Redex in non-normal terms), there cannot exist $P\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{V}}$—for some $\mathcal{V}$—such that $u = \mathcal{H}\langle(\lambda z.s)m\rangle$—for some $((\lambda z.s)m) \in \Lambda$—which would give that

    $$\begin{array}{rcl} p & = & P\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle(\lambda z.s)m\rangle] \\ & = & (P\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \\ & \to_{\mathsf{om}} & (P\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle \end{array}$$

    Said differently, p is also in $\to_{\mathsf{om}}$-normal form. We may thus conclude that p is in $\to_{\mathsf{ond}}$-normal form.
  - *Rule* $\mathsf{I_{GC}}$: Let $\mathsf{inert}(t, E)$ be derived as follows

    $$\frac{\mathsf{inert}(t, E') \quad y \notin \mathsf{nv}(t, E')}{\mathsf{inert}(t, E'[y{\leftarrow}u])}\ \mathsf{I_{GC}}$$

    where $p = (t, E'[y{\leftarrow}u])$. By *i.h.*, (t, E') is in $\to_{\mathsf{ond}}$-normal form. Let us separately consider the $\to_{\mathsf{ond}}$-reducibility of p:
    - Suppose there existed $P@[y{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}}$—for some $\mathcal{V}$—such that $(t, E'[y{\leftarrow}u]) = P\langle y\rangle@[y{\leftarrow}u]$ which would gives us that

      $$\begin{array}{rcl} p & = & P\langle y\rangle@[y{\leftarrow}u] \\ & = & (P@[y{\leftarrow}u])\langle y\rangle \\ & \to_{\mathsf{oe}} & (P@[y{\leftarrow}u])\langle u^{\alpha}\rangle \end{array}$$

      with $u \in \mathsf{Val}$. However, Lemma 13.3.4.2 (Open evaluation contexts give needed variables) would then give that $y \in \mathsf{nv}(t, E') = P\langle y\rangle$, which contradicts the hypothesis that $y \notin \mathsf{nv}(t, E')$. Therefore, this case is impossible.

∗ Suppose there existed $P\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{V}}$—for some $\mathcal{V}$—such that $(t, E'[y{\leftarrow}u]) = P\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle(\lambda z.s)m\rangle]$—*i.e.*, with $u = \mathcal{H}\langle(\lambda z.s)m\rangle$—which would give us that

$$\begin{array}{rcl} p & = & P\langle y\rangle@[y{\leftarrow}u] \\ & = & P@[y{\leftarrow}\mathcal{H}\langle(\lambda z.s)m\rangle] \\ & = & (P@[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \\ & \to_{\mathsf{oe}} & (P@[y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle \end{array}$$

However, Lemma 13.3.4.2 (Open evaluation contexts give needed variables) would then give that $y \in \mathsf{nv}(t, E') = P\langle y\rangle$, which contradicts the hypothesis that $y \notin \mathsf{nv}(t, E')$. Therefore, this case is also impossible.

We may thus conclude that $(t, E'[y{\leftarrow}u]) = p$ is in $\to_{\mathsf{ond}}$-normal form.

- *Abstraction predicate*: We proceed by induction on the derivation of $\mathsf{abs}(p)$, proceeding by case analysis on the last applied derivation rule:
  - *Rule* $\mathsf{A_{AX}}$: Let $\mathsf{abs}(t, E)$ be derived as follows

$$\frac{}{\mathsf{abs}(v, \epsilon)}\ \mathsf{A_{AX}}$$

where $p = (v, \epsilon)$. Note that p is in $\to_{\mathsf{oe}}$-normal form because the environment is empty. By Lemma 6.2.1 (Redex in non-normal terms), we also have that p is in $\to_{\mathsf{om}}$-normal form. Therefore, $(v, \epsilon) = p$ is in $\to_{\mathsf{ond}}$-normal form.

  - *Rule* $\mathsf{A_{GC}}$: Let $\mathsf{abs}(t, E)$ be derived as follows

$$\frac{\mathsf{abs}(t, E')}{\mathsf{abs}(t, E'[y{\leftarrow}u])}\ \mathsf{A_{GC}}$$

where $p = (t, E'[y{\leftarrow}u])$. By *i.h.*, (t, E') is in $\to_{\mathsf{ond}}$-normal form. Let us separately consider the $\to_{\mathsf{ond}}$-reducibility of p:

∗ Suppose there existed $P@[y{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}}$—for some $\mathcal{V}$—such that $(t, E'[y{\leftarrow}u]) = P\langle y\rangle@[y{\leftarrow}u]$ which would gives us that

$$\begin{array}{rcl} p & = & P\langle y\rangle@[y{\leftarrow}u] \\ & = & (P@[y{\leftarrow}u])\langle y\rangle \\ & \to_{\mathsf{oe}} & (P@[y{\leftarrow}u])\langle u^{\alpha}\rangle \end{array}$$

with $u \in \mathsf{Val}$. However, Lemma 13.3.4.2 (Open evaluation contexts give needed variables) would then give that $y \in \mathsf{nv}(t, E') = P\langle y\rangle$, which is absurd because abstraction programs do not have needed variables. Therefore, this case is impossible.

∗ Suppose there existed $P\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{V}}$—for some $\mathcal{V}$—such that $(t, E'[y{\leftarrow}u]) = P\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle(\lambda z.s)m\rangle]$—*i.e.*, with $u = \mathcal{H}\langle(\lambda z.s)m\rangle$—which would give us that

$$\begin{array}{rcl} p & = & P\langle y\rangle@[y{\leftarrow}u] \\ & = & P@[y{\leftarrow}\mathcal{H}\langle(\lambda z.s)m\rangle] \\ & = & (P@[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \\ & \to_{\mathsf{oe}} & (P@[y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle \end{array}$$

However, Lemma 13.3.4.2 (Open evaluation contexts give needed variables) would then give that $y \in \mathsf{nv}(t, E') = P\langle y\rangle$, which is absurd because abstraction programs do not have needed variables. Therefore, this case is also impossible.

□

(Click here to go back to main chapter.)

13.3.2 Determinism.

Lemma 13.3.8 (Unique decomposition of Λ-terms)**.**

Let $\mathcal{H}_1\langle t_1\rangle = \mathcal{H}_2\langle t_2\rangle$, with $\mathcal{H}_1, \mathcal{H}_2$ term contexts, let $S \supseteq (\mathsf{nv}(\mathcal{H}_1) \cup \mathsf{nv}(\mathcal{H}_2))$, and let t_i be a S-reduction place of $\mathcal{H}_i\langle t_i\rangle$, for $i = 1, 2$.

Then $t_1 = t_2$ and $\mathcal{H}_1 = \mathcal{H}_2$.

Proof. *(Click here to go back to main chapter.)*

By induction on $\mathcal{H}_1$. Cases:

- *Empty*: $\mathcal{H}_1 = \langle\cdot\rangle$. If t_1 is a multiplicative redex then $\mathcal{H}_2 = \langle\cdot\rangle$ and $t_2 = t_1$. The same is true if t_1 is a variable not in S.
- *Left of an application*: $\mathcal{H}_1 = \mathcal{J}_1 u$. Cases of $\mathcal{H}_2$:
  - *Empty*: $\mathcal{H}_2 = \langle\cdot\rangle$, then t_2 is a multiplicative redex, implying that $\mathcal{J}_1$ is empty and t_1 is a value, which is absurd. Therefore, this case is impossible.
  - *Left of an application*: $\mathcal{H}_2 = \mathcal{J}_2 u$. Then $\mathcal{J}_1\langle t_1\rangle = \mathcal{J}_2\langle t_2\rangle$ and t_i is a S-reduction places of $\mathcal{H}_i\langle t_i\rangle$ for $i = 1, 2$. By *i.h.*, $t_1 = t_2$ and $\mathcal{J}_1 = \mathcal{J}_2$, and then $\mathcal{H}_1 = \mathcal{H}_2$.
  - *Right of an application*: $\mathcal{H}_2 = i\mathcal{J}_2$. Then $\mathcal{J}_1\langle t_1\rangle = i$, and—by Lemma 6.2.1 (Redex in non-normal terms)—t_1 can only be a free variable x such that $x \notin S$. Note however, that $x \in \mathsf{fv}(i) \subseteq \mathsf{nv}(\mathcal{H}_2) \subseteq S$, which is absurd. Therefore, this case is impossible.
- *Right of an application*: $\mathcal{H}_1 = i\mathcal{J}_1$. Cases of $\mathcal{H}_2$:
  - *Empty*: $\mathcal{H}_2 = \langle\cdot\rangle$, then t_2 is an application that is not a multiplicative redex, absurd. Therefore, this case is impossible.
  - *Left of an application*: $\mathcal{H}_2 = \mathcal{J}_2 u$. This case is identical to the case where the hole of $\mathcal{H}_1$ is on the left of the application while the one of $\mathcal{H}_2$ is on the right, treated above.
  - *Right of an application*: $\mathcal{H}_2 = i\mathcal{J}_2$. Then $\mathcal{J}_1\langle t_1\rangle = \mathcal{J}_2\langle t_2\rangle$ and t_i is a S-reduction places of $\mathcal{H}_i\langle t_i\rangle$ for $i = 1, 2$. By *i.h.*, $t_1 = t_2$ and $\mathcal{J}_1 = \mathcal{J}_2$, and then $\mathcal{H}_1 = \mathcal{H}_2$.

□

(Click here to go back to main chapter.)

Theorem 13.3.9 (Unique decomposition of programs)**.**

Let $P_1\langle t_1\rangle = P_2\langle t_2\rangle$, with $P_1 \in \mathcal{E}_{\mathcal{V}_1}$, $P_2 \in \mathcal{E}_{\mathcal{V}_2}$, $S \supseteq (\mathcal{V}_1 \cup \mathcal{V}_2)$, and t_i be a S-reduction place of $P_i\langle t_i\rangle$ for $i = 1, 2$.

Then $t_1 = t_2$ and $P_1 = P_2$.

Proof. *(Click here to go back to main chapter.)*

By induction on P_1. Cases:

- Rule $\mathsf{O_{AX}}$: then $P_1 = (\mathcal{H}_1, \epsilon)$. Then $P_2 = (\mathcal{H}_2, \epsilon)$. By Lemma 6.3.1 (Unique decomposition of Λ-terms), we obtain $t_1 = t_2$ and $\mathcal{H}_1 = \mathcal{H}_2$, and then also $P_1 = P_2$.
- Rule $\mathsf{O_I}$: Let $P_1 \in \mathcal{E}_{\mathcal{V}_1}$ be derived as

$$\frac{Q_1 \in \mathcal{E}_{\mathcal{W}_1} \quad x \in \mathcal{W}_1}{Q_1@[x{\leftarrow}i] \in \mathcal{E}_{(\mathcal{W}_1 \setminus \{x\}) \cup \mathsf{nv}(i)}}\ \mathsf{O_I}$$

  with $P_1 = Q_1@[x{\leftarrow}i]$ with $\mathcal{V}_1 = (\mathcal{W}_1 \setminus \{x\}) \cup \mathsf{nv}(i)$. Cases of $P_2 \in \mathcal{E}_{\mathcal{V}_2}$:
  - Rule $\mathsf{O_{AX}}$: impossible.
  - Rule $\mathsf{O_I}$: Let $P_2 \in \mathcal{E}_{\mathcal{V}_2}$ be derived as

$$\frac{Q_2 \in \mathcal{E}_{\mathcal{W}_2} \quad x \in \mathcal{W}_2}{Q_2@[x{\leftarrow}i] \in \mathcal{E}_{(\mathcal{W}_2 \setminus \{x\}) \cup \mathsf{nv}(i)}}\ \mathsf{O_I}$$

with $P_2 = Q_2@[x{\leftarrow}i]$ with $\mathcal{V}_2 = (\mathcal{W}_2 \setminus \{x\}) \cup \mathsf{nv}(i)$.
Then $Q_1\langle t_1\rangle = Q_2\langle t_2\rangle$ and t_i is a S-reduction place of $Q_i\langle t_i\rangle$ for $i = 1, 2$. By *i.h.*, $t_1 = t_2$ and $Q_1 = Q_2$, and then $P_1 = P_2$.

– Rule $\mathsf{O_{GC}}$: Let $P_2 \in \mathcal{E}_{\mathcal{V}_2}$ be derived as

$$\frac{Q_2 \in \mathcal{E}_{\mathcal{W}_2} \quad x \notin \mathcal{W}_2}{Q_2@[x{\leftarrow}i] \in \mathcal{E}_{\mathcal{W}_2}}\ \mathsf{O_{GC}}$$

with $P_2 = Q_2@[x{\leftarrow}i]$ and $\mathcal{V}_2 = (\mathcal{W}_2 \setminus \{x\}) \cup \mathsf{nv}(i)$. Note that $t_i \neq x$ for $i = 1, 2$ because x is bound by P_1 and P_2 but not associated to a value. Note that $Q_1\langle t_1\rangle = Q_2\langle t_2\rangle$ and t_i is a S-reduction place of $Q_i\langle t_i\rangle$ for $i = 1, 2$. By *i.h.*, $t_1 = t_2$ and $Q_1 = Q_2$. This is absurd, because by Lemma 6.1.1 (Unique derivation parameterization of open evaluation contexts) we would have that $\mathcal{W}_1 = \mathcal{W}_2$, although we know that $x \in \mathcal{W}_1$ and $x \notin \mathcal{W}_2$. Therefore, this case is impossible.

– Rule $\mathsf{O_{HER}}$: Let $P_2 \in \mathcal{E}_{\mathcal{V}_2}$ be derived as

$$\frac{Q_2 \in \mathcal{E}_{\mathcal{W}_2} \quad x \notin \mathcal{W}_2}{Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{W}_2 \cup \mathsf{nv}(\mathcal{H})}}\ \mathsf{O_{HER}}$$

with $P_2 = Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}]$ and $\mathcal{V}_2 = \mathcal{W}_2 \cup \mathsf{nv}(\mathcal{H})$.
Note that t_2 cannot be a bound variable, because it occurs in the rightmost ES of the program, nor a β-redex, otherwise by Lemma 6.2.1 (Redex in non-normal terms) $\mathcal{H}\langle t_2\rangle = i$ would not be an inert term. Thus, it can only be that $t_2 = y \in \mathsf{Var}$ and $y \in \mathsf{nv}(i)$. But then $y \in \mathsf{nv}(i) \subseteq \mathcal{V}_1 \subseteq \mathcal{W}$, implying that t_2 is not a S-reduction place of $P_2\langle t_2\rangle$; absurd. Therefore, this case is impossible.

- Rule $\mathsf{O_{GC}}$: Let $P \in \mathcal{E}_{\mathcal{V}_1}$ be derived as

$$\frac{Q_1 \in \mathcal{E}_{\mathcal{V}_1} \quad x \notin \mathcal{V}_1}{Q_1@[x{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}_1}}\ \mathsf{O_{GC}}$$

with $P_1 = Q_1@[x{\leftarrow}t]$. Cases of $P_2 \in \mathcal{E}_{\mathcal{V}_2}$:

– Rule $\mathsf{O_{AX}}$: impossible.

– Rule $\mathsf{O_I}$: this case follows an identical but inversed analysis to that of the case of rule $\mathsf{O_I}$ for $P_1 \in \mathcal{E}_{\mathcal{V}_1}$ and $\mathsf{O_{GC}}$ for $P_2 \in \mathcal{E}_{\mathcal{V}_2}$, treated above.

– Rule $\mathsf{O_{GC}}$: Let $P_2 \in \mathcal{E}_{\mathcal{V}_2}$ be derived as

$$\frac{Q_2 \in \mathcal{E}_{\mathcal{V}_2} \quad x \notin \mathcal{V}_2}{Q_2@[x{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}_2}}\ \mathsf{O_{GC}}$$

where $P_2 = Q_2@[x{\leftarrow}t]$.
Then $Q_1\langle t_1\rangle = Q_2\langle t_2\rangle$ and t_i is a S-reduction place of $Q_i\langle t_i\rangle$ for $i = 1, 2$. By *i.h.*, $t_1 = t_2$ and $Q_1 = Q_2$, and then $P_1 = P_2$.

– Rule $\mathsf{O_{HER}}$: Let $P_2 \in \mathcal{E}_{\mathcal{V}_2}$ be derived as

$$\frac{Q_2 \in \mathcal{E}_{\mathcal{W}_2} \quad x \notin \mathcal{W}_2}{Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{W}_2 \cup \mathsf{nv}(\mathcal{H})}}\ \mathsf{O_{HER}}$$

where $P_2 = Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}]$ and $\mathcal{V}_2 = \mathcal{W}_2 \cup \mathsf{nv}(\mathcal{H})$.

Then $Q_1\langle t_1\rangle = Q_2\langle x\rangle$. Note we can assume that $x \notin \mathcal{W}$, since x was a bound name. Note also that $Q_1\langle t_1\rangle$ is a $\mathcal{W}$-reduction place and that $Q_2\langle x\rangle$ is also a $\mathcal{W}$-reduction place, because $x \notin \mathcal{W}$. By *i.h.*, $t_1 = x$, $Q_1 = Q_2$, and $\mathcal{V}'_1 = \mathcal{V}'_2$.
Now, let us look at the reduction places $t_1 = x$ and t_2. The fact that $t_1 = x$ is a reduction place in P_1 implies that u is a value. Then $\mathcal{H}$ is the empty context $\langle\cdot\rangle$ and t_2 is a value, which is absurd, since values cannot be reduction places. Therefore, this case is impossible.

- Rule $\mathsf{O_{HER}}$: Let $P_1 \in \mathcal{E}_{\mathcal{V}_1}$ be derived as

$$\frac{Q_1 \in \mathcal{E}_{\mathcal{W}_1} \quad x \notin \mathcal{W}_1}{Q_1\langle x\rangle@[x{\leftarrow}\mathcal{H}_1] \in \mathcal{E}_{\mathcal{W}_1 \cup \mathsf{nv}(\mathcal{H}_1)}}\ \mathsf{O_{HER}}$$

  where $P_1 = Q_1\langle x\rangle@[x{\leftarrow}\mathcal{H}_1]$ and $\mathcal{V}_1 = \mathcal{W}_1 \cup \mathsf{nv}(\mathcal{H}_1)$. Cases of $P_2 \in \mathcal{E}_{\mathcal{V}_2}$:
  - Rule $\mathsf{O_{AX}}$: impossible.
  - Rule $\mathsf{O_I}$: this case follows an identical but inversed analysis to that of the case of rule $\mathsf{O_I}$ for $P_1 \in \mathcal{E}_{\mathcal{V}_1}$ and rule $\mathsf{O_{HER}}$ for $P_2 \in \mathcal{E}_{\mathcal{V}_2}$, treated above.
  - Rule $\mathsf{O_{GC}}$: this case follows an identical but inversed analysis to that of the case of rule $\mathsf{O_{GC}}$ for $P_1 \in \mathcal{E}_{\mathcal{V}_1}$ and rule $\mathsf{O_{HER}}$ for $P_2 \in \mathcal{E}_{\mathcal{V}_2}$, treated above.
  - Rule $\mathsf{O_{HER}}$: Let $P_2 \in \mathcal{E}_{\mathcal{V}_2}$ be derived as

$$\frac{Q_2 \in \mathcal{E}_{\mathcal{W}_2} \quad x \notin \mathcal{W}_2}{Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}_2] \in \mathcal{E}_{\mathcal{W}_2 \cup \mathsf{nv}(\mathcal{H}_2)}}\ \mathsf{O_{HER}}$$

    where $P_2 = Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}_2]$ and $\mathcal{V}_2 = \mathcal{W}_2 \cup \mathsf{nv}(\mathcal{H}_2)$.
    Then $Q_1\langle x\rangle = Q_2\langle x\rangle$ and t_i is a S-reduction place of $Q_i\langle t_i\rangle$ for $i = 1, 2$. By *i.h.*, $Q_1 = Q_2$. Moreover, note that $\mathcal{H}_1\langle t_1\rangle = \mathcal{H}_2\langle t_2\rangle$ and that t_i is a S-reduction place of $\mathcal{H}_i\langle t_i\rangle$. Hence, by Lemma 6.3.1 (Unique decomposition of Λ-terms), we have that $t_1 = t_2$ and $\mathcal{H} = \mathcal{J}$. The latter allows us to finally conclude that $P_1 = P_2$.

□

(Click here to go back to main chapter.)

Corollary 13.3.10 (Determinism of Open CbNeed)**.**
Let $p, q, r \in \mathcal{PR}$. If $p \to_{\mathsf{ond}} q$ and $p \to_{\mathsf{ond}} r$, then $q = r$.

Proof. *(Click here to go back to main chapter.)*
Let $t_1, t_2 \in \Lambda$, $P_1 \in \mathcal{E}_{\mathcal{V}_1}$ and $P_2 \in \mathcal{E}_{\mathcal{V}_2}$ be such that $p = P_1\langle t_1\rangle = P_2\langle t_2\rangle$. Let moreover $S \coloneqq \mathcal{V}_1 \cup \mathcal{V}_2$, noting that t_1 is a S-reduction place of $P_1\langle t_1\rangle$ and t_2 is a reduction place of S-reduction place of $P_2\langle t_2\rangle$. By Theorem 6.3.2 (Unique decomposition of programs), $P_1 = P_2$ and $t_1 = t_2$. Therefore, $q = r$.

□

(Click here to go back to main chapter.)

13.4 Proofs of Chapter 7 (Multi types for Open CbNeed)

13.4.1 Open CbNeed correctness

Lemma 13.4.1 (Relevance of the Open CbNeed type system)**.**
Let e *be an expression and* $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} \mathsf{e} : M$ *be a type derivation. If* $x \notin \mathsf{fv}(\mathsf{e})$ *then* $x \notin \mathsf{dom}(\Gamma)$.

Proof. (Click here to go back to main chapter.)
The Open CbNeed multi type system being mostly based on the CbNeed one, this is trivially provable by induction on the number of typing rules applied in Φ, like we do for the CbNeed type system.

□

(Click here to go back to main chapter.)

Lemma 13.4.2 (Typing properties of normal terms)**.**
1. Values*: Let* $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} v : M$*. If* $M = [\mathsf{tight}]$*, then* $\mathsf{dom}(\Gamma) = \mathsf{nv}(v)$ *and* $(m, e, r) = (0, 0, |t|_{\mathsf{nd}})$.
2. Inert terms*: Let* $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} i : M$*, with* $\mathsf{inert}_\Gamma(\mathsf{nv}(i))$*. Then* $M = [\mathsf{inert}]_{i\in I}$ *with* $I \neq \emptyset$*,* $\mathsf{dom}(\Gamma) = \mathsf{nv}(i)$ *and* $(m, e, r) = (0, 0, |i|_{\mathsf{nd}})$.

Proof. (Click here to go back to main chapter.)
1. By a simple inspection of the typing rules, it must be that $M = [\mathsf{tight}]$ and that Φ is as follows:

$$\dfrac{\dfrac{}{\emptyset \vdash^{(0,0,0)} v : \mathsf{abs}}\ \mathsf{abs}}{\emptyset \vdash^{(0,0,0)} v : [\mathsf{abs}]}\ \mathsf{many}$$

2. By structural induction on t:
   - If $i = x \in \mathsf{Var}$, then Φ can only be of the form

$$\dfrac{M = \mathsf{Inert}}{x : M \vdash^{(0,0,0)} x : M}\ \mathsf{ax}_I$$

     The statement clearly holds.
   - Let $i = us$, for some inert Λ-term u and some normal term s. We proceed by case analysis on the last rule in Φ:
     – If Φ is of the form

$$\dfrac{\Psi \rhd_O \Pi \vdash^{(m',e',r')} u : [N \multimap M] \quad \Delta \vdash^{(m'',e'',r'')} s : N}{\Pi \biguplus \Delta \vdash^{(m'+m''+1,e'+e'',r'+r'')} us : M}\ \mathsf{app}$$

       then—by applying *i.h.* (1) on Ψ—we get that $[N \multimap M] = \mathsf{Inert}$, which is absurd.
     – Let Φ be of the form

$$\dfrac{\Psi \rhd_O \Pi \vdash^{(m',e',r')} u : [\mathsf{inert}]_{j\in J} \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} s : [\mathsf{tight}]}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e''+1,r'+r'')} us : [\mathsf{inert}]_{j\in J}}\ \mathsf{app}_i$$

       First of all, note that

$$\Psi \rhd_O \Pi \vdash^{(0,0,|u|_{\mathsf{nd}})} u : [\mathsf{inert}]_{j\in J}$$

with $\mathsf{dom}(\Pi) = \mathsf{nv}(u)$—by *i.h.* (1) on Ψ.
Now, if s is an inert Λ-term, then—by *i.h.* (1) on Θ—we have that

$$\Theta \rhd_O \Delta \vdash^{(0,0,|s|_{\mathsf{nd}})} s : [\mathsf{inert}]$$

with $\mathsf{dom}(\Delta) = \mathsf{nv}(s)$, and thus we can conclude that Φ is of the form

$$\frac{\Psi \rhd_O \Pi \vdash^{(0,0,|u|_{\mathsf{nd}})} u : [\mathsf{inert}]_{j\in J} \quad \Theta \rhd_O \Delta \vdash^{(0,0,|s|_{\mathsf{nd}})} s : [\mathsf{inert}]}{\Pi \biguplus \Delta \vdash^{(0,0,|u|_{\mathsf{nd}}+|s|_{\mathsf{nd}}+1)} us : [\mathsf{inert}]_{j\in J}}\ \mathsf{app}_i$$

with

$$\mathsf{dom}(\Pi \biguplus \Delta) = \mathsf{dom}(\Pi) \cup \mathsf{dom}(\Delta) = \mathsf{nv}(u) \cup \mathsf{nv}(s) = \mathsf{nv}(i)$$

If $s \in \mathsf{Val}$ instead, then Θ must be of the following form:

$$\Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} s : [\mathsf{abs}]$$

and so—by Lemma 7.1.2.1 (Typing properties of normal terms - values) on Θ—we have that

$$\Theta \rhd_O \emptyset \vdash^{(0,0,|s|_{\mathsf{nd}})} s : [\mathsf{abs}]$$

Thus, we can conclude that Φ is of the form

$$\frac{\Psi \rhd_O \Pi \vdash^{(0,0,|u|_{\mathsf{nd}})} u : [\mathsf{inert}]_{j\in J} \quad \Theta \rhd_O \emptyset \vdash^{(0,0,|s|_{\mathsf{nd}})} s : [\mathsf{abs}]}{\Pi \vdash^{(0,0,|u|_{\mathsf{nd}}+|s|_{\mathsf{nd}}+1)} us : [\mathsf{inert}]_{j\in J}}\ \mathsf{app}_i$$

with

$$\mathsf{dom}(\Pi) = \mathsf{nv}(u) = \mathsf{nv}(u) \cup \mathsf{nv}(s) = \mathsf{nv}(i)$$

– If Φ is of the form

$$\frac{\Psi \rhd_O \Pi \vdash^{(m',e',r')} u : [\mathbf{0} \multimap M]}{\Pi \vdash^{(m,e,r)} us : M}\ \mathsf{app}_{\mathrm{gc}}$$

then—by applying *i.h.* (1) on Ψ—we get that $[\mathbf{0} \multimap M] = \mathsf{Inert}$, which is absurd.

□

(Click here to go back to main chapter.)

Proposition 13.4.3 (Typing properties of Open CbNeed-normal forms)**.**
Let $p \in \mathcal{PR}$ be such that $\mathsf{onorm}(p)$, and let $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} p : M$ be a tight type derivation for it. Then $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$ and $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.

Proof. (Click here to go back to main chapter.)
We split the statement as follows

1. If $\mathsf{abs}(p)$ and $M = [\mathsf{tight}]$, then $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$ and $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$.
2. If $\mathsf{inert}(p)$ and $\mathsf{inert}_\Gamma(\mathsf{nv}(p))$, then $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$ and $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$.

Note that both items imply the statement. We prove them separately:

1. By induction on the derivation of $\mathsf{abs}(p)$.

- Let $p = (v, \epsilon)$, noting that $|(v,\epsilon)|_{\mathsf{nd}} = |v|_{\mathsf{nd}}$, and let

$$\frac{}{\mathsf{abs}(v,\epsilon)}\ \mathsf{A}_{\mathsf{Lift}}$$

Moreover, Φ must be of the form

$$\frac{\Phi' \rhd_O \Gamma \vdash^{(m,e,r)} v : [\mathsf{tight}]}{\Gamma \vdash^{(m,e,r)} (v,\epsilon) : [\mathsf{tight}]}\ \mathsf{Lift}$$

By Lemma 7.1.2.1 (Typing properties of normal terms - values) on Φ', we have that

$$\Phi' \rhd_O \Gamma \vdash^{(0,0,|v|_{\mathsf{nd}})} v : [\mathsf{abs}]$$

with $\mathsf{dom}(\Gamma) = \mathsf{nv}(v)$
Thus, Φ is of the form

$$\frac{\Phi' \rhd_O \Gamma \vdash^{(0,0,|v|_{\mathsf{nd}})} v : [\mathsf{abs}]}{\Gamma \vdash^{(0,0,|v|_{\mathsf{nd}})} (v,\epsilon) : [\mathsf{abs}]}\ \mathsf{Lift}$$

- Let $p = q@[x \leftarrow t]$ and let

$$\frac{\mathsf{abs}(q)}{\mathsf{abs}(q@[x \leftarrow t])}\ \mathsf{A}_{\mathsf{GC}}$$

We proceed by case analysis on the last rule in Φ
  - Suppose Φ is of the form

$$\frac{\Psi \rhd_O \Pi; x : N \vdash^{(m',e',r')} q : [\mathsf{tight}] \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} q@[x \leftarrow t] : [\mathsf{tight}]}\ \mathrm{ES}$$

However, application of the *i.h.* on Ψ would make us conclude that $\mathsf{dom}(\Pi; x : N) = \mathsf{nv}(q) = \emptyset$, and so we would have that $N = \mathbf{0}$, which is a contradiction.
  - Let Φ be of the form

$$\frac{\Phi' \rhd_O \Gamma \vdash^{(m,e,r)} q : [\mathsf{tight}] \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} q@[x \leftarrow t] : [\mathsf{tight}]}\ \mathrm{ES}_{\mathrm{gc}}$$

By *i.h.* on Φ', we have that

$$\Phi' \rhd_O \Gamma \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{tight}]$$

with $\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$. Since $\mathsf{nv}(q) = \emptyset$—provable by a simple induction on predicate $\mathsf{abs}(.)$—we can derive Φ as follows

$$\frac{\Phi' \rhd_O \emptyset \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{tight}] \quad \Gamma(x) = \mathbf{0}}{\emptyset \vdash^{(0,0,|q|_{\mathsf{nd}})} q@[x \leftarrow t] : [\mathsf{tight}]}\ \mathrm{ES}_{\mathrm{gc}}$$

Note that $\mathsf{nv}(q) = \emptyset = \mathsf{dom}(\Gamma)$—easily provable by induction on the derivation of $\mathsf{abs}(q)$. Thus, $|q@[x \leftarrow t]|_{\mathsf{nd}} = |q|_{\mathsf{nd}}$.

2. By induction on the derivation of $\mathsf{inert}(p)$.

- Let $p = (i, \epsilon)$, noting that $\mathsf{nv}(i, \epsilon) = \mathsf{nv}(i)$ and $|(i, \epsilon)|_{\mathsf{nd}} = |i|_{\mathsf{nd}}$, and let

$$\frac{}{\mathsf{inert}(i, \epsilon)}\ \mathsf{I}_{\mathsf{Lift}}$$

Moreover, Φ is of the form

$$\frac{\Phi' \rhd_O \Gamma \vdash^{(m,e,r)} i : M}{\Gamma \vdash^{(m,e,r)} (i, \epsilon) : M}\ \mathsf{Lift}$$

By Lemma 7.1.2.2 (Typing properties of normal terms - inert terms) on Φ', we have that

$$\Phi' \rhd_O \Gamma \vdash^{(0,0,|i|_{\mathsf{nd}})} i : M$$

with $\mathsf{dom}(\Gamma) = \mathsf{nv}(i)$. Thus, Φ is of the form

$$\frac{\Phi' \rhd_O \Gamma \vdash^{(0,0,|i|_{\mathsf{nd}})} i : M}{\Gamma \vdash^{(0,0,|i|_{\mathsf{nd}})} (i, \epsilon) : M}\ \mathsf{Lift}$$

- Let $p = q@[x \leftarrow i]$ and

$$\frac{\mathsf{inert}(q) \quad x \in \mathsf{nv}(q)}{\mathsf{inert}(q@[x \leftarrow i])}\ \mathsf{I}_{\mathsf{I}}$$

We proceed by case analysis on the last rule in Φ.

– Let Φ be of the form

$$\frac{\Psi \rhd_O \Pi; x : N \vdash^{(m',e',r')} q : M \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} i : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} q@[x \leftarrow i] : M}\ \mathrm{ES}$$

Note that if $x \notin \mathsf{nv}(q)$, then we can apply *i.h.* on Ψ to conclude that $N = \mathbf{0}$, which is absurd. Therefore, $\mathsf{nv}(q@[x \leftarrow u]) = (\mathsf{nv}(q) \setminus \{x\}) \cup \mathsf{nv}(u)$ and $|q@[x \leftarrow u]|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |u|_{\mathsf{nd}}$. Moreover, $\mathsf{inert}_{\Pi \uplus \Delta}(\mathsf{nv}(q@[x \leftarrow u]))$ implies that $\mathsf{inert}_{\Delta}(\mathsf{nv}(u))$. This allows us to apply Lemma 7.1.2.2 (Typing properties of normal terms - inert terms) on Θ to obtain that

$$\Theta \rhd_O \Delta \vdash^{(0,0,|i|_{\mathsf{nd}})} i : N$$

with $\mathsf{dom}(\Delta) = \mathsf{nv}(i)$ and $N \in \mathsf{Inert}$. This, in turn, allows us to apply *i.h.* on Ψ to obtain that

$$\Psi \rhd_O \Pi; x : N \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M$$

with $\mathsf{dom}(\Pi; x : N) = \mathsf{nv}(t, E)$. Finally, we see that Φ is of the form

$$\frac{\Pi; x : \mathsf{Inert} \vdash^{(0,0,|q|_{\mathsf{nd}})} q : \mathsf{Inert} \quad \Delta \vdash^{(0,0,|i|_{\mathsf{nd}})} i : \mathsf{Inert}}{\Pi \uplus \Delta \vdash^{(0,0,|q|_{\mathsf{nd}}+|i|_{\mathsf{nd}})} q@[x \leftarrow i] : \mathsf{Inert}}\ \mathrm{ES}$$

– Let Φ be of the form

$$\frac{\Phi' \rhd_O \Gamma \vdash^{(m,e,r)} (t, E) : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} (t, E[x \leftarrow u]) : M}\ \mathrm{ES}_{\mathrm{gc}}$$

By *i.h.* on Φ',

$$\Phi' \rhd_O \Gamma \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M$$

with $\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$. Since $x \notin \mathsf{dom}(\Gamma)$ then $x \notin \mathsf{nv}(q)$, thus having that $\mathsf{nv}(q@[x \leftarrow u]) = \mathsf{nv}(q)$ and $|q@[x \leftarrow u]|_{\mathsf{nd}} = |q|_{\mathsf{nd}}$. Finally, we see that Φ is of the form

$$\frac{\Gamma \vdash^{(0,0,|q|_{\mathsf{nd}})} q : \mathsf{Inert} \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(0,0,|q|_{\mathsf{nd}})} q@[x \leftarrow u] : \mathsf{Inert}}\ \mathrm{ES}_{\mathrm{gc}}$$

- Let $p = q@[x{\leftarrow}t]$ and
$$\frac{\mathsf{inert}(q) \quad x \notin \mathsf{nv}(q)}{\mathsf{inert}(q@[x{\leftarrow}t])}\ \mathsf{I_{GC}}$$
We proceed by case analysis on the last rule in Φ.
  - Suppose Φ is of the form
$$\frac{\Psi \rhd_O \Pi; x : N \vdash^{(m',e',r')} q : M \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} q@[x{\leftarrow}t] : M}\ \mathrm{ES}$$
Since $x \notin \mathsf{nv}(q)$, then $\mathsf{inert}_{\Pi \uplus \Delta}(\mathsf{nv}(q@[x{\leftarrow}t]))$ implies $\mathsf{inert}_{\Pi; x : N}(\mathsf{nv}(q))$, allowing us to apply *i.h.* on Ψ to conclude that $\mathsf{dom}(\Pi; x : N) = \mathsf{nv}(q)$, and thus $N = \mathbf{0}$, which is a contradiction.
  - Let Φ be of the form
$$\frac{\Phi' \rhd_O \Gamma \vdash^{(m,e,r)} q : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} q@[x{\leftarrow}u] : M}\ \mathrm{ES_{gc}}$$
By *i.h.*, we have that Φ' is of the form
$$\Phi' \rhd_O \Gamma \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M$$
with $\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$. Since $x \notin \mathsf{dom}(\Gamma)$, then $\mathsf{nv}(q@[x{\leftarrow}t]) = \mathsf{nv}(q)$ and $|q@[x{\leftarrow}t]|_{\mathsf{nd}} = |q|_{\mathsf{nd}}$. Finally, we see that Φ is of the form
$$\frac{\Phi' \rhd_O \Gamma \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(0,0,|q|_{\mathsf{nd}})} q@[x{\leftarrow}t] : M}\ \mathrm{ES_{gc}}$$

□

(Click here to go back to main chapter.)

Linear Substitution for Open CbNeed. Proving Linear Substitution for the Open CbNeed case requires extending the analysis of plugged variables and domain of type contexts in the CbNeed case—see Lemma 13.2.24 (Plugged variables and domain of type contexts) in Chapter 5 (Multi types for CbN, CbV and CbNeed)—to the Open CbNeed case. As a matter of fact, the required, analogous result can be obtained for the Open CbNeed type system assuming that the needed variables of the term context (resp. open evaluation context) are assigned to a multi type in Inert, as follows:

Lemma 13.4.4 (Plugged variables and domain of type contexts)**.**

1. *Let $\mathcal{H}$ be a term context such that $x \notin \mathsf{nv}(\mathcal{H})$, and let $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} \mathcal{H}\langle x\rangle : M$ be such that $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$.*
   Then $x \in \mathsf{dom}(\Gamma)$.
2. *Let $P \in \mathcal{E}_\mathcal{V}$ be such that $x \notin \mathcal{V}$ and $x \notin \mathsf{dom}(P)$, and let $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} P\langle x\rangle : M$ be such that $\mathsf{inert}_\Gamma(\mathcal{V})$.*
   Then $x \in \mathsf{dom}(\Gamma)$.

Proof.

1. By structural induction on $\mathcal{H}$:
   - Let $\mathcal{H} \coloneqq \langle\cdot\rangle$. Trivial, because $x \in \mathsf{dom}(\Gamma)$ by the fact that Φ is either an ax or an ax_I rule.
   - Let $\mathcal{H} \coloneqq \mathcal{J}t$. Note that $\mathsf{nv}(\mathcal{H}) = \mathsf{nv}(\mathcal{J})$. Since all three typing rules app, app_i and $\mathsf{app}_{\mathrm{gc}}$ are proven rather similarly, we only proceed to prove the statement for rule app:
     Let Φ be of the form
     $$\frac{\Psi \rhd_U \Pi \vdash^{(m',e',r')} \mathcal{J}\langle x\rangle : [N \multimap M] \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m''+1,e'+e'',r'+r'')} \mathcal{J}\langle x\rangle t : M}\ \mathsf{app}$$
     with $\Gamma = \Pi \uplus \Delta$. Since $\Pi \subseteq \Gamma$, then $\mathsf{inert}_\Pi(\mathsf{nv}(\mathcal{J}))$ and we may apply the *i.h.* on Ψ to obtain that $x \in \mathsf{dom}(\Pi) \subseteq \mathsf{dom}(\Gamma)$.
   - Let $\mathcal{H} \coloneqq i\mathcal{J}$. Note that $\mathsf{nv}(\mathcal{H}) = \mathsf{nv}(i) \cup \mathsf{nv}(\mathcal{J})$. We proceed by case analysis on the last typing rule in Φ.
     - Suppose Φ is of the form
       $$\frac{\Psi \rhd_U \Pi \vdash^{(m',e',r')} i : [N \multimap M] \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} \mathcal{J}\langle x\rangle : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m''+1,e'+e'',r'+r'')} i\mathcal{J}\langle x\rangle : M}\ \mathsf{app}$$
       with $\Gamma = \Pi \uplus \Delta$. However, $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$ implies that $\mathsf{inert}_\Gamma(\mathsf{nv}(i))$, then we would be able to apply Lemma 7.1.2.2 (Typing properties of normal terms - Inert terms) on Ψ to obtain that $[N \multimap M] \in \mathsf{Inert}$—which is also absurd. Therefore, this case is impossible.
     - The case where $\mathsf{app}_{\mathrm{gc}}$ is the last typing rule in Φ is ruled out as we did for the previous case.
     - Let Φ be of the form
       $$\frac{\Psi \rhd_U \Pi \vdash^{(m',e',r')} i : [\mathsf{inert}]_{j\in J} \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} \mathcal{J}\langle x\rangle : [\mathsf{tight}] \quad J \neq \emptyset}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e''+1,r'+r''+1)} i\mathcal{J}\langle x\rangle : [\mathsf{inert}]_{j\in J}}\ \mathsf{app}_i$$
       with $\Gamma = \Pi \uplus \Delta$. Since $\Delta \subseteq (\Pi \uplus \Delta)$, then we can apply the *i.h.* on Θ to obtain that $x \in \mathsf{dom}(\Delta) \subseteq \mathsf{dom}(\Gamma)$.
2. By induction on the derivation of $P \in \mathcal{E}_\mathcal{V}$:
   - Let $P \in \mathcal{E}_\mathcal{V}$ be derived as
     $$\frac{}{(\mathcal{H}, \epsilon) \in \mathcal{E}_{\mathsf{nv}(\mathcal{H})}}\ \mathsf{O}_{\mathsf{AX}}$$
     where $P = (\mathcal{H}, \epsilon)$ and $\mathcal{V} = \mathsf{nv}(\mathcal{H})$. Then Φ is of the form
     $$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} \mathcal{H}\langle x\rangle : M}{\Gamma \vdash^{(m,e,r)} (\mathcal{H}\langle x\rangle, \epsilon) : M}\ \mathsf{Lift}$$
     and the statement is proven by Lemma 13.6.8.1 (Plugged variables and domain of type contexts) on Ψ.
   - Let $P \in \mathcal{E}_\mathcal{V}$ be derived as
     $$\frac{Q \in \mathcal{E}_\mathcal{W} \quad z \in \mathcal{W}}{Q@[z{\leftarrow}i] \in \mathcal{E}_{(\mathcal{W}\setminus\{z\})\cup\mathsf{nv}(i)}}\ \mathsf{O}_{\mathsf{I}}$$
     with $P = Q@[z{\leftarrow}i]$ and $\mathcal{V} = (\mathcal{W} \setminus \{z\}) \cup \mathsf{nv}(i)$. Note that $x \neq z$ because $x \notin \mathsf{dom}(P)$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; z : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} i : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle@[z{\leftarrow}i] : M}\ \text{ES}$$

There are two cases concerning z and $\mathcal{W}$. On the one hand, if $z \notin \mathcal{W}$, then $\mathsf{inert}_\Gamma(\mathcal{V})$ implies that $\mathsf{inert}_{\Pi;z:N}(\mathcal{W})$, thus allowing us to apply the *i.h.* on Ψ to conclude that $x \in \mathsf{dom}(\Pi) \subseteq \mathsf{dom}(\Gamma)$. On the other hand, if $z \in \mathcal{W}$, and since $\mathsf{inert}_\Gamma(\mathcal{V})$ implies that $\mathsf{inert}_\Delta(\mathsf{nv}(i))$, then we can obtain that $N \in \mathsf{Inert}$—by Lemma 7.1.2.2 (Typing properties of normal terms - Inert terms). Therefore, $\mathsf{inert}_{\Pi;z:N}(\mathcal{W})$, and so we are able to apply the *i.h.* on Ψ to obtain that $x \in \mathsf{dom}(\Pi) \subseteq \mathsf{dom}(\Gamma)$.

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle x\rangle : M \quad \Gamma(z) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle x\rangle@[z{\leftarrow}i^+] : M}\ \text{ES}_{\text{gc}}$$

Since $z \notin \mathsf{dom}(\Gamma)$, then $\mathsf{inert}_\Gamma(\mathcal{V})$ implies that $\mathsf{inert}_\Gamma(\mathcal{W})$. By *i.h.* on Ψ, we conclude that $x \in \mathsf{dom}(\Gamma)$.

- Let $P \in \mathcal{E}_\mathcal{V}$ be derived as

$$\frac{Q \in \mathcal{E}_\mathcal{V} \quad x \notin \mathcal{V}}{Q@[z{\leftarrow}t] \in \mathcal{E}_\mathcal{V}}\ \mathsf{O}_{\mathsf{GC}}$$

with $P = Q@[z{\leftarrow}t]$. Note that $x \neq z$ because $x \notin \mathsf{dom}(P)$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; z : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle@[z{\leftarrow}t] : M}\ \text{ES}$$

Note that since $z \notin \mathcal{W}$, then $\mathsf{inert}_{\Pi \uplus \Delta}(\mathcal{V})$ implies that $\mathsf{inert}_{\Pi;z:N}(\mathcal{W})$. We may then apply the *i.h.* on Ψ to prove the statement, concluding that $x \in \mathsf{dom}(\Pi) \subseteq \mathsf{dom}(\Gamma)$.

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle x\rangle : M \quad \Gamma(z) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle x\rangle@[z{\leftarrow}t] : M}\ \text{ES}_{\text{gc}}$$

The statement holds by *i.h.* on Ψ.

- Let $P \in \mathcal{E}_\mathcal{V}$ be derived as

$$\frac{Q \in \mathcal{E}_\mathcal{W} \quad z \notin \mathcal{W}}{Q\langle z\rangle@[z{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{W}\cup\mathsf{nv}(\mathcal{H})}}\ \mathsf{O}_{\mathsf{HER}}$$

where $P = Q\langle z\rangle@[z{\leftarrow}\mathcal{H}]$ and $\mathcal{V} = \mathcal{W} \cup \mathsf{nv}(\mathcal{H})$. We may safely assume that $x \neq z$—by α-conversion. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be derived as

$$\frac{\Psi \rhd_U \Pi; z : N \vdash^{(m',e',r')} Q\langle z\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} \mathcal{H}\langle x\rangle : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle z\rangle@[z{\leftarrow}\mathcal{H}\langle x\rangle] : M}\ \text{ES}$$

By Lemma 13.4.4.1 (Plugged variables and domain of type contexts), we then have that $x \in \mathsf{dom}(\Delta) \subseteq \mathsf{dom}(\Gamma)$.

– Suppose Φ is of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle z\rangle : M \quad \Gamma(z) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle z\rangle @ [z \leftarrow \mathcal{H}\langle x\rangle] : M}\ \mathrm{ES_{gc}}$$

However, $\mathsf{inert}_\Gamma(\mathcal{V})$ would imply that $\mathsf{inert}_\Gamma(\mathcal{W})$. Thus, we would be able to apply the *i.h.* on Ψ to conclude that $z \in \mathsf{dom}(\Gamma)$; this makes this case absurd.

□

Lemma 13.4.5 (Linear Substitution for Open CbNeed)**.**
Let $x \in \mathsf{Var}$ and $v \in \mathsf{Val}$ such that $x \notin \mathsf{fv}(v)$.

1. *Let $\mathcal{H}$ be such that $x \notin \mathsf{nv}(\mathcal{H})$, and let*

$$\Phi_{\mathcal{H}\langle x\rangle} \rhd_O \Gamma; x : M \vdash^{(m,e,r)} \mathcal{H}\langle x\rangle : N$$

be such that $M, N \neq \mathbf{0}$ and $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$.
Then there exists splitting $M = M_1 \uplus M_2$, with $M_1 \neq \mathbf{0}$, such that for every

$$\Psi \rhd_O \Pi \vdash^{(m',e',r')} v : M_1$$

there exists

$$\Phi_{\mathcal{H}\langle v\rangle} \rhd_O \left(\Gamma \biguplus \Pi\right); x : M_2 \vdash^{(m+m',e+e'-1,r+r')} \mathcal{H}\langle v\rangle : N$$

2. *Let $P \in \mathcal{E}_\mathcal{V}$ be such that $x \notin \mathsf{dom}(P)$ and $x \notin \mathcal{V}$, and let*

$$\Phi_{P\langle x\rangle} \rhd_O \Gamma; x : M \vdash^{(m,e,r)} P\langle x\rangle : N$$

be such that $M, N \neq \mathbf{0}$ and $\mathsf{inert}_\Gamma(\mathcal{V})$.
Then there exists splitting $M = M_1 \uplus M_2$, with $M_1 \neq \mathbf{0}$, such that for every type derivation

$$\Psi \rhd_O \Pi \vdash^{(m',e',r')} v : M_1$$

there exists

$$\Phi_{P\langle v\rangle} \rhd_O \left(\Gamma \biguplus \Pi\right); x : M_2 \vdash^{(m+m',e+e'-1,r+r')} P\langle v\rangle : N$$

Proof. (Click here to go back to main chapter.)

1. By induction on the shape of $\mathcal{H}$:
   - Let $\mathcal{H} = \langle\cdot\rangle$. Then $\Phi_{\mathcal{H}\langle x\rangle}$ is either an ax or an ax_I rule. If $\Phi_{\mathcal{H}\langle x\rangle}$ is of the form

$$\frac{M \neq \mathsf{Inert}}{x : M \vdash^{(0,1,0)} x : M}\ \mathsf{ax}$$

   then $\Gamma = \emptyset$ and $N = M$. We then define $M_1 \coloneqq M$ and $M_2 \coloneqq \mathbf{0}$, noting that for every $\Psi \rhd_O \Pi \vdash^{(m',e',r')} v : M_1$ the statement holds by taking $\Phi_{\mathcal{H}\langle v\rangle} \coloneqq \Psi$. In particular, note that

$$(m', e', r') = (m + m', e + e' - 1, r + r')$$

   If $\Phi_{\mathcal{H}\langle x\rangle}$ is of the form

$$\frac{M = \mathsf{Inert}}{x : M \vdash^{(0,0,0)} x : M}\ \mathsf{ax}_I$$

   then $\Gamma = \emptyset$ and $N = M = \mathsf{Inert}$. Note then that the statement holds trivially for any splitting of M, considering that values are not typable with inert—verifiable by a simple inspection of the typing rules.

- Let $\mathcal{H} = \mathcal{J}t$. We consider the different cases corresponding to the last typing rule in $\Phi_{\mathcal{H}\langle x\rangle}$.
  - Let $\Phi_{\mathcal{H}\langle x\rangle}$ be of the form

$$\frac{\Delta; x : M_\Delta \vdash^{(m_1'', e_1'', r_1'')} \mathcal{J}\langle x\rangle : [O \multimap N] \quad \Sigma; x : M_\Sigma \vdash^{(m_2'', e_2'', r_2'')} t : O \quad O \neq \mathbf{0}}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1''+m_2''+1, e_1''+e_2'', r_1''+r_2'')} \mathcal{J}\langle x\rangle t : N}\ \mathsf{app}$$

    where $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$ and

$$(m, e, r) = (m_1'' + m_2'' + 1, e_1'' + e_2'', r_1'' + r_2'')$$

    By applying the *i.h.* on the left-hand side premise, there exists a splitting $M_\Delta = M_{\Delta,1} \uplus M_{\Delta,2}$, with $M_{\Delta,1} \neq \mathbf{0}$, such that for every $\Psi \rhd_O \Pi \vdash^{(m', e', r')} v : M_{\Delta,1}$ there exists

$$\Phi_{\mathcal{J}\langle v\rangle} \rhd_O (\Delta \biguplus \Pi); x : M_{\Delta,2} \vdash^{(m_1''+m', e_1''+e'-1, r_1''+r')} \mathcal{J}\langle v\rangle : [O \multimap N]$$

    We then propose splitting M in $M_1 \coloneqq M_{\Delta,1}$ and $M_2 \coloneqq M_{\Delta,2} \uplus M_\Sigma$, verifying that, for every $\Psi \rhd_O \Pi \vdash^{(m', e', r')} v : M_{\Delta,1}$, $\Phi_{\mathcal{H}\langle v\rangle}$ is given by

$$\frac{\Phi_{\mathcal{J}\langle v\rangle} \qquad \Sigma; x : M_\Sigma \vdash^{(m_2'', e_2'', r_2'')} t : O}{(\Delta \biguplus \Pi \biguplus \Sigma); x : (M_{\Delta,2} \uplus M_\Sigma) \vdash^{(m_1''+m'+m_2''+1, e_1''+e'-1+e_2'', r_1''+r'+r_2'')} \mathcal{J}\langle v\rangle t : N}\ \mathsf{app}$$

    In particular, note that

$$\begin{array}{rl} & (m_1'' + m' + m_2'' + 1, e_1'' + e' - 1 + e_2'', r_1'' + r' + r_2'') \\ = & ((m_1'' + m_2'' + 1) + m', (e_1'' + e_2'') + e' - 1, (r_1'' + r_2'') + r') \\ = & (m + m', e + e' - 1, r + r') \end{array}$$

  - Let $\Phi_{\mathcal{H}\langle x\rangle}$ be of the form

$$\frac{\Delta; x : M_\Delta \vdash^{(m_1'', e_1'', r_1'')} \mathcal{J}\langle x\rangle : \uplus_{j\in J} [\mathsf{inert}] \quad \Sigma; x : M_\Sigma \vdash^{(m_2'', e_2'', r_2'')} t : [\mathsf{tight}]}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1''+m_2''+1, e_1''+e_2'', r_1''+r_2'')} \mathcal{J}\langle x\rangle t : \uplus_{j\in J} [\mathsf{inert}]}\ \mathsf{app}_i$$

    where $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$ and

$$(m, e, r) = (m_1'' + m_2'' + 1, e_1'' + e_2'', r_1'' + r_2'')$$

    The statement holds by *i.h.* on the left-hand premise and then deriving $\Phi_{\mathcal{H}\langle v\rangle}$ via the app_i rule.
  - Let $\Phi_{\mathcal{H}\langle x\rangle}$ be of the form

$$\frac{\Gamma; x : M \vdash^{(m,e,r)} \mathcal{J}\langle x\rangle : [\mathbf{0} \multimap N]}{\Gamma; x : M \vdash^{(m,e,r)} \mathcal{J}\langle x\rangle t : N}\ \mathsf{app}_{\mathrm{gc}}$$

    The statement holds by *i.h.* on the premise and then deriving $\Phi_{\mathcal{H}\langle v\rangle}$ via the $\mathsf{app}_{\mathrm{gc}}$ rule.
- Let $\mathcal{H} = i\mathcal{J}$. We consider the different cases corresponding to the last typing rule in $\Phi_{\mathcal{H}\langle x\rangle}$:

– Let $\Phi_{\mathcal{H}\langle x\rangle}$ be of the form

$$\frac{\Delta; x : M_\Delta \vdash^{(m_1'',e_1'',r_1'')} i : [O \multimap N] \quad \Sigma; x : M_\Sigma \vdash^{(m_2'',e_2'',r_2'')} \mathcal{J}\langle x\rangle : O}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1''+m_2''+1,e_1''+e_2'',r_1''+r_2'')} i\mathcal{J}\langle x\rangle : N}\ \mathsf{app}$$

where $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$ and

$$(m, e, r) = (m_1'' + m_2'' + 1, e_1'' + e_2'', r_1'' + r_2'')$$

By applying the *i.h.* on the right-hand side premise, there exists a splitting $M_\Sigma = M_{\Sigma,1} \uplus M_{\Sigma,2}$, with $|M_{\Sigma,1}| \neq \mathbf{0}$, such that for every $\Psi \rhd_O \Pi \vdash^{(m',e',r')} v : M_{\Sigma,1}$ there exists

$$\Phi_{\mathcal{J}\langle v\rangle} \rhd_O \Sigma; x : M_{\Sigma,2} \vdash^{(m_2''+m',e_2''+e'-1,r_2'')} \mathcal{J}\langle v\rangle : O$$

Let M be split in $M_1 \coloneqq M_{\Sigma,1}$ and $M_2 \coloneqq M_\Delta \uplus M_{\Sigma,2}$. We then derive $\Phi_{\mathcal{H}\langle v\rangle}$ as

$$\frac{\Delta; x : M_\Delta \vdash^{(m_1'',e_1'',r_1'')} i : [O \multimap N] \qquad \Phi_{\mathcal{J}\langle v\rangle}}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_{\Sigma,2}) \vdash^{(m_1''+m_2''+m'+1,e_1''+e_2''+e'-1,r_1''+r_2''+r')} i\mathcal{J}\langle v\rangle : N}\ \mathsf{app}$$

noting that

$$\begin{array}{rl} & (m_1'' + m_2'' + m' + 1, e_1'' + e_2'' + e', r_1'' + r_2'' + r') \\ = & ((m_1'' + m_2'' + 1) + m', (e_1'' + e_2'') + e', (r_1'' + r_2'') + r') \\ = & (m + m', e + e' - 1, r + r') \end{array}$$

– Let $\Phi_{\mathcal{H}\langle x\rangle}$ be of the form

$$\frac{\Delta; x : M_\Delta \vdash^{(m_1'',e_1'',r_1'')} i : \uplus_{j\in J} [\mathsf{inert}] \quad \Sigma; x : M_\Sigma \vdash^{(m_2'',e_2'',r_2'')} \mathcal{J}\langle x\rangle : [\mathsf{tight}]}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1''+m_2''+1,e_1''+e_2'',r_1''+r_2''+1)} i\mathcal{J}\langle x\rangle : \uplus_{j\in J} [\mathsf{inert}]}\ \mathsf{app}_i$$

where $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$ and

$$(m, e, r) = (m_1'' + m_2'', e_1'' + e_2'', r_1'' + r_2'' + 1)$$

The statement holds by *i.h.* on the right-hand side premise and then deriving $\Phi_{\mathcal{H}\langle v\rangle}$ via the app_i rule.

– Suppose $\Phi_{\mathcal{H}\langle x\rangle}$ is of the form

$$\frac{\Phi_i \rhd_O \Gamma; x : M \vdash^{(m'',e'',r'')} i : [\mathbf{0} \multimap N]}{\Gamma; x : M \vdash^{(m''+1,e'',r'')} i\mathcal{J}\langle x\rangle : N}\ \mathsf{app}_{\mathrm{gc}}$$

where $(m, e, r) = (m''+1, e'', r'')$. Note that $x \notin \mathsf{nv}(\mathcal{H}) \supseteq \mathsf{nv}(i)$, and then $\mathsf{inert}_\Gamma(\mathsf{nv}(i))$. However, this implies—by Lemma 7.1.2.2 (Typing properties of normal terms - inert terms)—that $[\mathbf{0} \multimap N] = \mathsf{Inert}$, which is absurd.

2. By induction on the derivation of $P \in \mathcal{E}_\mathcal{V}$:
   - Let $P = (\mathcal{H}, \epsilon) \in \mathcal{E}_{\mathsf{nv}(\mathcal{H})}$. The statement holds by Lemma 7.1.4.1 (Linear Substitution for Open CbNeed).
   - Let $P \in \mathcal{E}_\mathcal{V}$ be derived as

$$\frac{Q \in \mathcal{E}_\mathcal{V} \quad y \notin \mathcal{V}}{Q@[y{\leftarrow}t] \in \mathcal{E}_\mathcal{V}}\ \mathsf{O}_{\mathsf{GC}}$$

   where $P = Q@[y{\leftarrow}t]$. Note that $y \neq x$—given that $x \notin \mathsf{dom}(P)$. We consider the different cases corresponding to the last typing rule in $\Phi_{P\langle x\rangle}$.

– Let $\Phi_{P\langle x\rangle}$ be of the form

$$\frac{\Delta; x : M_\Delta; y : O \vdash^{(m_1'', e_1'', r_1'')} Q\langle x\rangle : N \quad \Sigma; x : M_\Sigma \vdash^{(m_2'', e_2'', r_2'')} t : O \quad O \neq \mathbf{0}}{(\Delta \biguplus \Sigma) ; x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1''+m_2'', e_1''+e_2'', r_1''+r_2'')} Q\langle x\rangle@[y{\leftarrow}t] : N}\ \text{ES}$$

where $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$ and

$$(m, e, r) = (m_1'' + m_2'' + 1, e_1'' + e_2'', r_1'' + r_2'')$$

Since $y \notin \mathcal{V}$, then $\mathsf{inert}_{\Delta; y:O}(\mathcal{V})$ and so—by application of the *i.h.* on the left-hand side premise—there exists a splitting $M_\Delta = M_{\Delta,1} \uplus M_{\Delta,2}$, with $M_{\Delta,1} \neq \mathbf{0}$, such that for every type derivation $\Psi \rhd_O \Pi \vdash^{(m', e', r')} v : M_{\Delta,1}$ there exists

$$\Phi_{Q\langle v\rangle} \rhd_O \left(\Delta \biguplus \Pi\right) ; x : M_{\Delta,2}; y : O \vdash^{(m_1''+m', e_1''+e'-1, r_1''+r')} Q\langle v\rangle : N$$

Note that $y \notin \mathsf{dom}(\Pi)$ because $y \notin \mathsf{fv}(v)$—by α-conversion and Lemma 7.1.1 (Relevance of the cbneed type system). We then propose splitting M in $M_1 = M_{\Delta,1}$ and $M_2 = M_{\Delta,2} \uplus M_\Sigma$, verifying that for every $\Psi \rhd_O \Pi \vdash^{(m', e', r')} v : M_{\Delta,1}$, $\Phi_{P\langle v\rangle}$ is given by

$$\frac{\Phi_{Q\langle v\rangle} \qquad \Sigma; x : M_\Sigma \vdash^{(m_2'', e_2'', r_2'')} t : O}{(\Delta \biguplus \Sigma) ; x : (M_{\Delta,2} \uplus M_\Sigma) \vdash^{(m_1''+m'+m_2'', e_1''+e'-1+e_2'', r_1''+r'+r_2'')} Q\langle v\rangle@[y{\leftarrow}t] : N}\ \text{ES}$$

noting that

$$\begin{array}{rl} & (m_1'' + m' + m_2'', e_1'' + e' - 1 + e_2'', r_1'' + r' + r_2'') \\ = & ((m_1'' + m_2'') + m', (e_1'' + e_2'') + e' - 1, (r_1'' + r_2'') + r') \\ = & (m + m', e + e' - 1, r + r') \end{array}$$

– Let $\Phi_{P\langle x\rangle}$ be of the form

$$\frac{\Gamma; x : M \vdash^{(m,e,r)} Q\langle x\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle x\rangle@[y{\leftarrow}t] : N}\ \text{ES}_{\text{gc}}$$

The statement holds by *i.h.* on the premise and then deriving $\Phi_{P\langle v\rangle}$ via rule ES_{gc}.

- Let $P \in \mathcal{E}_\mathcal{V}$ be derived as

$$\frac{Q \in \mathcal{E}_\mathcal{W} \quad y \in \mathcal{W}}{Q@[y{\leftarrow}i] \in \mathcal{E}_{(\mathcal{W}\setminus\{y\})\cup\mathsf{nv}(i)}}\ \mathsf{O_I}$$

where $P = Q@[y{\leftarrow}i]$. Note that $y \neq x$—since $x \notin \mathsf{dom}(P)$. We consider the different cases corresponding to the last typing rule in $\Phi_{P\langle x\rangle}$.

– Let $\Phi_{P\langle x\rangle}$ be of the form

$$\frac{\Delta; x : M_\Delta; y : O \vdash^{(m_1'', e_1'', r_1'')} Q\langle x\rangle : N \quad \Sigma; x : M_\Sigma \vdash^{(m_2'', e_2'', r_2'')} i : O \quad O \neq \mathbf{0}}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1''+m_2'', e_1''+e_2'', r_1''+r_2'')} Q\langle x\rangle@[y{\leftarrow}i] : N}\ \text{ES}$$

where $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$ and

$$(m, e, r) = (m_1'' + m_2'', e_1'' + e_2'', r_1'' + r_2'')$$

Note that $x \notin \mathsf{nv}(i) \subseteq \mathsf{nv}(P)$, allowing us to apply Lemma 7.1.2.2 (Typing properties of normal terms - inert terms) and conclude that $O = \mathsf{Inert}$. Therefore, $\mathsf{inert}_{\Delta;y:O}(\mathcal{W})$ and so there exists—by *i.h.* on the left-hand side premise—a splitting $M_\Delta = M_{\Delta,1} \uplus M_{\Delta,2}$, with $M_{\Delta,1} \neq \mathbf{0}$, such that for every type derivation $\Psi \triangleright_O \Pi \vdash^{(m',e',r')} v : M_{\Delta,1}$ there exists

$$\Phi_{Q\langle v\rangle} \triangleright_O (\Delta \biguplus \Pi); x : M_{\Delta_2}; y : O \vdash^{(m_1''+m',e_1''+e'-1,r_1''+r')} Q\langle v\rangle : N$$

Note that $y \notin \mathsf{dom}(\Pi)$ because $y \notin \mathsf{fv}(v)$—by α-conversion and Lemma 7.1.1 (Relevance of the Open CbNeed type system). We then propose splitting M in $M_1 = M_{\Delta,1}$ and $M_2 = M_{\Delta,2} \uplus M_\Sigma$, verifying that for every $\Psi \triangleright_O \Pi \vdash^{(m',e',r')} v : M_{\Delta,1}$, $\Phi_{P\langle v\rangle}$ is given by

$$\frac{\Phi_{Q\langle v\rangle} \qquad \Sigma; x : M_\Sigma \vdash^{(m_2'',e_2'',r_2'')} i : O}{(\Delta \biguplus \Sigma); x : (M_{\Delta,2} \uplus M_\Sigma) \vdash^{(m_1''+m'+m_2'',e_1''+e'-1+e_2'',r_1''+r'+r_2'')} Q\langle v\rangle@[y{\leftarrow}i] : N}\ \mathrm{ES}$$

noting that

$$\begin{array}{rl} & (m_1'' + m' + m_2'', e_1'' + e' - 1 + e_2'', r_1'' + r' + r_2'') \\ = & ((m_1'' + m_2'') + m', (e_1'' + e_2'') + e' - 1, (r_1'' + r_2'') + r') \\ = & (m + m', e + e' - 1, r + r') \end{array}$$

– Let $\Phi_{P\langle x\rangle}$ be of the form

$$\frac{\Gamma; x : M \vdash^{(m,e,r)} Q\langle x\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle x\rangle@[y{\leftarrow}i] : N}\ \mathrm{ES}_{\mathrm{gc}}$$

Note that since $y \notin \mathsf{dom}(\Gamma)$, then $\mathsf{inert}_\Gamma(\mathcal{W})$. Then the statement holds by *i.h.* on the premise and then deriving $\Phi_{P\langle v\rangle}$ via rule $\mathrm{ES}_{\mathrm{gc}}$.

- Let $P \in \mathcal{E}_\mathcal{V}$ be derived as

$$\frac{Q \in \mathcal{E}_\mathcal{W} \quad y \notin \mathcal{W}}{Q\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{W}\cup\mathsf{nv}(\mathcal{H})}}\ \mathsf{O}_{\mathsf{HER}}$$

where $P = Q\langle y\rangle@[y{\leftarrow}\mathcal{H}]$ and $\mathcal{V} = \mathcal{W} \cup \mathsf{nv}(\mathcal{H})$. Note that $y \neq x$—since $x \notin \mathsf{dom}(P)$. We consider the different cases corresponding to the last typing rule in $\Phi_{P\langle x\rangle}$.

– Let $\Phi_{P\langle x\rangle}$ be of the form

$$\frac{\Delta; x : M_\Delta; y : O \vdash^{(m_1'',e_1'',r_1'')} Q\langle y\rangle : N \quad \Sigma; x : M_\Sigma \vdash^{(m_2'',e_2'',r_2'')} \mathcal{H}\langle x\rangle : O \quad O \neq \mathbf{0}}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1''+m_2'',e_1''+e_2'',r_1''+r_2'')} Q\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle x\rangle] : N}\ \mathrm{ES}$$

where $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$ and

$$(m, e, r) = (m_1'' + m_2'', e_1'' + e_2'', r_1'' + r_2'')$$

First, note that since $x \notin \mathcal{V}$ then $x \notin \mathsf{nv}(\mathcal{H})$. Thus, $\mathsf{inert}_\Sigma(\mathsf{nv}(\mathcal{H}))$ and so $M_\Sigma \neq \mathbf{0}$—by Lemma 13.4.4.2 (Plugged variables and domain of type contexts with respect to term contexts).
Thus, by application of Lemma 7.1.4.1 (Linear Substitution for Open CbNeed in term contexts) on the right-hand side premise, there exists a splitting $M_\Sigma = M_{\Sigma,1} \uplus M_{\Sigma,2}$ such that for every type derivation $\Psi \triangleright_O \Pi \vdash^{(m',e',r')} v : M_{\Sigma,1}$ there exists

$$\Phi_{\mathcal{H}\langle v\rangle} \triangleright_O (\Sigma \biguplus \Pi); x : M_{\Sigma,2} \vdash^{(m_2''+m',e_2''+e'-1,r_2''+r')} \mathcal{H}\langle v\rangle : O$$

We then propose splitting M in $M_1 \coloneqq M_{\Sigma,1}$ and $M_2 \coloneqq M_\Delta \uplus M_{\Sigma,2}$, verifying that, for every $\Psi \rhd_O \Pi \vdash^{(m',e',r')} v : M_{\Sigma,1}$, $\Phi_{P\langle v\rangle}$ is given by

$$\dfrac{\Delta; x : M_\Delta; y : O \vdash^{(m_1'',e_1'',r_1'')} Q\langle y\rangle : N \qquad \Phi_{\mathcal{H}\langle v\rangle}}{(\Delta \biguplus \Sigma \biguplus \Pi); x : (M_\Delta \uplus M_{\Sigma,2}) \vdash^{(m_1''+m_2''+m', e_1''+e_2''+e'-1, r_1''+r_2''+r')} Q\langle y\rangle @ [y \leftarrow \mathcal{H}\langle v\rangle] : N}\ \text{ES}$$

noting that

$$\begin{array}{rl} & (m_1'' + m_2'' + m', e_1'' + e_2'' + e' - 1, r_1'' + r_2'' + r') \\ = & ((m_1'' + m_2'') + m', (e_1'' + e_2'') + e' - 1, (r_1'' + r_2'') + r') \\ = & (m + m', e + e' - 1, r + r') \end{array}$$

– Suppose $\Phi_{P\langle x\rangle}$ is of the form

$$\dfrac{\Gamma; x : M \vdash^{(m,e,r)} Q\langle y\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle y\rangle @ [y \leftarrow \mathcal{H}\langle x\rangle] : N}\ \text{ES}_{\text{gc}}$$

Since $x \notin \mathcal{W}$ then $\mathsf{inert}_{\Gamma; x : M}(\mathcal{W})$ and so—by Lemma 7.1.1 (Relevance of the Open CbNeed type system)—$y \in \mathsf{dom}(\Gamma)$, making this case absurd.

□

(Click here to go back to main chapter.)

The following is required to apply Lemma 7.1.4 (Linear Substitution for Open CbNeed) in the proof of Proposition 7.1.6.2 (Quantitative Subject Reduction for Open CbNeed - exponential case) to obtain the right indices.

Lemma 13.4.6 (Splitting multi types of Open CbNeed type derivations)**.**
Let $v \in \mathsf{Val}$, $M \coloneqq N \biguplus O$, and let $\Phi \rhd_O \Gamma \vdash^{(m,e)} v : M$ be a type derivation. Then there exist type derivations

$$\begin{array}{l} \Psi \rhd_O \Pi \vdash^{(m',e')} v : N \\ \Theta \rhd_O \Delta \vdash^{(m'',e'')} v : O \end{array}$$

such that $\Gamma = \Pi \biguplus \Delta$ and $(m, e) = (m' + m'', e' + e'')$.

Proof.
Trivial, given that the many typing rule is the only one that can derive a multi type on the right—*i.e.*, the derived type of the subject, in this case M—for values.

□

Lemma 13.4.7 (Quantitative Subject Reduction for $\to_{\mathsf{om}}$ in term contexts)**.**
Let $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} \mathcal{H}\langle(\lambda x.u)s\rangle : M$, with $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$. Then $m \geq 1$ and there exists $\Phi' \rhd_O \Gamma \vdash^{(m-1,e,r)} (\mathcal{H}\langle u\rangle, [x \leftarrow s]) : M$.

Proof. (Click here to go back to main chapter.)
By structural induction on $\mathcal{H}$:

- *Empty context*; *i.e.*, $\mathcal{H} = \langle\cdot\rangle$. We do case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\dfrac{\dfrac{\dfrac{\Theta \rhd_O \Pi; x:N \vdash^{(m',e',r')} u:M}{\Pi \vdash^{(m',e',r')} \lambda x.u : N \multimap M}\,\mathsf{fun}}{\Pi \vdash^{(m',e',r')} \lambda x.u : [N \multimap M]}\,\mathsf{many} \qquad \Xi \rhd_O \Delta \vdash^{(m'',e'',r'')} s:N}{\Pi \biguplus \Delta \vdash^{(m'+m''+1,e'+e'',r'+r'')} (\lambda x.u)s : M}\,\mathsf{app}$$

with $\Gamma = \Pi \biguplus \Delta$ and $(m,e,r) = (m'+m''+1, e'+e'', r'+r'')$. Note that $m \geq 1$. We can then derive Φ' as

$$\dfrac{\dfrac{\Theta}{\Pi; x:N \vdash^{(m',e',r')} (u,\epsilon) : M}\,\mathsf{Lift} \qquad \Xi}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} (u, [x \leftarrow s]) : M}\,\mathrm{ES}$$

– Note that app_i is ruled out as the last typing rule in Φ by the fact that the only linear type assignable to values are of the form $M \multimap N$ and abs.

– Let Φ be of the form

$$\dfrac{\dfrac{\dfrac{\Theta \rhd_O \Gamma \vdash^{(m',e',r')} u:M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m',e',r')} \lambda x.u : \mathbf{0} \multimap M}\,\mathsf{fun}}{\Gamma \vdash^{(m',e',r')} \lambda x.u : [\mathbf{0} \multimap M]}\,\mathsf{many}}{\Gamma \vdash^{(m'+1,e',r')} (\lambda x.u)s : M}\,\mathsf{app}_{\mathrm{gc}}$$

with $(m,e,r) = (m'+1, e', r')$. Note that $m \geq 1$. We can then derive Φ' as

$$\dfrac{\dfrac{\Theta}{\Gamma \vdash^{(m',e',r')} (u,\epsilon) : M}\,\mathsf{Lift} \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m',e',r')} (u, [x \leftarrow s]) : M}\,\mathrm{ES}_{\mathrm{gc}}$$

- *Application left*; *i.e.*, $\mathcal{H} = \mathcal{J}t$. We do case analysis on the last typing rule in Ψ:

– Let Ψ be of the form

$$\dfrac{\Psi \rhd_O \Pi \vdash^{(m',e',r')} \mathcal{J}\langle(\lambda x.u)s\rangle : [N \multimap M] \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} t:N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m''+1,e'+e'',r'+r'')} \mathcal{J}\langle(\lambda x.u)s\rangle t : M}\,\mathsf{app}$$

with $\Gamma = \Pi \biguplus \Delta$ and $(m,e,r) = (m'+m''+1, e'+e'', r'+r'')$. Note that $\mathsf{inert}_{\Pi \biguplus \Delta}(\mathsf{nv}(\mathcal{H}))$ implies $\mathsf{inert}_{\Pi}(\mathsf{nv}(\mathcal{J}))$. By *i.h.* on Ψ, we have that $m' \geq 1$ and there exists $\Psi' \rhd_O \Pi \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, [x \leftarrow s]) : [N \multimap M]$. We do case analysis on the last typing rule in Ψ':

∗ Let Ψ' be of the form

$$\dfrac{\dfrac{\Omega \rhd_O \Pi_1; x:O \vdash^{(m_1',e_1',r_1')} \mathcal{J}\langle u\rangle : [N \multimap M]}{\Pi_1; x:O \vdash^{(m_1',e_1',r_1')} (\mathcal{J}\langle u\rangle, \epsilon) : [N \multimap M]}\,\mathsf{Lift} \qquad Z \rhd_O \Pi_2 \vdash^{(m_2',e_2',r_2')} s:O \quad O \neq \mathbf{0}}{\Pi_1 \biguplus \Pi_2 \vdash^{(m_1'+m_2',e_1'+e_2',r_1'+r_2')} (\mathcal{J}\langle u\rangle, [x \leftarrow s]) : [N \multimap M]}\,\mathrm{ES}$$

with $\Pi = \Pi_1 \biguplus \Pi_2$ and $(m'-1, e', r') = (m_1'+m_2', e_1'+e_2', r_1'+r_2')$. We can then derive Φ' as follows

$$\dfrac{\dfrac{\Omega \qquad \Theta}{(\Pi_1 \biguplus \Delta); x:O \vdash^{(m_1'+m''+1,e_1'+e'',r_1'+r'')} \mathcal{J}\langle u\rangle t : M}\,\mathsf{app} \qquad Z}{\Pi_1 \biguplus \Delta \biguplus \Pi_2 \vdash^{(m_1'+m''+1+m_2',e_1'+e''+e_2',r_1'+r''+r_2')} (\mathcal{J}\langle u\rangle t, [x \leftarrow s]) : M}\,\mathrm{ES}$$

noting that $\Pi_1 \uplus \Delta \uplus \Pi_2 = \Pi \uplus \Delta = \Gamma$ and that

$$\begin{array}{rl} & (m'_1 + m'' + 1 + m'_2, e'_1 + e'' + e'_2, r'_1 + r'' + r'_2) \\ = & (m' - 1 + m'' + 1, e' + e'', r' + r'') \\ = & (m - 1, e, r) \end{array}$$

∗ Let Ψ' be of the form

$$\dfrac{\dfrac{\Omega \rhd_O \Pi \vdash^{(m'-1,e',r')} \mathcal{J}\langle u\rangle : [N \multimap M]}{\Pi \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, \epsilon) : [N \multimap M]}\ \mathsf{Lift} \qquad \Pi(x) = \mathbf{0}}{\Pi \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [N \multimap M]}\ \mathrm{ES_{gc}}$$

We can then derive Φ' as follows

$$\dfrac{\dfrac{\Omega \qquad \Theta}{\Pi \uplus \Delta \vdash^{(m'-1+m''+1,e'+e'',r'+r'')} \mathcal{J}\langle u\rangle t : M}\ \mathsf{app} \qquad (\Pi \uplus \Delta)(x) = \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'-1+m''+1,e'+e'',r'+r'')} (\mathcal{J}\langle u\rangle t, [x{\leftarrow}s]) : M}\ \mathrm{ES_{gc}}$$

noting that $(m' - 1 + m'' + 1, e' + e'', r' + r'') = (m - 1, e, r)$.

– Let Φ be of the form

$$\dfrac{\Psi \rhd_O \Pi \vdash^{(m',e',r')} \mathcal{J}\langle(\lambda x.u)s\rangle : \uplus_{j\in J}\,[\mathsf{inert}] \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} t : [\mathsf{tight}] \quad J \neq \emptyset}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r''+1)} \mathcal{J}\langle(\lambda x.u)s\rangle t : \uplus_{j\in J}\,[\mathsf{inert}]}\ \mathsf{app}_i$$

with $\Gamma = \Pi \uplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'' + 1)$. Note that $x \notin \mathsf{dom}(\Delta)$—because of Lemma 7.1.1 (Relevance of the Open CbNeed type system) and the fact that $x \notin \mathsf{fv}(t)$, due to the variable convention. Moreover, note that $\mathsf{inert}_{\Pi \uplus \Delta}(\mathsf{nv}(\mathcal{H}))$ implies $\mathsf{inert}_{\Pi}(\mathsf{nv}(\mathcal{J}))$. Hence, by *i.h.* on Ψ, $m' \geq 1$ and there exists

$$\Psi' \rhd_O \Pi \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : \uplus_{j\in J}\,[\mathsf{inert}]$$

We do case analysis on the last typing rule in Ψ':

∗ Let Ψ' be of the form

$$\dfrac{\dfrac{\Xi \rhd_O \Pi_1; x : O \vdash^{(m'_1,e'_2,r'_1)} \mathcal{J}\langle u\rangle : \uplus_{j\in J}\,[\mathsf{inert}]}{\Pi_1; x : O \vdash^{(m'_1,e'_2,r'_1)} (\mathcal{J}\langle u\rangle, \epsilon) : \uplus_{j\in J}\,[\mathsf{inert}]}\ \mathsf{Lift} \qquad \Omega \rhd_O \Pi_2 \vdash^{(m'_2,e'_2,r'_2)} s : O \quad O \neq \mathbf{0}}{\Pi_1 \uplus \Pi_2 \vdash^{(m'_1+m'_2,e'_1+e'_2,r'_1+r'_2)} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : \uplus_{j\in J}\,[\mathsf{inert}]}\ \mathrm{ES}$$

with $\Pi = \Pi_1 \uplus \Pi_2$ and

$$(m' - 1, e', r') = (m'_1 + m'_2, e'_1 + e'_2, r'_1 + r'_2)$$

We can then derive Φ' as follows

$$\dfrac{\dfrac{\Xi \qquad \Theta}{(\Pi_1 \uplus \Delta); x : O \vdash^{(m'_1+m'',e'_1+e'',r'_1+r''+1)} \mathcal{J}\langle u\rangle t : \uplus_{j\in J}\,[\mathsf{inert}]}\ \mathsf{app}_i \qquad \Omega}{\Pi_1 \uplus \Delta \uplus \Pi_2 \vdash^{(m'_1+m''+m'_2,e'_1+e''+e'_2,r'_1+r''+1+r'_2)} (\mathcal{J}\langle u\rangle t, [x{\leftarrow}s]) : \uplus_{j\in J}\,[\mathsf{inert}]}\ \mathrm{ES}$$

Note that $\Pi_1 \uplus \Delta \uplus \Pi_2 = \Pi \uplus \Delta = \Gamma$ and that

$$\begin{array}{rl} & (m'_1 + m'' + m'_2, e'_1 + e'' + e'_2, r'_1 + r'' + 1 + r'_2) \\ = & (m' - 1 + m'', e' + e'', r' + r'' + 1) \\ = & (m - 1, e, r) \end{array}$$

∗ Let Ψ' be of the form

$$\dfrac{\dfrac{\Xi \rhd_O \Pi \vdash^{(m'-1,e',r')} \mathcal{J}\langle u\rangle : \uplus_{j\in J}\,[\mathsf{inert}]}{\Pi \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, \epsilon) : \uplus_{j\in J}\,[\mathsf{inert}]}\ \mathsf{Lift} \qquad \Pi(x) = \mathbf{0}}{\Pi \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : \uplus_{j\in J}\,[\mathsf{inert}]}\ \mathrm{ES_{gc}}$$

We can then derive Φ' as follows

$$\dfrac{\dfrac{\dfrac{\Xi \qquad \Theta}{\Pi \biguplus \Delta \vdash^{(m'-1+m'',e'+e'',r'+r''+1)} \mathcal{J}\langle u\rangle t : \uplus_{j\in J}\,[\mathsf{inert}]}\ \mathsf{app}_i}{\Pi \biguplus \Delta \vdash^{(m'-1+m'',e'+e'',r'+r''+1)} (\mathcal{J}\langle u\rangle t, \epsilon) : \uplus_{j\in J}\,[\mathsf{inert}]}\ \mathsf{Lift} \qquad (\Pi \biguplus \Delta)(x) = \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'-1+m'',e'+e'',r'+r''+1)} (\mathcal{J}\langle u\rangle t, [x{\leftarrow}s]) : \uplus_{j\in J}\,[\mathsf{inert}]}\ \mathrm{ES_{gc}}$$

noting that

$$(m' - 1 + m'', e' + e'', r' + r'' + 1) = (m - 1, e, r)$$

– Let Ψ be of the form

$$\dfrac{\Psi \rhd_O \Gamma \vdash^{(m',e',r')} \mathcal{J}\langle(\lambda x.u)s\rangle : [\mathbf{0} \multimap M]}{\Gamma \vdash^{(m'+1,e',r')} \mathcal{J}\langle(\lambda x.u)s\rangle t : M}\ \mathsf{app}_{\mathrm{gc}}$$

with $(m, e, r) = (m' + 1, e', r')$. Note that $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$ implies $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{J}))$. Hence, by *i.h.* on Ψ, $m' \geq 1$ and there exists $\Psi' \rhd_O \Gamma \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [\mathbf{0} \multimap M]$. We do case analysis on the last typing rule in Ψ':

∗ Let Ψ' be of the form

$$\dfrac{\dfrac{\Theta \rhd_O \Pi; x : N \vdash^{(m'_1,e'_1,r'_1)} \mathcal{J}\langle u\rangle : [\mathbf{0} \multimap M]}{\Pi; x : N \vdash^{(m'_1,e'_1,r'_1)} (\mathcal{J}\langle u\rangle, \epsilon) : [\mathbf{0} \multimap M]}\ \mathsf{Lift} \qquad \Xi \rhd_O \Delta \vdash^{(m'_2,e'_2,r'_2)} s : N}{\Pi \biguplus \Delta \vdash^{(m'_1+m'_2,e'_1+e'_2,r'_1+r'_2)} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [\mathbf{0} \multimap M]}\ \mathrm{ES}$$

with $\Pi \biguplus \Delta = \Gamma$ and $(m'_1 + m'_2, e'_1 + e'_2, r'_1 + r'_2) = (m' - 1, e', r')$. We can then build Φ' as

$$\dfrac{\dfrac{\Theta}{\Pi; x : N \vdash^{(m'_1+1,e'_1,r'_1)} \mathcal{J}\langle u\rangle t : M}\ \mathsf{app}_{\mathrm{gc}} \qquad \Xi}{\Pi \biguplus \Delta \vdash^{(m'_1+1+m'_2,e'_1+e'_2,r'_1+r'_2)} (\mathcal{J}\langle u\rangle t, [x{\leftarrow}s]) : M}\ \mathrm{ES}$$

noting that

$$(m'_1 + 1 + m'_2, e'_1 + e'_2, r'_1 + r'_2) = (m', e', r') = (m - 1, e, r)$$

∗ Let Ψ' be of the form

$$\dfrac{\dfrac{\Theta \rhd_O \Gamma \vdash^{(m'-1,e',r')} \mathcal{J}\langle u\rangle : [\mathbf{0} \multimap M]}{\Gamma \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, \epsilon) : [\mathbf{0} \multimap M]}\ \mathsf{Lift} \qquad \Pi(x) = \mathbf{0}}{\Gamma \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [\mathbf{0} \multimap M]}\ \mathrm{ES_{gc}}$$

We can then build Φ' as

$$\dfrac{\dfrac{\Theta}{\Pi \vdash^{(m',e',r')} \mathcal{J}\langle u\rangle t : M}\ \mathsf{app}_{\mathrm{gc}} \qquad \Pi(x) = \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m',e',r')} (\mathcal{J}\langle u\rangle t, [x{\leftarrow}s]) : M}\ \mathrm{ES_{gc}}$$

noting that $(m', e', r') = (m - 1, e, r)$.

- *Application right*; *i.e.*, $\mathcal{H} = i\mathcal{J}$: We do case analysis on the last typing rule in Φ.
  - Let Φ be of the form

$$\frac{\Psi \rhd_O \Pi \vdash^{(m',e',r')} i : [N \multimap M] \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} \mathcal{J}\langle(\lambda x.u)s\rangle : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m''+1,e'+e'',r'+r'')} i\mathcal{J}\langle(\lambda x.u)s\rangle : M}\ \mathsf{app}$$

with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'' + 1, e' + e'', r' + r'')$. Note that $x \notin \mathsf{dom}(\Pi)$ because $x \notin \mathsf{fv}(i)$—by α-conversion and Lemma 7.1.1 (Relevance of the Open CbNeed type system).
Moreover, note that $\mathsf{inert}_{\Pi \biguplus \Delta}(\mathsf{nv}(\mathcal{H}))$ implies $\mathsf{inert}_{\Delta}(\mathsf{nv}(\mathcal{J}))$. Hence, by *i.h.* on Θ, $m'' \geq 1$ (hence, $m \geq 1$) and there exists $\Theta' \rhd_O \Delta \vdash^{(m''-1,e'',r'')} (\mathcal{J}\langle u\rangle, [x \leftarrow s]) : N$. We do case analysis on the last typing rule in Θ':
    - Let Θ' be of the form

$$\frac{\dfrac{\Xi \rhd_O \Delta_1 \vdash^{(m_1'',e_1'',r_1'')} \mathcal{J}\langle u\rangle : N}{\Delta_1; x : O \vdash^{(m_1'',e_1'',r_1'')} (\mathcal{J}\langle u\rangle, \epsilon) : N}\ \mathsf{Lift} \quad \Omega \rhd_O \Delta_2 \vdash^{(m_2'',e_2'',r_2'')} s : O}{\Delta_1 \biguplus \Delta_2 \vdash^{(m_1''+m_2'',e_1''+e_2'',r_1''+r_2'')} (\mathcal{J}\langle u\rangle, [x \leftarrow s]) : N}\ \mathrm{ES}$$

with $\Delta = \Delta_1 \biguplus \Delta_2$ and $(m'' - 1, e'', r'') = (m_1'' + m_2'', e_1'' + e_2'', r_1'' + r_2'')$. We can then derive Φ' as follows

$$\frac{\dfrac{\dfrac{\Psi \quad \Xi}{(\Pi \biguplus \Delta_1); x : O \vdash^{(m'+m_1''+1,e'+r_1'',r'+r_1'')} i\mathcal{J}\langle u\rangle : M}\ \mathsf{app}}{(\Pi \biguplus \Delta_1); x : O \vdash^{(m'+m_1''+1,e'+r_1'',r'+r_1'')} (i\mathcal{J}\langle u\rangle, \epsilon) : M}\ \mathsf{Lift} \quad Z}{\Pi \biguplus \Delta_1 \biguplus \Delta_2 \vdash^{(m'+m_1''+1+m_2'',e'+r_1''+e_2'',r'+r_1''+r_2'')} (i\mathcal{J}\langle u\rangle, [x \leftarrow s]) : M}\ \mathrm{ES}$$

noting that $\Pi \biguplus \Delta_1 \biguplus \Delta_2 = \Pi \biguplus \Delta = \Gamma$ and that

$$\begin{array}{rl} & (m' + m_1'' + 1 + m_2'', e' + r_1'' + e_2'', r' + r_1'' + r_2'') \\ = & (m' + (m'' - 1) + 1, e' + e'', r' + r'') \\ = & (m - 1, e, r) \end{array}$$

    - Let Θ' be of the form

$$\frac{\dfrac{\Omega \rhd_O \Delta \vdash^{(m''-1,e'',r'')} \mathcal{J}\langle u\rangle : N}{\Delta \vdash^{(m''-1,e'',r'')} (\mathcal{J}\langle u\rangle, \epsilon) : N}\ \mathsf{Lift} \quad \Delta(x) = \mathbf{0}}{\Delta \vdash^{(m''-1,e'',r'')} (\mathcal{J}\langle u\rangle, [x \leftarrow s]) : N}\ \mathrm{ES}_{\mathrm{gc}}$$

We can then derive Φ' as follows

$$\frac{\dfrac{\dfrac{\Psi \qquad \Omega}{\Pi \biguplus \Delta \vdash^{(m'+m''-1+1,e'+e'',r'+r'')} i\mathcal{J}\langle u\rangle : M}\ \mathsf{app}}{\Pi \biguplus \Delta \vdash^{(m'+m''-1+1,e'+e'',r'+r'')} (i\mathcal{J}\langle u\rangle, \epsilon) : M}\ \mathsf{Lift} \quad (\Pi \biguplus \Delta)(x) = \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m''-1+1,e'+e'',r'+r'')} (i\mathcal{J}\langle u\rangle, [x \leftarrow s]) : M}\ \mathrm{ES}_{\mathrm{gc}}$$

noting that $(m' + m'' - 1 + 1, e' + e'', r' + r'') = (m - 1, e, r)$.

– Let Φ be of the form

$$\frac{\Psi \rhd_O \Pi \vdash^{(m',e',r')} i : \uplus_{i\in I}\,[\mathsf{inert}] \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} \mathcal{J}\langle(\lambda x.u)s\rangle : [\mathsf{tight}]}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r''+1)} i\mathcal{J}\langle(\lambda x.u)s\rangle : \uplus_{i\in I}\,[\mathsf{inert}]}\ \mathsf{app}_i$$

with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'' + 1)$. Note that $x \notin \mathsf{dom}(\Pi)$ because $x \notin \mathsf{fv}(i)$—by α-conversion and Lemma 7.1.1 (Relevance of the Open CbNeed type system).
Moreover, note that $\mathsf{inert}_{\Pi\biguplus\Delta}(\mathsf{nv}(\mathcal{H}))$ implies $\mathsf{inert}_{\Delta}(\mathsf{nv}(\mathcal{J}))$. Hence, by *i.h.* on Θ, $m'' \geq 1$ (hence, $m \geq 1$) and there exists $\Theta' \rhd_O \Delta \vdash^{(m''-1,e'',r'')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [\mathsf{tight}]$. We do case analysis on the last typing rule in Θ':

* Let Θ' be of the form

$$\frac{\dfrac{\Xi \rhd_O \Delta_1; x : O \vdash^{(m_1'',e_1'',r_1'')} \mathcal{J}\langle u\rangle : N}{\Delta_1; x : O \vdash^{(m_1'',e_1'',r_1'')} (\mathcal{J}\langle u\rangle, \epsilon) : N}\ \mathsf{Lift} \quad \Omega \rhd_O \Delta_2 \vdash^{(m_2'',e_2'',r_2'')} s : O}{\Delta_1 \biguplus \Delta_2 \vdash^{(m_1''+m_2'',e_1''+e_2'',r_1''+r_2'')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : N}\ \mathrm{ES}$$

with $\Delta = \Delta_1 \biguplus \Delta_2$ and $(m'' - 1, e'', r'') = (m_1'' + m_2'', e_1'' + e_2'', r_1'' + r_2'')$. We can then derive Ψ' as follows

$$\frac{\dfrac{\dfrac{\Psi \qquad \Xi}{(\Pi \biguplus \Delta_1) ; x : O \vdash^{(m'+m_1'',e'+r_1'',r'+r_1''+1)} i\mathcal{J}\langle u\rangle : M}\ \mathsf{app}_i}{(\Pi \biguplus \Delta_1) ; x : O \vdash^{(m'+m_1'',e'+r_1'',r'+r_1''+1)} (i\mathcal{J}\langle u\rangle, \epsilon) : M}\ \mathsf{Lift} \quad \Omega}{\Pi \biguplus \Delta_1 \biguplus \Delta_2 \vdash^{(m'+m_1''+m_2'',e'+r_1''+e_2'',r'+r_1''+1+r_2'')} (i\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : M}\ \mathrm{ES}$$

noting that $\Pi \biguplus \Delta_1 \biguplus \Delta_2 = \Pi \biguplus \Delta = \Gamma$ and that

$$\begin{array}{rl} & (m' + m_1'' + m_2'', e' + r_1'' + e_2'', r' + r_1'' + 1 + r_2'') \\ = & (m' + (m'' - 1), e' + e'', r' + r'' + 1) \\ = & (m - 1, e, r) \end{array}$$

* Let Θ' be of the form

$$\frac{\dfrac{\Xi \rhd_O \Delta \vdash^{(m''-1,e'',r'')} \mathcal{J}\langle u\rangle : [\mathsf{tight}]}{\Delta \vdash^{(m''-1,e'',r'')} (\mathcal{J}\langle u\rangle, \epsilon) : [\mathsf{tight}]}\ \mathsf{Lift} \quad \Delta(x) = \mathbf{0}}{\Delta \vdash^{(m''-1,e'',r'')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [\mathsf{tight}]}\ \mathrm{ES}_{\mathrm{gc}}$$

We can then derive Φ' as follows

$$\frac{\dfrac{\dfrac{\Psi \qquad \Xi}{\Pi \biguplus \Delta \vdash^{(m'+m''-1,e'+e'',r'+r''+1)} i\mathcal{J}\langle u\rangle : M}\ \mathsf{app}_i}{\Pi \biguplus \Delta \vdash^{(m'+m''-1,e'+e'',r'+r''+1)} (i\mathcal{J}\langle u\rangle, \epsilon) : M}\ \mathsf{Lift} \quad (\Pi \biguplus \Delta)(x) = \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m''-1,e'+e'',r'+r''+1)} (i\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : M}\ \mathrm{ES}_{\mathrm{gc}}$$

noting that

$$(m' + m'' - 1, e' + e'', r' + r'' + 1) = (m - 1, e, r)$$

– Suppose Φ is of the form

$$\frac{\Psi \rhd_O \Gamma \vdash^{(m',e',r')} i : [\mathbf{0} \multimap M]}{\Gamma \vdash^{(m'+1,e',r')} i\mathcal{H}\langle(\lambda x.u)s\rangle : M}\ \mathsf{app}_{\mathrm{gc}}$$

with $(m, e, r) = (m' + 1, e', r')$. But then Lemma 7.1.2.2 (Typing properties of normal terms - inert terms) together with the fact that $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{J}))$—given by $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$—would imply that $[\mathbf{0} \multimap M] = [\mathsf{inert}]$—absurd.

□

(Click here to go back to main chapter.)

Proposition 13.4.8 (Quantitative Subject Reduction for Open CbNeed)**.**
Let $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} p : M$ be a tight *type derivation.*

1. Multiplicative*: If $p \to_{\mathsf{om}} p'$, then $m \geq 1$ and there exists a type derivation $\Phi' \rhd_O \Gamma \vdash^{(m-1,e,r)} p' : M$.*
2. Exponential*: If $p \to_{\mathsf{oe}} p'$, then $e \geq 1$ and there exists a type derivation $\Phi' \rhd_O \Gamma \vdash^{(m,e-1,r)} p' : M$.*

Proof. (Click here to go back to main chapter.)

1. We prove this by means of a weaker statement:

   Let $p = P\langle(\lambda x.u)s\rangle \to_{\mathsf{om}} P\langle u, [x{\leftarrow}s]\rangle = p'$ with $P \in \mathcal{E}_\mathcal{V}$, and let $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} P\langle(\lambda x.u)s\rangle : M$ be a type derivation such that $\mathsf{inert}_\Gamma(\mathcal{V})$. Then $m \geq 1$ and there exists $\Phi' \rhd_O \Gamma \vdash^{(m-1,e,r)} P\langle u, [x{\leftarrow}s]\rangle : M$.

   We proceed by induction on the derivation of $P \in \mathcal{E}_\mathcal{V}$.

   - *Empty environment.* Let $P = (\mathcal{H}, \epsilon)$. Note that $\mathcal{V} = \mathsf{nv}(\mathcal{H})$. Then Φ is of the form

   $$\frac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} \mathcal{H}\langle(\lambda x.u)s\rangle : M}{\Gamma \vdash^{(m,e,r)} (\mathcal{H}\langle(\lambda x.u)s\rangle, \epsilon) : M}\ \mathsf{Lift}$$

     The statement holds by application of Lemma 7.1.5 (Quantitative Subject Reduction for $\to_{\mathsf{om}}$ in term contexts) on Ψ, giving us that $m \geq 1$ and the existence of $\Psi' \rhd_O \Gamma \vdash^{(m-1,e,r)} (\mathcal{H}\langle u\rangle, [x{\leftarrow}s]) : M$.
   - Let

   $$\frac{Q \in \mathcal{E}_\mathcal{V} \quad y \notin \mathcal{V}}{Q@[y{\leftarrow}t] \in \mathcal{E}_\mathcal{V}}\ \mathsf{O}_{\mathsf{GC}}$$

     We proceed by case analysis on the last typing rule in Φ.
     – Let Φ be of the form

     $$\frac{\Psi \rhd_O \Pi; y : N \vdash^{(m',e',r')} Q\langle(\lambda x.u)s\rangle : M \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle(\lambda x.u)s\rangle@[y{\leftarrow}t] : M}\ \mathrm{ES}$$

       where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. Note that since $y \notin \mathcal{V}$, then $\mathsf{inert}_{\Pi \biguplus \Delta}(\mathcal{V})$ implies $\mathsf{inert}_{\Pi; y:N}(\mathcal{V})$. Hence, by *i.h.* on Ψ, $m' \geq 1$—thus, $m \geq 1$—and there exists $\Psi' \rhd_O \Pi; y : N \vdash^{(m'-1,e',r')} Q\langle u, [x{\leftarrow}s]\rangle : M$. We can simply derive Φ' as follows

       $$\frac{\Psi' \qquad \Theta}{\Pi \biguplus \Delta \vdash^{(m'-1+m'',e'+e'',r'+r'')} Q\langle u, [x{\leftarrow}s]\rangle@[y{\leftarrow}t] : M}\ \mathrm{ES}$$

       noting that $(m' - 1 + m'', e' + e'', r' + r'') = (m - 1, e, r)$

– Let Φ be of the form

$$\frac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} Q\langle(\lambda x.u)s\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle(\lambda x.u)s\rangle@[y{\leftarrow}t] : M}\ \mathrm{ES_{gc}}$$

By *i.h.* on Ψ, $m \geq 1$ and there exists $\Psi' \rhd_O \Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle : M$. We can simply derive Φ' as follows

$$\frac{\Psi' \qquad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle@[y{\leftarrow}t] : M}\ \mathrm{ES_{gc}}$$

- Let

$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \in \mathcal{W}}{Q@[y{\leftarrow}i] \in \mathcal{E}_{(\mathcal{W}\setminus\{y\})\cup \mathsf{nv}(i)}}\ \mathsf{O_I}$$

where $\mathcal{V} = (\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(i)$. We proceed by case analysis on the last typing rule in Φ.

– Let Φ be of the form

$$\frac{\Psi \rhd_O \Pi; y : N \vdash^{(m',e',r')} Q\langle(\lambda x.u)s\rangle : M \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} i : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle(\lambda x.u)s\rangle@[y{\leftarrow}i] : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$.
First, note that $\mathsf{inert}_{\Pi \biguplus \Delta}((\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(i))$ implies that $\mathsf{inert}_{\Delta}(\mathsf{nv}(i))$, and so we have that N is Inert—by Lemma 7.1.2.2 (Typing properties of normal terms - inert terms) on Θ.
Thus, we have that $\mathsf{inert}_{\Pi;y:N}(Q)$ and we can apply the *i.h.* on Ψ to get that $m' \geq 1$—thus, $m \geq 1$—and the existence of $\Psi' \rhd_O \Pi; y : N \vdash^{(m'-1,e',r')} Q\langle u, [x{\leftarrow}s]\rangle : M$. We can simply derive Φ' as follows

$$\frac{\Psi' \qquad \Theta}{\Pi \biguplus \Delta \vdash^{(m'-1+m'',e'+e'',r'+r'')} Q\langle u, [x{\leftarrow}s]\rangle@[y{\leftarrow}i] : M}\ \mathrm{ES}$$

– Let Φ be of the form

$$\frac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} Q\langle(\lambda x.u)s\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle(\lambda x.u)s\rangle@[y{\leftarrow}i] : M}\ \mathrm{ES_{gc}}$$

Since $\mathsf{inert}_{\Gamma}((\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(i))$ and the fact that $y \notin \mathsf{dom}(\Gamma)$ imply that $\mathsf{inert}_{\Gamma}(\mathcal{W})$, then applying *i.h.* on Ψ gives us that $m \geq 1$ and the existence of $\Psi' \rhd_O \Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle : M$. We can simply derive Φ' as follows

$$\frac{\Psi' \qquad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle@[y{\leftarrow}i] : M}\ \mathrm{ES_{gc}}$$

- Let

$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \notin \mathcal{W}}{Q\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{W}\cup \mathsf{nv}(\mathcal{H})}}\ \mathsf{O_{HER}}$$

where $\mathcal{V} = \mathcal{W} \cup \mathsf{nv}(\mathcal{H})$. Note that $x \neq y$. We proceed by case analysis on the last typing rule in Φ.

– Let Φ be of the form

$$\frac{\Psi \rhd_O \Pi; y : N \vdash^{(m',e',r')} Q\langle y\rangle : M \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} \mathcal{H}\langle(\lambda x.u)s\rangle : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle y\rangle @ [y \leftarrow \mathcal{H}\langle(\lambda x.u)s\rangle] : M}\ \text{ES}$$

where $\Gamma = \Pi \uplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. Note that $x \notin \mathsf{dom}(\Pi)$ because $x \notin \mathsf{fv}(Q\langle y, \epsilon\rangle)$—by α-conversion and Lemma 7.1.1 (Relevance of the Open CbNeed type system).
Note that, by Lemma 7.1.5 (Quantitative Subject Reduction for $\to_{\mathsf{om}}$ in term contexts), $m'' \geq 1$—thus, $m \geq 1$—and there exists $\Theta' \rhd_O \Delta \vdash^{(m''-1,e'',r'')} \mathcal{H}\langle u\rangle[x \leftarrow s] : N$. We now proceed by case analysis on the last typing rule in Θ':

∗ Let Θ' be of the form

$$\frac{\dfrac{\Xi \rhd_O \Delta_1; x : O \vdash^{(m_1'',e_1'',r_1'')} \mathcal{H}\langle u\rangle : N}{\Delta_1; x : O \vdash^{(m_1'',e_1'',r_1'')} (\mathcal{H}\langle u\rangle, \epsilon) : N}\ \mathsf{Lift} \qquad \Omega \rhd_O \Delta_2 \vdash^{(m_2'',e_2'',r_2'')} s : O}{\Delta_1 \uplus \Delta_2 \vdash^{(m_1''+m_2'',e_1''+e_2'',r_1''+r_2'')} (\mathcal{H}\langle u\rangle, [x \leftarrow s]) : N}\ \text{ES}$$

with $\Delta_1 \uplus \Delta_2 = \Delta$ and

$$(m_1'' + m_2'', e_1'' + e_2'', r_1'' + r_2'') = (m'' - 1, e'', r'')$$

We can then derive Φ' as follows

$$\frac{\dfrac{\Psi \qquad \Xi}{(\Pi \uplus \Delta_1); x : O \vdash^{(m'+m_1'',e'+e_1'',r'+r_1'')} Q\langle y\rangle @ [y \leftarrow \mathcal{H}\langle u\rangle] : M}\ \text{ES} \qquad \Omega}{\Pi \uplus \Delta_1 \uplus \Delta_2 \vdash^{(m'+m_1''+m_2'',e'+e_1''+e_2'',r'+r_1''+r_2'')} (Q\langle y\rangle @ [y \leftarrow \mathcal{H}\langle u\rangle]) @ [x \leftarrow s] : M}\ \text{ES}$$

noting that $\Pi \uplus \Delta_1 \uplus \Delta_2 = \Pi \uplus \Delta = \Gamma$, that

$$(Q\langle y\rangle @ [y \leftarrow \mathcal{H}\langle u\rangle]) @ [x \leftarrow s] = (Q\langle y\rangle @ [y \leftarrow \mathcal{H}])\,\langle u, [x \leftarrow s]\rangle = P\langle u, [x \leftarrow s]\rangle$$

and that

$$\begin{array}{rl} & (m' + m_1'' + m_2'', e' + e_1'' + e_2'', r' + r_1'' + r_2'') \\ = & (m' + m'' - 1, e' + e'', r' + r'') \\ = & (m - 1, e, r) \end{array}$$

∗ Let Θ' be of the form

$$\frac{\dfrac{\Xi \rhd_O \Delta \vdash^{(m''-1,e'',r'')} \mathcal{H}\langle u\rangle : N}{\Delta \vdash^{(m''-1,e'',r'')} (\mathcal{H}\langle u\rangle, \epsilon) : N}\ \mathsf{Lift} \qquad \Delta(x) = \mathbf{0}}{\Delta \vdash^{(m''-1,e'',r'')} (\mathcal{H}\langle u\rangle, [x \leftarrow s]) : N}\ \text{ES}_{\text{gc}}$$

We can then derive Φ' as follows

$$\frac{\dfrac{\Psi \qquad \Xi}{\Pi \uplus \Delta \vdash^{(m'+m''-1,e'+e'',r'+r'')} Q\langle y\rangle @ [y \leftarrow \mathcal{H}\langle u\rangle] : M}\ \text{ES} \qquad (\Pi \uplus \Delta)(x) = \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m''-1,e'+e'',r'+r'')} (Q\langle y\rangle @ [y \leftarrow \mathcal{H}\langle u\rangle]) @ [x \leftarrow s] : M}\ \text{ES}_{\text{gc}}$$

noting that

$$(Q\langle y, \epsilon\rangle @ [y \leftarrow \mathcal{H}\langle u\rangle]) @ [x \leftarrow s] = (Q\langle y\rangle @ [y \leftarrow \mathcal{H}])\,\langle u, [x \leftarrow s]\rangle = P\langle u, [x \leftarrow s]\rangle$$

and that $(m' + m'' - 1, e' + e'', r' + r'') = (m - 1, e, r)$.

– Suppose Φ is of the form

$$\frac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} Q\langle y\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle y\rangle @ [y \leftarrow \mathcal{H}\langle(\lambda x.u)s\rangle] : M}\ \mathrm{ES_{gc}}$$

Note that $\mathsf{inert}_\Gamma(\mathcal{W} \cup \mathsf{nv}(\mathcal{H}))$ implies that $\mathsf{inert}_\Gamma(\mathcal{W})$, recalling that $Q \in \mathcal{E}_\mathcal{W}$. But then Lemma 7.1.1 (Relevance of the Open CbNeed type system) on Ψ gives that $y \in \mathsf{dom}(\Gamma)$—absurd.

2. We prove this by means of a weaker statement:

Let $p = P\langle x\rangle \to_{\mathsf{oe}} P\langle v^\alpha\rangle = q$, with $P \in \mathcal{E}_\mathcal{V}$ and $x \in \mathsf{dom}(P)$, and let $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} P\langle x\rangle : M$ such that $\mathsf{inert}_\Gamma(\mathcal{V})$. Then $e \geq 1$ and there exists $\Phi' \rhd_O \Gamma \vdash^{(m,e-1,r)} P\langle v^\alpha\rangle : M$.

We proceed by induction on the derivation of $P \in \mathcal{E}_\mathcal{V}$.

- The case where $P = (\mathcal{H}, \epsilon)$ is impossible because $x \notin \mathsf{dom}(P) = \emptyset$.
- Let

$$\frac{Q \in \mathcal{E}_\mathcal{V} \quad y \notin \mathcal{V}}{Q@[y \leftarrow t] \in \mathcal{E}_\mathcal{V}}\ \mathsf{O_{GC}}$$

We proceed by case analysis on the last typing rule in Φ and on whether $x = y$ or $x \neq y$.

– Let $x \neq y$ and Φ be of the form

$$\frac{\Psi \rhd_O \Pi; y : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle @ [y \leftarrow t] : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. Note that $\mathsf{inert}_{\Pi \biguplus \Delta}(P)$ together with the fact that $y \notin \mathcal{W}$ imply $\mathsf{inert}_{\Pi; y:N}(\mathcal{W})$. Moreover, note that $x \in \mathsf{dom}(Q)$, because $x \in \mathsf{dom}(P)$ and $x \neq y$. Hence, we can apply the *i.h.* on Ψ to get that $e' \geq 1$ (hence $e \geq 1$) and get

$$\Psi' \rhd_O \Pi; y : N \vdash^{(m',e'-1,r')} Q\langle v^\alpha\rangle : M$$

We can then derive Φ' as follows

$$\frac{\Psi' \qquad \Theta}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e''-1,r'+r'')} Q\langle v^\alpha\rangle @ [y \leftarrow t] : M}\ \mathrm{ES}$$

noting that $(m' + m'', e' + e'' - 1, r' + r'') = (m, e - 1, r)$.

– Let $x \neq y$ and Φ be of the form

$$\frac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} Q\langle x\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle x\rangle @ [y \leftarrow t] : M}\ \mathrm{ES_{gc}}$$

Note that $x \in \mathsf{dom}(Q)$. Since $\mathsf{inert}_\Gamma(\mathcal{V})$, we can apply the *i.h.* on Ψ to obtain that $e \geq 1$ and that there exists $\Psi \rhd_O \Gamma \vdash^{(m,e,r)} Q\langle x\rangle : M$.

– Let $x = y$ and Φ be of the form

$$\frac{\Psi \rhd_O \Pi; x : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} v : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle @ [x \leftarrow v] : M}\ \mathrm{ES}$$

with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. Note that $N \neq \mathbf{0}$, that $M \neq \mathbf{0}$—because Φ is tight—that $x \notin \mathsf{dom}(Q)$—because of the variable convention—that $x = y \notin \mathcal{V}$, and that $\mathsf{inert}_\Gamma(\mathcal{V})$. Hence, we can apply Lemma 7.1.4.2 (Linear Substitution for Open CbNeed) on Ψ to obtain a splitting of the form $N = N_1 \uplus N_2$, with $N_1 \neq \mathbf{0}$ such that for every $\Xi \rhd_O \Sigma \vdash^{(m''', e''', r''')} v : N_1$ there exists

$$\Omega \rhd_O (\Pi \biguplus \Delta_1); x : N_2 \vdash^{(m'+m''', e'+e'''-1, r'+r''')} Q\langle v^\alpha \rangle : M$$

We now apply Lemma 13.4.6 (Splitting multi types of Open CbNeed type derivations) on Θ with said splitting, obtaining type derivations $\Theta_1 \rhd_O \Delta_1 \vdash^{(m_1'', e_1'', r_1'')} v : N_1$ and $\Theta_2 \rhd_O \Delta_2 \vdash^{(m_2'', e_2'', r_2'')} v : N_2$.
Therefore, by what is given by Lemma 7.1.4.2 (Linear Substitution for Open CbNeed), there exists

$$\Omega \rhd_O (\Pi \biguplus \Delta_1); x : N_2 \vdash^{(m'+m_1'', e'+e_1''-1, r'+r_1'')} Q\langle v^\alpha \rangle : M$$

We finally do case analysis on whether $N_2 = \mathbf{0}$ or not:

* Let $N_2 = \mathbf{0}$. By Lemma 13.4.6 (Splitting multi types of Open CbNeed type derivations), this means that $\Delta_1 = \Delta$, $\Delta_2 = \mathbf{0}$, $(m_1'', e_1'', r_1'') = (m'', e'', r'')$ and $(m_2'', e_2'', r_2'') = (0, 0, 0)$. We can then derive Φ' as follows

$$\frac{\Omega \qquad (\Pi \biguplus \Delta_1)(x) = \mathbf{0}}{\Pi \biguplus \Delta_1 \vdash^{(m'+m_1'', e'+e_1''-1, r'+r_1'')} Q\langle v^\alpha \rangle @ [x \leftarrow v] : M} \ \mathrm{ES}_{\mathrm{gc}}$$

  noting that $\Pi \biguplus \Delta_1 = \Pi \biguplus \Delta = \Gamma$, that $Q\langle v^\alpha \rangle @ [x \leftarrow v] = P\langle v^\alpha, \epsilon \rangle$ and that

$$\begin{array}{rl} & (m' + m_1'', e' + e_1'' - 1, r' + r_1'') \\ = & (m' + m'', e' + e'' - 1, r' + r'') \\ = & (m, e - 1, r) \end{array}$$

* Let $N_2 \neq \mathbf{0}$. We can then derive Φ' as follows

$$\frac{\Omega \qquad \Theta_2 \qquad N_2 \neq \mathbf{0}}{\Pi \biguplus \Delta_1 \biguplus \Delta_2 \vdash^{(m'+m_1''+m_2'', e'+e_1''-1+e_2'', r'+r_1''+r_2'')} Q\langle v^\alpha \rangle @ [x \leftarrow v] : M} \ \mathrm{ES}$$

  noting that $\Pi \biguplus \Delta_1 \biguplus \Delta_2 = \Pi \biguplus \Delta = \Gamma$, that $Q\langle v^\alpha \rangle @ [x \leftarrow v] = P\langle v^\alpha \rangle$ and that

$$\begin{array}{rl} & (m' + m_1'' + m_2'', e' + e_1'' - 1 + e_2'', r' + r_1'' + r_2'') \\ = & (m' + m'', e' + e'' - 1, r' + r'') \\ = & (m, e - 1, r) \end{array}$$

– Suppose $x = y$ and Φ is of the form

$$\frac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} Q\langle y \rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle y \rangle @ [y \leftarrow t] : M} \ \mathrm{ES}_{\mathrm{gc}}$$

But then Lemma 13.4.4.2 (Plugged variables and domain of type contexts) on Ψ gives that $y \in \mathsf{dom}(\Gamma)$—absurd.

- Let
$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \in \mathcal{W}}{Q@[y{\leftarrow}i] \in \mathcal{E}_{(\mathcal{W}\setminus\{y\})\cup \mathsf{nv}(i)}}\ \mathsf{O_I}$$
where $\mathcal{V} = (\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(i)$. Note that $x \neq y$, because values are not inert terms. We proceed by case analysis on the last typing rule in Φ.
  - Let Φ be of the form
$$\frac{\Psi \rhd_O \Pi; y : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \rhd_O \Delta \vdash^{(m'',e'',r'')} i : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle@[y{\leftarrow}i] : M}\ \mathrm{ES}$$
where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$.
Note that $N = \mathsf{Inert}$—by Lemma 7.1.2.2 (Typing properties of normal terms - inert terms). Hence, $\mathsf{inert}_{\Pi \biguplus \Delta}((\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(i))$ implies $\mathsf{inert}_{\Pi; y : N}(\mathcal{W})$, and so we can apply the *i.h.* on Ψ to get that $e' \geq 1$ (hence $e \geq 1$) and obtain $\Psi' \rhd_O \Pi; y : N \vdash^{(m',e'-1,r')} Q\langle x\rangle : M$.
We can then derive Φ' as follows
$$\frac{\Psi' \qquad \Theta}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'-1+e'',r'+r'')} Q\langle v^\alpha\rangle@[y{\leftarrow}i] : M}\ \mathrm{ES}$$
noting that $\Pi \biguplus \Delta = \Gamma$, that $Q\langle v^\alpha\rangle@[y{\leftarrow}i] = P\langle v^\alpha\rangle$ and that $(m' + m'', e' - 1 + e'', r' + r'') = (m, e - 1, r)$.
  - Let Φ be of the form
$$\frac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} Q\langle x\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle x\rangle@[y{\leftarrow}i] : M}\ \mathrm{ES_{gc}}$$
Note that since $y \notin \mathsf{dom}(\Gamma)$ and $\mathsf{inert}_\Gamma((\mathcal{W} \setminus \{y\}) \cup \mathsf{nv}(i))$ then $\mathsf{inert}_\Gamma(\mathcal{W})$. Thus, we can apply the *i.h.* on Ψ, getting that $m \geq 1$ and the existence of $\Psi' \rhd_O \Gamma \vdash^{(m,e-1,r)} Q\langle v^\alpha\rangle : M$. We can then derive Φ' as follows
$$\frac{\Psi' \qquad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e-1,r)} Q\langle v^\alpha\rangle@[y{\leftarrow}i] : M}\ \mathrm{ES_{gc}}$$
noting that $Q\langle v^\alpha\rangle@[y{\leftarrow}i] = P\langle v^\alpha\rangle$.
- Suppose Φ is of the form
$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \notin \mathcal{W}}{Q\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{W}\cup \mathsf{nv}(\mathcal{H})}}\ \mathsf{O_{HER}}$$
where $\mathcal{V} = \mathcal{W} \cup \mathsf{nv}(\mathcal{H})$. Then, $x \notin \mathsf{dom}(P)$ by the variable convention—absurd.

□

(Click here to go back to main chapter.)

Theorem 13.4.9 (Tight Correctness for Open CbNeed)**.**
Let $p \in \mathcal{PR}$ and $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} p : M$ be a tight type derivation. Then there exists $q \in \mathcal{PR}$ such that

1. *q is in $\to_{\mathsf{ond}}$-normal form,*
2. *there exists a reduction sequence $d : p \to^*_{\mathsf{ond}} q$,*

3. $(m, e, r) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})$, *and*
4. $\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$.

Proof. (Click here to go back to main chapter.)

By induction on $m + e$, and proceeding by case analysis on whether $p \to_{\mathsf{ond}}$-reduces or not.

First, note that if p is in $\to_{\mathsf{ond}}$-normal form, then we can apply Proposition 7.1.3 (Typing properties of Open CbNeed-normal forms) to get that $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$ and $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.

Otherwise, if $p \to_{\mathsf{ond}} q$ for some $q \in \mathcal{PR}$, then there are two sub-cases to consider:

1. *Multiplicative steps*: Let $p \to_{\mathsf{om}} r$. By Proposition 7.1.6.1 (Quantitative Subject Reduction for Open CbNeed - multiplicative steps), there exists $\Psi \triangleright_O \Gamma \vdash^{(m-1,e,r)} r : M$. By *i.h.* on Ψ, there exists $q \in \mathcal{PR}$ such that
   (a) q is in $\to_{\mathsf{ond}}$-normal form,
   (b) there exists a reduction sequence $d' : r \to^*_{\mathsf{ond}} q$,
   (c) $(m - 1, e, r) = (|d'|_{\mathsf{m}}, |d'|_{\mathsf{e}}, |q|_{\mathsf{nd}})$, and
   (d) $\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$.

   Since $p \to_{\mathsf{om}} r$, we see that there exists $\to_{\mathsf{ond}}$-normalizing reduction sequence d given as follows

$$d \,:\, p \;\to^*_{\mathsf{ond}} q = p \to_{\mathsf{om}} \underbrace{r \;\to^*_{\mathsf{ond}} q}_{d'}$$

   and so the statement is satisfied. In particular, note that

$$(m, e, r) = (|d'|_{\mathsf{m}} + 1, |d'|_{\mathsf{e}}, |q|_{\mathsf{nd}}) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})$$

2. *Exponential steps*: Let $p \to_{\mathsf{oe}} r$. By Proposition 7.1.6.2 (Quantitative Subject Reduction for Open CbNeed - exponential steps), there exists $\Psi \triangleright_O \Gamma \vdash^{(m,e-1,r)} r : M$. By *i.h.* on Ψ, there exists $q \in \mathcal{PR}$ such that
   (a) q is in $\to_{\mathsf{ond}}$-normal form,
   (b) there exists a reduction sequence $d' : r \to^*_{\mathsf{ond}} q$,
   (c) $(m, e - 1, r) = (|d'|_{\mathsf{m}}, |d'|_{\mathsf{e}}, |q|_{\mathsf{nd}})$, and
   (d) $\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$.

   Since $p \to_{\mathsf{oe}} r$, we see that there exists $\to_{\mathsf{ond}}$-normalizing reduction sequence d given as follows

$$d \,:\, p \;\to^*_{\mathsf{ond}} q = p \to_{\mathsf{oe}} \underbrace{r \;\to^*_{\mathsf{ond}} q}_{d'}$$

   and so the statement is satisfied. In particular, note that

$$(m, e, r) = (|d'|_{\mathsf{m}}, |d'|_{\mathsf{e}} + 1, |q|_{\mathsf{nd}}) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})$$

□

(Click here to go back to main chapter.)

13.4.2 Open CbNeed completeness

Lemma 13.4.10 (Tight typability of normal terms)**.**

1. Values*: For every* $v \in \mathsf{Val}$ *there exists a type derivation* $\Phi \rhd_O \Gamma \vdash^{(0,0,|v|_{\mathsf{nd}})} v : [\mathsf{abs}]$ *such that* $\mathsf{dom}(\Gamma) = \mathsf{nv}(v)$.
2. Inert terms*: For every inert* Λ*-term* i *and* $J \neq \emptyset$*, there exists a tight type derivation* $\Phi \rhd_O \Gamma \vdash^{(0,0,|i|_{\mathsf{nd}})} i : [\mathsf{inert}]_{j \in J}$ *such that* $\mathsf{dom}(\Gamma) = \mathsf{nv}(i)$.

Proof. (Click here to go back to main chapter.)

1. *Values*: Simply take Φ as follows:

$$\dfrac{\dfrac{}{\emptyset \vdash^{(0,0,0)} v : \mathsf{abs}}\ \mathsf{abs}}{\emptyset \vdash^{(0,0,0)} v : [\mathsf{abs}]}\ \mathsf{many}$$

2. *Inert* Λ*-terms*: By structural induction on i:
   - If $i = x$, take Φ to be
   $$\dfrac{}{x : [\mathsf{inert}]_{j \in J} \vdash^{(0,0,0)} x : [\mathsf{inert}]_{j \in J}}\ \mathsf{ax}_I$$
   - Let $i = tu$ for some inert Λ-term t and normal term u. Note that $|i|_{\mathsf{nd}} = |t|_{\mathsf{nd}} + |u|_{\mathsf{nd}} + 1$ and that $\mathsf{nv}(i) = \mathsf{nv}(t) \cup \mathsf{nv}(u)$. By *i.h.* on t, there exists a tight type derivation of the form $\Psi_J \rhd_O \Pi_J \vdash^{(0,0,|t|_{\mathsf{nd}})} t : [\mathsf{inert}]_{j \in J}$ such that $\mathsf{dom}(\Pi_J) = \mathsf{nv}(t)$, for every set of indices J. There are two possible cases for the shape of u:
     - If u is an inert Λ-term, then we can apply ih on u and any singleton set of indices K to obtain a tight type derivation $\Theta \rhd_O \Delta \vdash^{(0,0,|u|_{\mathsf{nd}})} u : M$ such that $M = [\mathsf{inert}]$ and $\mathsf{dom}(\Delta) = \mathsf{nv}(u)$.
     - If $u \in \mathsf{Val}$, then we can apply Lemma 7.1.8.1 (Tight typability of normal terms - values) on it to obtain a tight type derivation $\Theta \rhd_O \Delta \vdash^{(0,0,|u|_{\mathsf{nd}})} u : M$ such that $M = [\mathsf{abs}]$ and $\mathsf{dom}(\Delta) = \mathsf{nv}(u)$.

     Therefore, in either case, we can derive Φ for i as follows:
   $$\dfrac{\Psi_J \rhd_O \Pi_J \vdash^{(0,0,|t|_{\mathsf{nd}})} t : [\mathsf{inert}]_{j \in J} \quad \Theta \rhd_O \Delta \vdash^{(0,0,|u|_{\mathsf{nd}})} u : M}{\Pi_J \uplus \Delta \vdash^{(0,0,|t|_{\mathsf{nd}}+|u|_{\mathsf{nd}}+1)} tu : [\mathsf{inert}]_{j \in J}}\ \mathsf{app}_i$$

□

(Click here to go back to main chapter.)

Proposition 13.4.11 (Tight typability of Open CbNeed-normal forms)**.**
Let $p \in \mathcal{PR}$ *be such that* $\mathsf{onorm}(p)$*. Then there exists a tight type derivation* $\Phi \rhd_O \Gamma \vdash^{(0,0,|p|_{\mathsf{nd}})} p : M$ *such that* $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.

Proof. (Click here to go back to main chapter.)

We split the analysis on whether $\mathsf{abs}(p)$ or $\mathsf{inert}(p)$:

- *Abstraction programs*: We proceed by induction on the derivation of $\mathsf{abs}(p)$:
  - If $p = (v, \epsilon)$, then the statement holds by Lemma 7.1.8.1 (Tight typability of normal terms - values) and then obtaining Φ via typing rule Lift.
  - Let $p = q@[x \leftarrow t]$ and
  $$\dfrac{\mathsf{abs}(q)}{\mathsf{abs}(q@[x \leftarrow t])}\ \mathsf{A}_{\mathsf{GC}}$$

By *i.h.* on $\mathsf{abs}(q)$, there exists a tight type derivation $\Psi \rhd_O \Pi \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{abs}]$ such that $\mathsf{dom}(\Pi) = \mathsf{nv}(q)$. Since $\mathsf{nv}(q) = \emptyset$—easily provable by induction on the derivation of $\mathsf{abs}(q)$—then we can derive Φ for p as follows

$$\frac{\Psi \rhd_O \Pi \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{abs}] \quad \Pi(x) = \mathbf{0}}{\Pi \vdash^{(0,0,|q|_{\mathsf{nd}})} q@[x{\leftarrow}t] : [\mathsf{abs}]}\ \mathrm{ES}_{\mathrm{gc}}$$

- *Inert programs*: We proceed by induction on the derivation of $\mathsf{inert}(p)$:
  - Let $p = (i, \epsilon)$ for some inert Λ-term i. The statement holds by taking any singleton set of indices J, applying Lemma 7.1.8.2 (Tight typability of normal terms - inert terms) on i and J, and then applying typing rule Lift to obtain Φ for p.
  - Let $p = q@[x{\leftarrow}i]$ and
  $$\frac{\mathsf{inert}(q) \quad x \in \mathsf{nv}(q)}{\mathsf{inert}(q@[x{\leftarrow}i])}\ \mathsf{I_I}$$
  Note that $|q@[x{\leftarrow}i]|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |i|_{\mathsf{nd}}$ and $\mathsf{nv}(q@[x{\leftarrow}i]) = (\mathsf{nv}(q) \setminus \{x\}) \cup \mathsf{nv}(i)$ By *i.h.* on q, there exists a tight type derivation $\Psi \rhd_O \Pi \vdash^{(0,0,|p|_{\mathsf{nd}})} q : [\mathsf{inert}]$ such that $\mathsf{dom}(\Pi) = \mathsf{nv}(q)$. Note that $x \in \mathsf{nv}(q) = \mathsf{dom}(\Pi)$. Next, take $\Pi(x)$ as the set of indices J and apply Lemma 7.1.8.2 (Tight typability of normal terms - inert terms) on i to obtain a tight type derivation $\Theta \rhd_O \Delta \vdash^{(0,0,|i|_{\mathsf{nd}})} i : [\mathsf{inert}]_{j \in J}$ such that $\mathsf{dom}(\Delta) = \mathsf{nv}(i)$.
  Therefore, we can derive Φ for p as follows:
  $$\frac{\Psi \rhd_O (\Pi \setminus\!\setminus \{x\}); x : \Pi(x) \vdash^{(0,0,|p|_{\mathsf{nd}})} q : [\mathsf{inert}] \quad \Theta \rhd_O \Delta \vdash^{(0,0,|i|_{\mathsf{nd}})} i : [\mathsf{inert}]_{j \in J}}{(\Pi \setminus\!\setminus \{x\}) \uplus \Delta \vdash^{(0,0,|q|_{\mathsf{nd}}+|i|_{\mathsf{nd}})} q@[x{\leftarrow}i] : [\mathsf{inert}]}\ \mathrm{ES}$$
  - Let $p = q@[x{\leftarrow}t]$ and
  $$\frac{\mathsf{inert}(q) \quad x \notin \mathsf{nv}(q)}{\mathsf{inert}(q@[x{\leftarrow}t])}\ \mathsf{I_{GC}}$$
  Note that $|q@[x{\leftarrow}t]|_{\mathsf{nd}} = |q|_{\mathsf{nd}}$ and $\mathsf{nv}(q@[x{\leftarrow}t]) = \mathsf{nv}(q)$. By *i.h.* on $\mathsf{inert}_{\mathcal{V}}(q)$, there exists a tight type derivation $\Psi \rhd_O \Pi \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{inert}]$ such that $\mathsf{dom}(\Pi) = \mathsf{nv}(q)$. Hence, $x \notin \mathsf{dom}(\Pi)$ and so we can derive Φ for p as follows
  $$\frac{\Psi \rhd_O \Pi \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{inert}] \quad \Pi(x) = \mathbf{0}}{\Psi \rhd_O \Pi \vdash^{(0,0,|q|_{\mathsf{nd}})} q@[x{\leftarrow}t] : [\mathsf{inert}]}\ \mathrm{ES}_{\mathrm{gc}}$$

□

(Click here to go back to main chapter.)

Lemma 13.4.12 (Linear Removal for Open CbNeed)**.**
Let $x \in \mathsf{Var}$ and $v \in \mathsf{Val}$ such that $x \notin \mathsf{fv}(v)$.
1. Let $\mathcal{H}$ be such that $x \notin \mathsf{nv}(\mathcal{H})$, and let

$$\Phi \rhd_O \Gamma; x : M \vdash^{(m,e,r)} \mathcal{H}\langle v \rangle : N$$

be such that $N \neq \mathbf{0}$, and $\mathsf{inert}_{\Gamma}(\mathsf{nv}(\mathcal{H}))$.
Then there exist type derivations

$$\Psi \rhd_O \Pi \vdash^{(m',e',r')} v : O$$
$$\Theta \rhd_O \Delta; x : (M \uplus O) \vdash^{(m'',e'',r'')} \mathcal{H}\langle x \rangle : N$$

such that

- $\Gamma = \Pi \biguplus \Delta$, *and*
- $(m' + m'', e' + e'', r' + r'') = (m, e + 1, r)$.

2. *Let* $P \in \mathcal{E}_{\mathcal{V}}$ *be such that* $x \notin (\mathcal{V} \cup \mathsf{dom}(P))$ *and that* $\mathsf{fv}(v) \cap \mathsf{dom}(P) = \emptyset$. *Let, moreover,*

$$\Phi \triangleright_O \Gamma; x : M \vdash^{(m,e,r)} P\langle v \rangle : N$$

be such that $N \neq \mathbf{0}$, *and* $\mathsf{inert}_\Gamma(\mathcal{V})$.
Then there exist
- *multi type* O,
- *type derivation* $\Psi \triangleright_O \Pi \vdash^{(m',e',r')} v : O$, *and*
- *type derivation* $\Theta \triangleright_O \Delta; x : (M \uplus O) \vdash^{(m'',e'',r'')} P\langle x \rangle : N$

such that
- $\Gamma = \Pi \biguplus \Delta$, *and*
- $(m' + m'', e' + e'', r' + r'') = (m, e + 1, r)$.

Proof. (Click here to go back to main chapter.)

1. By induction on the shape of $\mathcal{H}$:
   - Let $\mathcal{H} \coloneqq \langle \cdot \rangle$. Since N is a multi type, then Φ must be of the following form

$$\frac{\left(\Phi_i \triangleright_O \Gamma_i \vdash^{(m_i,e_i,r_i)} v : L_i\right)_{i \in I}}{\biguplus_{i \in I} \Gamma_i \vdash^{(\sum_{i \in I} m_{i \in I}, \sum_{i \in I} e_i, \sum_{i \in I} r_i)} v : [L_i]_{i \in I}}\ \mathsf{many}$$

   where $\Gamma = \biguplus_{i \in I} \Gamma_i$, $N = [L_i]_{i \in I}$ and $(m, e, r) = (\sum_{i \in I} m_{i \in I}, \sum_{i \in I} e_i, \sum_{i \in I} r_i)$. Note that since values are not inert-typable, then we have that $[L_i]_{i \in I} \notin \mathsf{Inert}$.
   The statement follows by taking $\Psi \coloneqq \Phi$ and Θ as follows

$$\frac{[L_i]_{i \in I} \notin \mathsf{Inert}}{x : [L_i]_{i \in I} \vdash^{(0,1,0)} x : [L_i]_{i \in I}}\ \mathsf{ax}$$

   - Let $\mathcal{H} \coloneqq \mathcal{J}t$. We proceed by case analysis on the last typing rule in Φ:
     - Let Φ be derived as

$$\frac{\Gamma_1; x : M_1 \vdash^{(m_1,e_1,r_1)} \mathcal{J}\langle v \rangle : [P \multimap N] \quad \Gamma_2; x : M_2 \vdash^{(m_2,e_2,r_2)} t : P \quad P \neq \mathbf{0}}{(\Gamma_1 \biguplus \Gamma_2); x : (M_1 \uplus M_2) \vdash^{(m_1+m_2+1, e_1+e_2, r_1+r_2)} \mathcal{J}\langle v \rangle t : N}\ \mathsf{app}$$

     with $\Gamma = \Gamma_1 \biguplus \Gamma_2$, $M = M_1 + M_2$ and $(m, e, r) = (m_1 + m_2 + 1, e_1 + e_2, r_1 + r_2)$.
     Since $\mathsf{nv}(\mathcal{H}) = \mathsf{nv}(\mathcal{J})$, then we can infer from $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$ that $\mathsf{inert}_{\Gamma_1}(\mathsf{nv}(\mathcal{J}))$. Hence, we can apply the *i.h.* on Ψ to obtain type derivations $\Psi' \triangleright_O \Pi' \vdash^{(m'_1,e'_1,r'_1)} v : O'$ and $\Theta' \triangleright_O \Delta'; x : (M_1 \uplus O') \vdash^{(m''_1,e''_1,r''_1)} \mathcal{J}\langle x \rangle : [P \multimap N]$ such that $\Gamma_1 = \Pi' \biguplus \Delta'$ and $(m'_1 + m''_1, e'_1 + e''_1, r'_1 + r''_1) = (m_1, e_1 + 1, r_1)$.
     Therefore, the statement follows by taking $\Psi \coloneqq \Psi'$ and deriving Θ as follows

$$\frac{\Delta'; x : (M_1 \uplus O') \vdash^{(m''_1,e''_1,r''_1)} \mathcal{J}\langle x \rangle : [P \multimap N] \quad \Gamma_2; x : M_2 \vdash^{(m_2,e_2,r_2)} t : P}{(\Delta' \biguplus \Gamma_2); x : (M_1 \uplus O' \uplus M_2) \vdash^{(m''_1+m_2+1, e''_1+e_2, r''_1+r_2)} \mathcal{J}\langle x \rangle t : N}\ \mathsf{app}$$

     verifying that
     * $\Gamma = \Gamma_1 \biguplus \Gamma_2 = (\Pi' \biguplus \Delta') \biguplus \Gamma_2 = \Pi' \biguplus (\Delta' \biguplus \Gamma_2) = \Pi \biguplus \Delta$, and

$*$ $(m' + m'', e' + e'', r' + r'') = (m'_1 + (m''_1 + m_2 + 1), e'_1 + (e''_1 + e_2), r'_1 + (r''_1 + r_2)) = (m_1 + m_2 + 1, e_1 + e_2 + 1, r_1 + r_2) = (m, e + 1, r)$.

– The case where app_i is the last typing rule in Φ can be proven similarly to app.

– The case where $\mathsf{app}_{\mathrm{gc}}$ is the last typing rule in Φ can be proven similarly to the two previous cases.

- Let $\mathcal{H} = i\mathcal{J}$. We proceed by case analysis on the last typing rule in Φ:

– Suppose Φ were derived as follows:

$$\frac{\Xi \rhd_O \Gamma_1; x : M_1 \vdash^{(m_1,e_1,r_1)} i : [P \multimap N] \quad \Omega \rhd_O \Gamma_2; x : M_2 \vdash^{(m_2,e_2,r_2)} \mathcal{J}\langle v\rangle : P \quad P \neq \mathbf{0}}{(\Gamma_1 \uplus \Gamma_2); x : (M_1 \uplus M_2) \vdash^{(m_1+m_2+1,e_1+e_2,r_1+r_2)} i\mathcal{J}\langle v\rangle : N}\ \mathsf{app}$$

with $\Gamma = (\Gamma_1 \biguplus \Gamma_2)$. But since $x \notin \mathsf{nv}(\mathcal{H})$, then we would have that $x \notin \mathsf{nv}(i)$. Hence, $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$ would imply that $\mathsf{inert}_{\Gamma_1;x:M_1}(\mathsf{nv}(i))$. However, we would now be able to apply Lemma 7.1.2.2 (Typing properties of normal terms - Inert terms) on Ξ to obtain that $[P \multimap N] \in \mathsf{Inert}$, which is absurd. Hence, this case is impossible.

– Let Φ be derived as

$$\frac{\Xi \rhd_O \Gamma_1; x : M_1 \vdash^{(m_1,e_1,r_1)} i : [\mathsf{inert}]_{i\in I} \quad \Omega \rhd_O \Gamma_2; x : M_2 \vdash^{(m_2,e_2,r_2)} \mathcal{J}\langle v\rangle : [\mathsf{tight}] \quad I \neq \emptyset}{(\Gamma_1 \uplus \Gamma_2); x : (M_1 \uplus M_2) \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2+1)} i\mathcal{J}\langle v\rangle : [\mathsf{inert}]_{i\in I}}\ \mathsf{app}_i$$

with $\Gamma = (\Gamma_1 \biguplus \Gamma_2)$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2 + 1)$.
Note that $x \notin \mathsf{nv}(\mathcal{H})$, and so $x \notin \mathsf{nv}(\mathcal{J})$. This allows us to infer from $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$ that $\mathsf{inert}_{\Gamma_2;x:M_2}(\mathsf{nv}(\mathcal{J}))$. Hence, we can apply the *i.h.* on Ω to obtain type derivations

$$\Psi' \rhd_O \Pi' \vdash^{(m'_2,e'_2,r'_2)} v : O'$$
$$\Theta' \rhd_O \Delta'; x : (M_2 \uplus O') \vdash^{(m''_2,e''_2,r''_2)} \mathcal{J}\langle x\rangle : [\mathsf{tight}]$$

such that $\Gamma_2 = \Pi' \biguplus \Delta'$ and $(m'_2 + m''_2, e'_2 + e''_2, r'_2 + r''_2) = (m_2, e_2 + 1, r_2)$.
Therefore, the statement follows by taking $\Psi \coloneqq \Psi'$ and deriving Θ as follows

$$\frac{\Xi \rhd_O \Gamma_1; x : M_1 \vdash^{(m_1,e_1,r_1)} i : [\mathsf{inert}]_{i\in I} \quad \Theta' \rhd_O \Delta'; x : (M_2 \uplus O') \vdash^{(m''_2,e''_2,r''_2)} \mathcal{J}\langle x\rangle : [\mathsf{tight}]}{(\Gamma_1 \uplus \Delta'); x : (M_1 \uplus M_2 \uplus O') \vdash^{(m_1+m''_2,e_1+e''_2,r_1+r''_2+1)} i\mathcal{J}\langle x\rangle : [\mathsf{inert}]_{i\in I}}\ \mathsf{app}_i$$

verifying that

$*$ $\Gamma = \Gamma_1 \biguplus \Gamma_2 = \Gamma_1 \biguplus (\Pi' \biguplus \Delta') = (\Gamma_1 \biguplus \Delta') \biguplus \Pi' = \Delta \biguplus \Pi$, and

$*$ $(m' + m'', e' + e'', r' + r'') = (m'_2 + (m_1 + m''_2), e'_2 + (e_1 + e''_2), r'_2 + (r_1 + r''_2 + 1)) = (m_1 + m_2, e_1 + e_2 + 1, r_1 + r_2 + 1) = (m, e + 1, r)$.

– The case where $\mathsf{app}_{\mathrm{gc}}$ is the last typing rule in Φ is impossible, which can be proven similarly to the case of app.

2. By induction on the derivation of $P \in \mathcal{E}_\mathcal{V}$:

- Let $P \in \mathcal{E}_\mathcal{V}$ be derived as

$$\frac{}{(\mathcal{H}, \epsilon) \in \mathcal{E}_{\mathsf{nv}(\mathcal{H})}}\ \mathsf{O_{AX}}$$

where $P = (\mathcal{H}, \epsilon)$ and $\mathcal{V} = \mathsf{nv}(\mathcal{H})$. Then Φ must be of the form

$$\frac{\Phi' \rhd_O \Gamma; x : M \vdash^{(m,e,r)} \mathcal{H}\langle v\rangle : N}{\Gamma; x : M \vdash^{(m,e,r)} (\mathcal{H}\langle v\rangle, \epsilon) : N}\ \mathsf{Lift}$$

and we can apply Lemma 7.1.10.1 (Linear Removal for Open CbNeed) on Φ' to obtain type derivations

$$\Psi' \rhd_O \Pi' \vdash^{(m'_1,e'_1,r'_1)} v : O'$$
$$\Theta' \rhd_O \Delta'; x : (M \uplus O') \vdash^{(m''_1,e''_1,r''_1)} \mathcal{H}\langle x\rangle : N$$

such that $\Gamma = \Pi' \biguplus \Delta'$ and $(m'_1 + m''_1, e'_1 + e''_1, r'_1 + r''_1) = (m, e+1, r)$.
Therefore, the statement follows by taking $\Psi \coloneqq \Psi'$ and deriving Θ as follows

$$\frac{\Theta' \rhd_O \Delta'; x : (M \uplus O') \vdash^{(m''_1, e''_1, r''_1)} \mathcal{H}\langle x \rangle : N}{\Delta'; x : (M \uplus O') \vdash^{(m''_1, e''_1, r''_1)} \mathcal{H}\langle x \rangle : N}\ \mathsf{Lift}$$

- Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{V}} \quad y \in \mathcal{V}}{Q@[y \leftarrow i] \in \mathcal{E}_{(\mathcal{V} \setminus \{y\}) \cup \mathsf{nv}(i)}}\ \mathsf{O_I}$$

where $P = Q@[y \leftarrow i]$ and $\mathcal{V} = (\mathcal{V} \setminus \{y\}) \cup \mathsf{nv}(i)$. Note that $x \neq y$—since $x \notin \mathsf{dom}(P)$—and that $\mathsf{fv}(v) \cap \mathsf{dom}(P) = \emptyset$—by hypothesis—yielding that $y \notin \mathsf{fv}(v)$. We proceed by case analysis on the last typing rule in Φ:

  - Let Φ be of the form

$$\frac{\Xi \rhd_O \Gamma_1; y : P; x : M_1 \vdash^{(m_1, e_1, r_1)} Q\langle v \rangle : N \quad \Omega \rhd_O \Gamma_2; x : M_2 \vdash^{(m_2, e_2, r_2)} i : P \quad P \neq \mathbf{0}}{(\Gamma_1 \biguplus \Gamma_2); x : (M_1 \uplus M_2) \vdash^{(m_1 + m_2, e_1 + e_2, r_1 + r_2)} Q\langle v \rangle@[y \leftarrow i] : N}\ \mathrm{ES}$$

where $\Gamma = \Gamma_1 \biguplus \Gamma_2$, $M = M_1 \biguplus M_2$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$.
Note that $x \notin \mathcal{V}$, and so $\mathsf{inert}_\Gamma(\mathcal{V})$ implies that $\mathsf{inert}_{\Gamma_2; x : M_2}(\mathsf{nv}(i))$. Hence, we can apply Lemma 7.1.2.2 (Typing properties of normal term - Inert terms) on Ω to obtain that $P \in \mathsf{Inert}$. In turn, this allows us to infer from $\mathsf{inert}_\Gamma(\mathcal{V})$ that $\mathsf{inert}_{\Gamma_1; y : P}(\mathcal{W})$. Thus, we can apply the *i.h.* on Ξ to obtain type derivations

$$\begin{array}{l} \Psi' \rhd_O \Pi' \vdash^{(m'_1, e'_1, r'_1)} v : O' \\ \Theta' \rhd_O \Delta'; y : P; x : (M_1 \uplus O') \vdash^{(m''_1, e''_1, r''_1)} Q\langle x \rangle : N \end{array}$$

such that $\Gamma_1; y : P = \Pi' \biguplus (\Delta'; y : P)$ and $(m'_1 + m''_1, e'_1 + e''_1, r'_1 + r''_1) = (m_1, e+1, r_1)$.
Note that $x \notin \mathsf{dom}(\Pi')$—by Lemma 7.1.1 (Relevance of the Open CbNeed type system) and given that $x \notin \mathsf{fv}(v)$.
Therefore, the statement follows by taking $\Psi \coloneqq \Psi'$ and deriving Θ as follows

$$\frac{\Theta' \rhd_O \Delta'; y : P; x : (M_1 \uplus O') \vdash^{(m''_1, e''_1, r''_1)} Q\langle x \rangle : N \quad \Omega \rhd_O \Gamma_2; x : M_2 \vdash^{(m_2, e_2, r_2)} i : P}{(\Delta' \biguplus \Gamma_2); x : (M_1 \uplus O' \uplus M_2) \vdash^{(m''_1 + m_2, e''_1 + e_2, r''_1 + r_2)} Q\langle x \rangle@[y \leftarrow i] : N}\ \mathrm{ES}$$

verifying that
    - $M_1 \uplus O' \uplus M_2 = M \uplus O$,
    - $\Gamma = \Gamma_1 \biguplus \Gamma_2 = ((\Pi' \biguplus (\Delta'; y : P)) \setminus\setminus \{y\}) \biguplus \Gamma_2 = \Pi' \biguplus (\Delta' \biguplus \Gamma_2) = \Pi \biguplus \Delta$, and
    - $(m' + m'', e' + e'', r' + r'') = (m'_1 + (m''_1 + m_2), e'_1 + (e''_1 + e_2), r'_1 + (r''_1 + r_2)) = (m_1 + m_2, e_1 + 1 + e_2, r_1 + r_2) = (m, e+1, r)$.
  - Let Φ be of the form

$$\frac{\Phi' \rhd_O \Gamma; x : M \vdash^{(m,e,r)} Q\langle v \rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle v \rangle@[y \leftarrow i] : N}\ \mathrm{ES_{gc}}$$

Since hypothesis $\mathsf{inert}_\Gamma(\mathcal{V})$ and fact $y \notin \mathsf{dom}(\Gamma; x : M)$ togther imply that $\mathsf{inert}_\Gamma(\mathcal{W})$, then we can apply the *i.h.* on Φ' and easily prove the statement.

- Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as
$$\frac{Q \in \mathcal{E}_{\mathcal{V}} \quad y \notin \mathcal{V}}{Q@[y \leftarrow t] \in \mathcal{E}_{\mathcal{V}}}\ \mathsf{O_{GC}}$$
where $P = Q@[y \leftarrow t]$. Note that $x \neq y$—since $x \notin \mathsf{dom}(P)$—and that $\mathsf{fv}(v) \cap \mathsf{dom}(P) = \emptyset$—by hypothesis—yielding that $y \notin \mathsf{fv}(v)$. We proceed by case analysis on the last typing rule in Φ:
  - Let Φ be of the form
$$\frac{\Xi \rhd_O \Gamma_1; y : P; x : M_1 \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : N \quad \Omega \rhd_O \Gamma_2; x : M_2 \vdash^{(m_2,e_2,r_2)} t : P \quad P \neq \mathbf{0}}{(\Gamma_1 \biguplus \Gamma_2); x : (M_1 \uplus M_2) \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle v\rangle@[y \leftarrow t] : N}\ \mathrm{ES}$$
where $\Gamma = \Gamma_1 \biguplus \Gamma_2$, $M = M_1 \biguplus M_2$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$.
Note that the facts that $\mathsf{inert}_\Gamma(\mathcal{V})$ and that $y \notin \mathcal{V}$ together imply that $\mathsf{inert}_{\Gamma_1; y:P}(\mathcal{V})$.
Hence, we can apply the *i.h.* on Ξ to obtain type derivations
$$\begin{array}{l}\Psi' \rhd_O \Pi' \vdash^{(m'_1,e'_1,r'_1)} v : O \\ \Theta' \rhd_O \Delta'; y : P; x : (M_1 \uplus O') \vdash^{(m''_1,e''_1,r''_1)} Q\langle x\rangle : N\end{array}$$
such that $\Gamma_1; y : P = \Pi' \biguplus (\Delta'; y : P)$ and $(m'_1 + m''_1, e'_1 + e''_1, r'_1 + r''_1) = (m_1, e_1 + 1, r_1)$.
Note that $y \notin \mathsf{dom}(\Pi')$—by Lemma 7.1.1 (Relevance of the Open CbNeed type system) and by the fact that $y \notin \mathsf{fv}(v)$.
Therefore, the statement follows by taking $\Psi \coloneqq \Psi'$ and deriving Θ as follows
$$\frac{\Theta' \rhd_O \Delta'; y : P; x : (M_1 \uplus O') \vdash^{(m''_1,e''_1,r''_1)} Q\langle x\rangle : N \quad \Omega \rhd_O \Gamma_2; x : M_2 \vdash^{(m_2,e_2,r_2)} t : P}{(\Delta' \biguplus \Gamma_2); x : (M_1 \uplus O' \uplus M_2) \vdash^{(m''_1+m_2,e''_1+e_2,r''_1+r_2)} Q\langle x\rangle@[y \leftarrow t] : N}\ \mathrm{ES}$$
verifying that
    - $M_1 \uplus O' \uplus M_2 = M \uplus O$,
    - $\Gamma = \Gamma_1 \biguplus \Gamma_2 = ((\Pi' \biguplus (\Delta'; y : P)) \setminus\setminus \{y\}) \biguplus \Gamma_2 = \Pi' \biguplus (\Delta' \biguplus \Gamma_2) = \Pi \biguplus \Delta$, and
    - $(m' + m'', e' + e'', r' + r'') = (m'_1 + (m''_1 + m_2), e'_1 + (e''_1 + e_2), r'_1 + (r''_1 + r_2)) = (m_1 + m_2, e_1 + 1 + e_2, r_1 + r_2) = (m, e + 1, r)$.
  - Let Φ be of the form
$$\frac{\Phi' \rhd_O \Gamma; x : M \vdash^{(m,e,r)} Q\langle v\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle v\rangle@[y \leftarrow t] : N}\ \mathrm{ES_{gc}}$$
We can apply the *i.h.* on Φ' and easily prove the statement.
- Let $P \in \mathcal{E}_{\mathcal{V}}$ be derived as
$$\frac{Q \in \mathcal{E}_{\mathcal{V}} \quad y \notin \mathcal{V}}{Q\langle y\rangle@[y \leftarrow \langle\cdot\rangle] \in \mathcal{E}_{\mathcal{V}}}\ \mathsf{O_{HER}}$$
where $P = Q\langle y\rangle@[y \leftarrow \langle\cdot\rangle]$. We may safely assume that $x \neq y$—by α-conversion. We proceed by case analysis on the last typing rule in Φ:
  - Let Φ be of the form
$$\frac{\Xi \rhd_O \Gamma_1; x : M; y : P \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : N \quad \Omega \rhd_O \Gamma_2 \vdash^{(m_2,e_2,r_2)} v : P \quad P \neq \mathbf{0}}{(\Gamma_1 \biguplus \Gamma_2); x : M \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle y\rangle@[y \leftarrow v] : N}\ \mathrm{ES}$$
where $\Gamma = \Gamma_1 \biguplus \Gamma_2$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that $x \notin \mathsf{dom}(\Gamma_2)$, because $x \notin \mathsf{fv}(v)$—and applying Lemma 7.1.1 (Relevance of the Open CbNeed type system). Note that, since values are not inert-typable, then $P \notin \mathsf{Inert}$.

Therefore, the statement follows by taking $\Psi \coloneqq \Omega$ and deriving Θ as follow:

$$\dfrac{\Theta' \rhd_O \Gamma_1; x : M; y : P \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : N \quad \dfrac{P \notin \mathsf{Inert}}{x : P \vdash^{(0,1,0)} x : P}\ \mathsf{ax}}{\Gamma_1; x : (M \uplus P) \vdash^{(m_1,e_1+1,r_1)} Q\langle y\rangle @ [y \leftarrow x] : N}\ \mathrm{ES}$$

verifying that
- $M \uplus P = M \uplus O$,
- $\Gamma = \Gamma_1 \biguplus \Gamma_2 = \Pi \biguplus \Delta$
- $(m' + m'', e' + e'', r' + r'') = (m_2 + m_1, e_2 + e_1 + 1, r_2 + r_1) = (m, e + 1, r)$

– Suppose Φ is of the form

$$\dfrac{\Phi' \rhd_O \Gamma; x : M \vdash^{(m,e,r)} Q\langle y\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle y\rangle @ [y \leftarrow v] : N}\ \mathrm{ES_{gc}}$$

However, since $x \notin \mathcal{V} = \mathcal{W}$, then $\mathsf{inert}_\Gamma(\mathcal{V})$ would imply that $\mathsf{inert}_\Gamma(\mathcal{W})$, and so we would be able to apply Lemma 13.4.4.2 (Plugged variables and domain of type contexts) on Φ' to obtain that $y \in \mathsf{dom}(\Gamma; x : M)$. Since that is absurd, then so is this case.

□

(Click here to go back to main chapter.)

The following is required to apply Lemma 7.1.10 (Linear Removal for Open CbNeed) in the proof of Proposition 7.1.12.2 (Quantitative Subject Expansion for Open CbNeed - exponential case) to obtain the right indices.

Lemma 13.4.13 (Merging multi types of Open CbNeed type derivations)**.**
Let $v \in \mathsf{Val}$. For any two type derivations

$$\Phi_N \rhd_O \Gamma_N \vdash^{(m_N,e_N,r_N)} v : N$$
$$\Phi_O \rhd_O \Gamma_O \vdash^{(m_O,e_O,r_O)} v : O$$

there exists type derivation

$$\Phi_N \rhd_O \Gamma_N \biguplus \Gamma_O \vdash^{(m_N+m_O,e_N+e_O,r_N+r_O)} v : N \uplus O$$

Proof.
Trivial, given that the many typing rule is the only one that can derive a multi type on the right—*i.e.*, the derived type of the subject, in this case M—for values.

□

Lemma 13.4.14 (Quantitative Subject Expansion for $\to_{\mathsf{om}}$ in term contexts)**.**
Let $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} (\mathcal{H}\langle u\rangle, [x \leftarrow s]) : M$ such that $\mathsf{inert}_\Gamma(\mathsf{nv}(\mathcal{H}))$. Then there exists $\Phi' \rhd_O \Gamma \vdash^{(m+1,e,r)} \mathcal{H}\langle(\lambda x.u)s\rangle : M$.

Proof. (Click here to go back to main chapter.)
By structural induction on $\mathcal{H}$:
- *Empty context*: Let $\mathcal{H} \coloneqq \langle\cdot\rangle$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\dfrac{\dfrac{\Psi \rhd_O \Pi; x : N \vdash^{(m_1,e_1,r_1)} u : M}{\Pi; x : N \vdash^{(m_1,e_1,r_1)} (u, \epsilon) : M}\,\mathsf{Lift} \qquad \Theta \rhd_O \Delta \vdash^{(m_2,e_2,r_2)} s : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} (u, [x \leftarrow s]) : M}\,\mathrm{ES}$$

with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. We can then derive Φ' as follows

$$\dfrac{\dfrac{\dfrac{\Psi \rhd_O \Pi; x : N \vdash^{(m_1,e_1,r_1)} u : M}{\Pi \vdash^{(m_1,e_1,r_1)} \lambda x.u : N \multimap M}\,\mathsf{fun}}{\Pi \vdash^{(m_1,e_1,r_1)} \lambda x.u : [N \multimap M]}\,\mathsf{many} \qquad \Theta \rhd_O \Delta \vdash^{(m_2,e_2,r_2)} s : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2+1,e_1+e_2,r_1+r_2)} (\lambda x.u)s : M}\,\mathsf{app}$$

verifying that $(m_1 + m_2 + 1, e_1 + e_2, r_1 + r_2) = (m, e + 1, r)$.

– Let Φ be of the form

$$\dfrac{\dfrac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} u : M}{\Gamma \vdash^{(m,e,r)} (u, \epsilon) : M}\,\mathsf{Lift} \qquad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} (u, [x \leftarrow s]) : M}\,\mathrm{ES}_{\mathrm{gc}}$$

We can then derive Φ' as follows

$$\dfrac{\dfrac{\dfrac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} u : M}{\Gamma \vdash^{(m,e,r)} \lambda x.u : \mathbf{0} \multimap M}\,\mathsf{fun}}{\Gamma \vdash^{(m,e,r)} \lambda x.u : [\mathbf{0} \multimap M]}\,\mathsf{many}}{\Gamma \vdash^{(m+1,e,r)} (\lambda x.u)s : M}\,\mathsf{app}_{\mathrm{gc}}$$

- *Application left*: Let $\mathcal{H} \coloneqq \mathcal{J}t$. We proceed by case analysis on the last typing rule in Φ:
  – Let Φ be of the form

$$\dfrac{\dfrac{\Psi \rhd_O \Pi; x : N \vdash^{(m_1,e_1,r_1)} \mathcal{J}\langle u \rangle t : M}{\Pi; x : N \vdash^{(m_1,e_1,r_1)} (\mathcal{J}\langle u \rangle t, \epsilon) : M}\,\mathsf{Lift} \qquad \Theta \rhd_O \Delta \vdash^{(m_2,e_2,r_2)} s : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} (\mathcal{J}\langle u \rangle t, [x \leftarrow s]) : M}\,\mathrm{ES}$$

with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. We proceed by case analysis on the last typing rule in Ψ:

  * Let Ψ be of the form

$$\dfrac{\Xi \rhd_O \Sigma; x : N \vdash^{(m',e',r')} \mathcal{J}\langle u \rangle : [P \multimap M] \quad \Omega \rhd_O T \vdash^{(m'',e'',r'')} t : P}{(\Sigma \biguplus T); x : N \vdash^{(m'+m'',e'+e'',r'+r'')} \mathcal{J}\langle u \rangle t : M}\,\mathsf{app}$$

with $\Pi = \Sigma \biguplus T$ and $(m' + m'', e' + e'', r' + r'') = (m_1, e_1, r_1)$. By *alpha*-conversoin, we may safely assume that $x \notin \mathsf{fv}(t)$, and then we may safely assume as well that $x \notin \mathsf{dom}(T)$—by Lemma 7.1.1 (Relevance of the Open CbNeed type system).
We can now derive an auxiliary type derivation Z as follows

$$\dfrac{\dfrac{\Xi \rhd_O \Sigma; x : N \vdash^{(m',e',r')} \mathcal{J}\langle u \rangle : [P \multimap M]}{\Sigma; x : N \vdash^{(m',e',r')} (\mathcal{J}\langle u \rangle, \epsilon) : [P \multimap M]}\,\mathsf{Lift} \qquad \Theta \rhd_O \Delta \vdash^{(m_2,e_2,r_2)} s : N}{\Sigma \biguplus \Delta \vdash^{(m'+m_2,e'+e_2,r'+r_2)} (\mathcal{J}\langle u \rangle, [x \leftarrow s]) : [P \multimap M]}\,\mathrm{ES}$$

on which we can apply the *i.h.* to obtain

$$Z' \rhd_O \Sigma \biguplus \Delta \vdash^{(m'+m_2+1,e'+e_2,r'+r_2)} \mathcal{J}\langle(\lambda x.u)s\rangle : M$$

We can finally derive Φ' as follows

$$\frac{Z' \rhd_O \Sigma \biguplus \Delta \vdash^{(m'+m_2+1,e'+e_2,r'+r_2)} \mathcal{J}\langle(\lambda x.u)s\rangle : [P \multimap M] \quad \Omega \rhd_O T \vdash^{(m'',e'',r'')} t : P}{\Sigma \biguplus \Delta \biguplus T \vdash^{(m'+m_2+1+m'',e'+e_2+e'',r'+r_2+r'')} \mathcal{J}\langle(\lambda x.u)s\rangle : [P \multimap M]}\ \mathsf{app}$$

verifying that
- $\Sigma \biguplus \Delta \biguplus T = \Pi \biguplus \Delta = \Gamma$, and
- $(m' + m_2 + 1 + m'', e' + e_2 + e'', r' + r_2 + r'') = (m_1 + m_2 + 1, e_1 + e_2, r_1 + r_2) = (m + 1, e, r)$.

$*$ Let Φ be of the form

$$\frac{\dfrac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} \mathcal{J}\langle u\rangle t : M}{\Gamma \vdash^{(m,e,r)} (\mathcal{J}\langle u\rangle t, \epsilon) : M}\ \mathsf{Lift} \qquad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} (\mathcal{J}\langle u\rangle t, [x \leftarrow s]) : M}\ \mathrm{ES_{gc}}$$

We proceed by case analysis on the last typing rule in Φ. The cases where app_i and $\mathsf{app}_{\mathrm{gc}}$ are proven similarly, and are left to the reader: Let Ψ be derived as

$$\frac{\Theta \rhd_O \Gamma_1 \vdash^{(m_1,e_1,r_1)} \mathcal{J}\langle u\rangle : [P \multimap M] \quad \Xi \rhd_O \Gamma_2 \vdash^{(m_2,e_2,r_2)} t : P}{\Gamma_1 \biguplus \Gamma_2 \vdash^{(m_1+m_2+1,e_1+e_2,r_1+r_2)} \mathcal{J}\langle u\rangle t : M}\ \mathsf{app}$$

with $\Gamma = \Gamma_1 \biguplus \Gamma_2$ and $(m, e, r) = (m_1 + m_2 + 1, e_1 + e_2, r_1 + r_2)$.
We can now derive an auxiliary type derivation Z as follows

$$\frac{\dfrac{\Theta \rhd_O \Gamma_1 \vdash^{(m_1,e_1,r_1)} \mathcal{J}\langle u\rangle : [P \multimap M]}{\Gamma_1 \vdash^{(m_1,e_1,r_1)} (\mathcal{J}\langle u\rangle, \epsilon) : [P \multimap M]}\ \mathsf{Lift} \qquad \Gamma_1(x) = \mathbf{0}}{\Gamma_1 \vdash^{(m_1,e_1,r_1)} (\mathcal{J}\langle u\rangle, [x \leftarrow s]) : [P \multimap M]}\ \mathrm{ES_{gc}}$$

on which we can apply the *i.h.* to obtain $Z' \rhd_O \Gamma_1 \vdash^{(m_1+1,e_1,r_1)} \mathcal{J}\langle(\lambda x.u)s\rangle : [P \multimap M]$, and finally derive Φ' as follows

$$\frac{Z' \rhd_O \Gamma_1 \vdash^{(m_1+1,e_1,r_1)} \mathcal{J}\langle(\lambda x.u)s\rangle : [P \multimap M] \quad \Omega \rhd_O \Gamma_2 \vdash^{(m_2,e_2,r_2)} t : P}{\Gamma_1 \biguplus \Gamma_2 \vdash^{(m_1+1+m_2+1,e_1+e_2,r_1+r_2)} \mathcal{J}\langle(\lambda x.u)s\rangle t : M}\ \mathsf{app}$$

verifying that $(m_1 + 1 + m_2 + 1, e_1 + e_2, r_1 + r_2) = (m + 1, e, r)$.

- *Application right*: Let $\mathcal{H} := i\mathcal{J}$. By α-conversion may safely assume that $x \notin \mathsf{fv}(i)$. We proceed by case analysis on the last typing rule in Φ:
  - Let Φ be of the form

$$\frac{\dfrac{\Psi \rhd_O \Pi; x : N \vdash^{(m_1,e_1,r_1)} i\mathcal{J}\langle u\rangle : M}{\Pi; x : N \vdash^{(m_1,e_1,r_1)} (i\mathcal{J}\langle u\rangle, \epsilon) : M}\ \mathsf{Lift} \qquad \Theta \rhd_O \Delta \vdash^{(m_2,e_2,r_2)} s : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} (i\mathcal{J}\langle u\rangle, [x \leftarrow s]) : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \uplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. By proceeding by case analysis on the last typing rule in Φ, it is easy to verify that the sub-type derivation corresponding to t can be applied with Lemma 7.1.2.2 (Typing properties of normal terms - Inert terms) to infer that $M \in \mathsf{Inert}$. Thus, let $M = [\mathsf{inert}]_{i\in I}$, and note that Ψ must be of the form

$$\frac{\Xi \rhd_O \Pi_1 \vdash^{(m'_1,e'_1,r'_1)} i : [\mathsf{inert}]_{i\in I} \quad \Omega \rhd_O \Pi_2; x : N \vdash^{(m''_1,e''_1,r''_1)} \mathcal{J}\langle u\rangle : [\mathsf{tight}]}{(\Pi_1 \uplus \Pi_2); x : N \vdash^{(m'_1+m''_1,e'_1+e''_1,r'_1+r''_1+1)} i\mathcal{J}\langle u\rangle : [\mathsf{inert}]_{i\in I}}\ \mathsf{app}_i$$

where $\Pi = \Pi_1 \uplus \Pi_2$ and $(m_1, e_1, r_1) = (m'_1 + m''_1, e'_1 + e''_1, r'_1 + r''_1 + 1)$. Note that $x \notin \mathsf{dom}(\Pi_1)$ because $x \notin \mathsf{fv}(t)$—and by an application of Lemma 7.1.1 (Relevance of the Open CbNeed type system).
We can now derive an auxiliary type derivation Z as follows

$$\frac{\dfrac{\Omega \rhd_O \Pi_2; x : N \vdash^{(m''_1,e''_1,r''_1)} \mathcal{J}\langle u\rangle : [\mathsf{tight}]}{\Pi_2; x : N \vdash^{(m''_1,e''_1,r''_1)} (\mathcal{J}\langle u\rangle, \epsilon) : [\mathsf{tight}]}\ \mathsf{Lift} \qquad \Theta \rhd_O \Delta \vdash^{(m_2,e_2,r_2)} s : N}{(\Pi_2 \uplus \Delta); x : N \vdash^{(m''_1+m_2,e''_1+e_2,r''_1+r_2)} (\mathcal{J}\langle u\rangle, [x\leftarrow s]) : [\mathsf{tight}]}\ \mathrm{ES}$$

on which we can apply the *i.h.* to obtain a type derivation

$$Z' \rhd_O (\Pi_2 \biguplus \Delta); x : N \vdash^{(m''_1+m_2+1,e''_1+e_2,r''_1+r_2)} \mathcal{J}\langle(\lambda x.u)s\rangle : [\mathsf{tight}]$$

and finally derive Φ' as follows

$$\frac{\Xi \rhd_O \Pi_1 \vdash^{(m'_1,e'_1,r'_1)} i : [\mathsf{inert}]_{i\in I} \qquad Z'}{(\Pi_1 \uplus \Pi_2 \uplus \Delta); x : N \vdash^{(m'_1+m''_1+m_2+1,e'_1+e''_1+e_2,r'_1+r''_1+r_2+1)} i\mathcal{J}\langle u\rangle : [\mathsf{inert}]_{i\in I}}\ \mathsf{app}_i$$

verifying that
  * $\Pi_1 \uplus \Pi_2 \uplus \Delta = \Pi \uplus \Delta = \Gamma$, and
  * $(m'_1 + m''_1 + m_2 + 1, e'_1 + e''_1 + e_2, r'_1 + r''_1 + r_2 + 1) = (m_1 + m_2 + 1, e_1 + e_2, r_1 + r_2) = (m + 1, e, r)$.

– The case where Φ ends in the application of an $\mathrm{ES}_{\mathrm{gc}}$ rule is proven very similarly to the previous case, where ES is replaced with $\mathrm{ES}_{\mathrm{gc}}$ both in the analysis of the shape of Φ and of Z, and changes are made accordingly.

□

(Click here to go back to main chapter.)

Proposition 13.4.15 (Quantitative Subject Expansion for Open CbNeed)**.**
Let $\Phi' \rhd_O \Gamma \vdash^{(m,e,r)} p' : M$ be a tight type derivation.
1. Multiplicative*: If $p \to_{\mathsf{om}} p'$, then there exists a type derivation $\Phi \rhd_O \Gamma \vdash^{(m+1,e,r)} p : M$.*
2. Exponential*: If $p \to_{\mathsf{oe}} p'$, then there exists a type derivation $\Phi \rhd_O \Gamma \vdash^{(m,e+1,r)} p : M$.*

Proof. *(Click here to go back to main chapter.)*
1. *Multiplicative case*: We prove this by means of a weaker statement:

Let $p = P\langle(\lambda x.t)u\rangle \to_{\mathsf{om}} P\langle t, [x{\leftarrow}u]\rangle = p'$, with $P \in \mathcal{E}_{\mathcal{V}}$, and let

$$\Phi \rhd_O \Gamma \vdash^{(m,e,r)} p' : M$$

be a type derivation such that $\mathsf{inert}_{\Gamma}(\mathcal{V})$. Then there exists

$$\Phi' \rhd_O \Gamma \vdash^{(m+1,e,r)} p : M$$

We proceed by induction on the derivation of P:

- Let $P = (\mathcal{H}, \epsilon)$. Note that then $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} (\mathcal{H}\langle t\rangle, [x{\leftarrow}u]) : M$. Note that $\mathsf{inert}_{\Gamma}(\mathsf{nv}(\mathcal{H}))$, and so the statement follows by application of Lemma 7.1.11 (Quantitative Subject Expansion for $\to_{\mathsf{om}}$ in term contexts) on Φ.
- Let P be derived as
$$\frac{Q \in \mathcal{E}_{\mathcal{V}} \quad y \in \mathcal{V}}{Q@[y{\leftarrow}i] \in \mathcal{E}_{(\mathcal{V}\setminus\{x\})\cup\mathsf{nv}(i)}}\ \mathsf{O_I}$$
where $P = Q@[y{\leftarrow}i]$ and $\mathcal{V} = (\mathcal{V} \setminus \{x\}) \cup \mathsf{nv}(i)$. We proceed by case analysis on the last typing rule in Φ:
  - Let Φ be of the form
$$\frac{\Psi \rhd_O \Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle t, [x{\leftarrow}u]\rangle : M \quad \Theta \rhd_O \Delta \vdash^{(m_2,e_2,r_2)} i : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle t, [x{\leftarrow}u]\rangle@[y{\leftarrow}i] : M}\ \mathrm{ES}$$
where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that $\mathsf{inert}_{\Gamma}(\mathcal{V})$ implies that $\mathsf{inert}_{\Pi}(\mathsf{nv}(i))$, and so we can apply the Lemma 7.1.2.2 (Typing properties of normal terms - Inert terms) on Θ to obtain that $N \in \mathsf{Inert}$. Thus, $\mathsf{inert}_{\Gamma}(\mathcal{V})$ implies that $\mathsf{inert}_{\Pi;y:N}(\mathcal{W})$, and so we can apply the *i.h.* on Ψ to obtain type derivation $\Psi' \rhd_O \Pi; y : N \vdash^{(m_1+1,e_1,r_1)} Q\langle(\lambda x.t)u\rangle : M$. The statement follows by taking Ψ' and Θ as premises for the ES rule, thus deriving Φ'.
  - Let Φ be of the form
$$\frac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} Q\langle t, [x{\leftarrow}u]\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle t, [x{\leftarrow}u]\rangle@[y{\leftarrow}i] : M}\ \mathrm{ES_{gc}}$$
Note that since $y \notin \mathsf{dom}(\Gamma)$, then $\mathsf{inert}_{\Gamma}(\mathcal{V})$ implies that $\mathsf{inert}_{\Gamma}(\mathcal{W})$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi' \rhd_O \Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle : M$. The statement follows by deriving Φ' as follows
$$\frac{\Psi' \rhd_O \Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle@[y{\leftarrow}i] : M}\ \mathrm{ES_{gc}}$$
- Let P be derived as
$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \notin \mathcal{W}}{Q@[y{\leftarrow}s] \in \mathcal{E}_{\mathcal{W}}}\ \mathsf{O_{GC}}$$
where $P = Q@[y{\leftarrow}s]$ and $\mathcal{V} = \mathcal{W}$. We proceed by case analysis on the last typing rule in Φ:
  - Let Φ be of the form
$$\frac{\Psi \rhd_O \Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle t, [x{\leftarrow}u]\rangle : M \quad \Theta \rhd_O \Delta \vdash^{(m_2,e_2,r_2)} s : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle t, [x{\leftarrow}u]\rangle@[y{\leftarrow}s] : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \uplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that since $y \notin \mathcal{W}$, then $\mathsf{inert}_\Gamma(\mathcal{V})$ implies that $\mathsf{inert}_{\Pi; y:N}(\mathcal{W})$. Hence, we can apply the *i.h.* on Ψ to obtain type derivation $\Psi \rhd_O \Pi; y : N \vdash^{(m_1+1, e_1, r_1)} Q\langle(\lambda x.t)u\rangle : M$. The statement follows by taking Ψ' and Θ as premises for the ES rule, thus deriving Φ'.

– Let Φ be of the form

$$\dfrac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} Q\langle t, [x{\leftarrow}u]\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle t, [x{\leftarrow}u]\rangle @ [y{\leftarrow}s] : M}\ \mathrm{ES_{gc}}$$

Note that $\mathsf{inert}_\Gamma(\mathcal{V})$ implies that $\mathsf{inert}_\Gamma(\mathcal{W})$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi \rhd_O \Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle : M$. The statement follows by deriving Φ' as follows:

$$\dfrac{\Psi \rhd_O \Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle @ [y{\leftarrow}s] : M}\ \mathrm{ES_{gc}}$$

- Let P be derived as

$$\dfrac{Q \in \mathcal{E}_\mathcal{W} \quad y \notin \mathcal{W}}{Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{W} \cup \mathsf{nv}(\mathcal{H})}}\ \mathsf{O_{HER}}$$

where $P = Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}]$ and $\mathcal{V} = \mathcal{W} \cup \mathsf{nv}(\mathcal{H})$. Note that then $p = Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}\langle(\lambda x.t)u\rangle] \to_{\mathsf{om}} (Q\langle y\rangle @ [x{\leftarrow}\mathcal{H}\langle t\rangle]) @ [x{\leftarrow}u] = p'$. Moreover, since x is bound in $\lambda x.t$, then note that $x \notin \mathsf{fv}(Q\langle y\rangle)$.

We proceed by case analysis on the shape of Φ:

– Let Φ be of the form

$$\dfrac{\dfrac{\vdots\,\Psi \qquad \vdots\,\Theta}{\Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : M \qquad \Delta; x : O \vdash^{(m_2,e_2,r_2)} \mathcal{H}\langle t\rangle : N}\ \mathrm{ES} \qquad \vdots\,\Xi}{(\Pi \uplus \Delta); x : O \vdash^{(m_1+m_2, e_1+e_2, r_1+r_2)} Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}\langle t\rangle] : M \qquad \Sigma \vdash^{(m_3,e_3,r_3)} u : O}\ \mathrm{ES}$$
$$\Pi \uplus \Delta \uplus \Sigma \vdash^{(m_1+m_2+m_3, e_1+e_2+e_3, r_1+r_2+r_3)} (Q\langle y\rangle @ [x{\leftarrow}\mathcal{H}\langle t\rangle]) @ [x{\leftarrow}u] : M$$

where $\Gamma = \Pi \uplus \Delta \uplus \Sigma$ and $(m, e, r) = (m_1 + m_2 + m_3, e_1 + e_2 + e_3, r_1 + r_2 + r_3)$. Note that since $x \notin \mathsf{fv}(Q\langle y\rangle)$, then $x \notin \mathsf{dom}(\Pi)$—by Lemma 7.1.1 (Relevance of the Open CbNeed type system).

Let us now derive an auxiliary type derivation Z as follows

$$\dfrac{\dfrac{\Theta \rhd_O \Delta; x : O \vdash^{(m_2,e_2,r_2)} \mathcal{H}\langle t\rangle : N}{\Delta; x : O \vdash^{(m_2,e_2,r_2)} (\mathcal{H}\langle t\rangle, \epsilon) : N}\ \mathsf{Lift} \qquad \Xi \rhd_O \Sigma \vdash^{(m_3,e_3,r_3)} u : O}{\Delta \uplus \Sigma \vdash^{(m_2+m_3, e_2+e_3, r_2+r_3)} (\mathcal{H}\langle t\rangle, [x{\leftarrow}u]) : N}\ \mathrm{ES}$$

We can apply Lemma 7.1.11 (Quantitative Subject Expansion for $\to_{\mathsf{om}}$ in term contexts) on Z to obtain type derivation $Z' \rhd_O \Delta \uplus \Sigma \vdash^{(m_2+m_3+1, e_2+e_3, r_2+r_3)} \mathcal{H}\langle(\lambda x.t)u\rangle : N$. Therefore, the statement follows by deriving Φ' as follows

$$\dfrac{\vdots\,\Psi \qquad \vdots\,Z'}{\Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : M \qquad \Delta \uplus \Sigma \vdash^{(m_2+m_3+1, e_2+e_3, r_2+r_3)} \mathcal{H}\langle(\lambda x.t)u\rangle : N}\ \mathrm{ES}$$
$$\Pi \uplus \Delta \uplus \Sigma \vdash^{(m_1+m_2+m_3+1, e_1+e_2+e_3, r_1+r_2+r_3)} Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}\langle(\lambda x.t)u\rangle] : M$$

noting that $Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}\langle(\lambda x.t)u\rangle] = P\langle(\lambda x.t)u\rangle$.

– We can proceed analogously to the previous case to prove the statement for the case where Φ is of the following form

$$\dfrac{\dfrac{\vdots\,\Psi \qquad \vdots\,\Theta}{\Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : M \qquad \Delta \vdash^{(m_2,e_2,r_2)} \mathcal{H}\langle t\rangle : N}\ \text{ES}}{\dfrac{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle t\rangle] : M \qquad (\Pi \biguplus \Delta)(x) = \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} (Q\langle y\rangle@[x{\leftarrow}\mathcal{H}\langle t\rangle])@[x{\leftarrow}u] : M}\ \text{ES}_{\text{gc}}}$$

– Suppose Φ were of the form

$$\dfrac{\dfrac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} Q\langle y\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle t\rangle] : M}\ \text{ES}_{\text{gc}} \qquad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} (Q\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle t\rangle])@[x{\leftarrow}u] : M}\ \text{ES}_{\text{gc}}$$

However, we would have that $y \notin \mathsf{fv}(Q\langle y\rangle)$—by Lemma 7.1.1 (Relevance of the Open CbNeed type system) on Ψ—which contradicts the fact that $y \in \mathsf{fv}(Q\langle y\rangle)$ given by Lemma 13.4.4 (Plugged variables and domain of type contexts). Hence, this case is impossible.

– Suppose Φ were of the form

$$\dfrac{\dfrac{\vdots\,\Psi}{\Pi; x : O \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : M \qquad \Pi(y) = \mathbf{0}}\ \text{ES}_{\text{gc}} \qquad \dfrac{\vdots\,\Xi}{\Sigma \vdash^{(m_3,e_3,r_3)} u : O \qquad O \neq \mathbf{0}}}{\dfrac{\Pi; x : O \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle t\rangle] : M}{\Pi \biguplus \Sigma \vdash^{(m,e,r)} (Q\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle t\rangle])@[x{\leftarrow}u] : M}\ \text{ES}}$$

But then $x \in \mathsf{fv}(Q\langle y\rangle)$—by Lemma 7.1.1 (Relevance of the Open CbNeed type system) on Ψ—which is absurd.

2. *Exponential case*: We prove this by means of a weaker statement:

Let $p = P\langle x\rangle \to_{\mathsf{oe}} P\langle v\rangle = p'$, with $P \in \mathcal{E}_{\mathcal{V}}$ and $P(x) = v \in \mathsf{Val}$, and let $\Phi \rhd_O \Gamma \vdash^{(m,e,r)} p' : M$ be a type derivation such that $\mathsf{inert}_\Gamma(\mathcal{V})$. Then there exists $\Phi' \rhd_O \Gamma \vdash^{(m,e+1,r)} p : M$

We proceed by induction on the derivation of P:

- The case where $P = (\mathcal{H}^{@}, \epsilon)$ is impossible, since it would be that $\mathsf{dom}(P) = \emptyset$.
- Let P be derived as

$$\dfrac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \in \mathcal{V}}{Q@[y{\leftarrow}i] \in \mathcal{E}_{(\mathcal{W}\setminus\{x\})\cup\mathsf{nv}(i)}}\ \mathsf{O_I}$$

where $P = Q@[y{\leftarrow}i]$ and $\mathcal{V} = (\mathcal{W} \setminus \{x\}) \cup \mathsf{nv}(i)$. Note that since $i \notin \mathsf{Val}$, then $x \neq y$ and then $x \in \mathsf{dom}(Q)$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\dfrac{\Psi \rhd_O \Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : M \quad \Theta \rhd_O \Delta \vdash^{(m_2,e_2,r_2)} i : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle v\rangle@[y{\leftarrow}i] : M}\ \text{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that $\mathsf{inert}_\Gamma(\mathcal{V})$ implies that $\mathsf{inert}_\Pi(\mathsf{nv}(i))$, and so we can apply the Lemma 7.1.2.2 (Typing properties of normal terms - Inert terms) on Θ to obtain that $N \in \mathsf{Inert}$. Thus, $\mathsf{inert}_\Gamma(\mathcal{V})$ implies

that $\mathsf{inert}_{\Pi;y:N}(\mathcal{W})$ and so we can apply the *i.h.* on Ψ to obtain type derivation $\Psi' \rhd_O \Pi; y:N \vdash^{(m_1,e_1+1,r_1)} Q\langle x\rangle : M$. The statement follows by taking Ψ' and Θ as premises for the ES rule, thus deriving Φ'.

– Let Φ be of the form

$$\frac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} Q\langle v\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle v\rangle @[y{\leftarrow}i] : M}\ \mathrm{ES_{gc}}$$

Note that since $y \notin \mathsf{dom}(\Gamma)$, then $\mathsf{inert}_\Gamma(\mathcal{V})$ implies that $\mathsf{inert}_\Gamma(\mathcal{W})$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi' \rhd_O \Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle : M$. The statement follows by deriving Φ' as follows

$$\frac{\Psi' \rhd_O \Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle @[y{\leftarrow}i] : M}\ \mathrm{ES_{gc}}$$

- Let P be derived as

$$\frac{Q \in \mathcal{E}_\mathcal{W} \quad y \notin \mathcal{W}}{Q@[y{\leftarrow}s] \in \mathcal{E}_\mathcal{W}}\ \mathsf{O_{GC}}$$

where $P = Q@[y{\leftarrow}s]$ and $\mathcal{V} = \mathcal{W}$.
We distinguish two cases, namely the one where $x = y$ and $s = v$—*i.e.*, when $p' = Q\langle v\rangle@[x{\leftarrow}v]$—and one where $x \neq y$. Let us first assume that $x \neq y$ and proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Psi \rhd_O \Pi; y:N \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : M \quad \Theta \rhd_O \Delta \vdash^{(m_2,e_2,r_2)} s : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle v\rangle @[y{\leftarrow}s] : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \uplus \Delta$ and $(m,e,r) = (m_1+m_2, e_1+e_2, r_1+r_2)$. Note that since $y \notin \mathcal{B}$, then $\mathsf{inert}_\Gamma(\mathcal{V})$ implies that $\mathsf{inert}_{\Pi;y:N}(\mathcal{W})$. Hence, we can apply the *i.h.* on Ψ to obtain type derivation $\Psi \rhd_O \Pi; y:N \vdash^{(m_1,e_1+1,r_1)} Q\langle x\rangle : M$. The statement follows by taking Ψ' and Θ as premises for the ES rule, thus deriving Φ'.

– Let Φ be of the form

$$\frac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} Q\langle v\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle v\rangle @[y{\leftarrow}s] : M}\ \mathrm{ES_{gc}}$$

Note that $\mathsf{inert}_\Gamma(\mathcal{V})$ implies that $\mathsf{inert}_\Gamma(\mathcal{W})$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi \rhd_O \Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle : M$. The statement follows by deriving Φ' as follows:

$$\frac{\Psi \rhd_O \Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle @[y{\leftarrow}s] : M}\ \mathrm{ES_{gc}}$$

This completes the case where $x \neq y$. Let us now assume that $x = y$ and $s = v$. That is,

$$p = Q\langle x\rangle@[x{\leftarrow}v] \to_{\mathsf{oe}} Q\langle v^\alpha\rangle@[x{\leftarrow}v] = p'$$

We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Psi \rhd_O \Pi; x : N \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : M \quad \Theta \rhd_O \Delta \vdash^{(m_2,e_2,r_2)} v : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle v\rangle @ [x \leftarrow v] : M}\ \text{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that $x \notin \mathcal{W}$ and $x \notin \mathsf{dom}(Q)$ because $x = y$, and that we can safely assume that $\mathsf{fv}(v) \cap \mathsf{dom}(Q) = \emptyset$. Hence, we can apply Lemma 7.1.10.2 (Linear Removal for Open CbNeed - exponential case) on Ψ to obtain multi type O and type derivations

$$\begin{array}{l} \Xi \rhd_O \Pi' \vdash^{(m'_1,e'_1,r'_1)} v : O \\ \Omega \rhd_O \Pi''; x : (N \uplus O) \vdash^{(m''_1,e''_1,r''_1)} Q\langle x\rangle : M \end{array}$$

such that $\Pi = \Pi' \biguplus \Pi''$ and $(m'_1 + m''_1, e'_1 + e''_1, r'_1 + r''_1) = (m_1, e_1 + 1, r_1)$. Next, we apply Lemma 13.4.13 (Merging multi types of Open CbNeed type derivations) on Ξ and Θ to obtain

$$Z \rhd_O \Pi' \biguplus \Delta \vdash^{(m'_1+m_2,e'_1+e_2,r'_1+r_2)} v : N \uplus O$$

Finally, the statement follows by deriving Φ' as follows:

$$\frac{\Omega \qquad Z \qquad (N \uplus O) \neq \mathbf{0}}{\Pi'' \biguplus \Pi' \biguplus \Delta \vdash^{(m''_1+m'_1+m2,e''_1+e'_1+e_2,r''_1+r'_1+r_2)} Q\langle x\rangle @ [x \leftarrow v] : M}\ \text{ES}$$

where $\Pi'' \biguplus \Pi' \biguplus \Delta = \Pi \biguplus \Delta = \Gamma$ and

$$\begin{array}{ll} & (m''_1 + m'_1 + m2, e''_1 + e'_1 + e_2, r''_1 + r'_1 + r_2) \\ = & (m_1 + m_2, r_1 + r2, (e_1 + 1) + e_2) \\ = & (m, r, e + 1) \end{array}$$

– Let Φ be of the form

$$\frac{\Psi \rhd_O \Gamma \vdash^{(m,e,r)} Q\langle v\rangle : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle v\rangle @ [x \leftarrow v] : M}\ \text{ES}_{\text{gc}}$$

By application of Lemma 7.1.10.2 (Linear Removal for Open CbNeed - exponential case) on Ψ, there exist multi type $O \neq \mathbf{0}$ and type derivations

$$\begin{array}{l} \Theta \rhd_O \Pi; x : O \vdash^{(m',e',r')} Q\langle x\rangle : M \\ \Xi \rhd_O \Delta \vdash^{(m'',e'',r'')} v : O \end{array}$$

such that $\Gamma = \Pi \biguplus \Delta$ and $(m, e + 1, r) = (m' + m'', e' + e'', r' + r'')$.
Therefore, we can derive Φ' as follows:

$$\frac{\Theta \rhd_O \Pi; x : O \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Xi \rhd_O \Delta \vdash^{(m'',e'',r'')} v : O \quad O \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle @ [x \leftarrow v] : M}\ \text{ES}_{\text{gc}}$$

- Suppose P is derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{W}} \quad y \notin \mathcal{W}}{Q\langle y\rangle @ [y \leftarrow \mathcal{H}] \in \mathcal{E}_{\mathcal{W} \cup \mathsf{nv}(\mathcal{H})}}\ \mathsf{O}_{\mathsf{HER}}$$

where $P = Q\langle y\rangle @ [y \leftarrow \mathcal{H}]$ and $\mathcal{V} = \mathcal{W} \cup \mathsf{nv}(\mathcal{H})$. But then $x \notin \mathsf{dom}(P)$, which is absurd.

□

(Click here to go back to main chapter.)

Theorem 13.4.16 (Tight Completeness for Open CbNeed)**.**
Let $p \in \mathcal{PR}$. If there exists $d : p \to^*_{\mathsf{ond}} q$ for some $q \in \mathcal{PR}$ in $\to_{\mathsf{ond}}$-normal form, then there exists a tight type derivation $\Phi \rhd_O \Gamma \vdash^{(|d|_{\mathsf{m}},|d|_{\mathsf{e}},|q|_{\mathsf{nd}})} p : M$ such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$.

Proof. *(Click here to go back to main chapter.)*
By induction on the length $|d|$ of the reduction sequence $d : p \to^*_{\mathsf{ond}} q$:

- *Base case*: Let $k \coloneqq 0$. Then p is in $\to_{\mathsf{ond}}$-normal form and $p = q$. The statement follows by application of Proposition 7.1.9 (Tight typability of Open CbNeed-normal forms), which yields a tight type derivation $\Phi \rhd_O \Gamma \vdash^{(0,0,|p|_{\mathsf{nd}})} t : M$ such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.
- *Inductive case*: Let $k > 0$. That is, d is of the following form

$$d : p \to_{\mathsf{ond}} \underbrace{r \to^{k-1}_{\mathsf{ond}} q}_{d'}$$

  By *i.h.*, there exists tight type derivation

$$\Psi \rhd_O \Gamma \vdash^{(|d'|_{\mathsf{m}},|d'|_{\mathsf{e}},|q|_{\mathsf{nd}})} r : M$$

  such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$. We proceed by case analyis on the kind of reduction step in $p \to_{\mathsf{ond}} r$:
  - *Multiplicative step*: Let $p \to_{\mathsf{om}} r$. By Proposition 7.1.12.1 (Quantitative Subject Expansion for Open CbNeed - Multiplicative), there exists (tight) type derivation

$$\Psi' \rhd_O \Gamma \vdash^{(|d'|_{\mathsf{m}}+1,|d'|_{\mathsf{e}},|q|_{\mathsf{nd}})} p : M$$

    Therefore, the statement follows by taking $\Phi \coloneqq \Psi'$, noting in particular that

$$(|d'|_{\mathsf{m}} + 1, |d'|_{\mathsf{e}}, |q|_{\mathsf{nd}}) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})$$

  - *Exponential step*: Let $p \to_{\mathsf{oe}} r$. By Proposition 7.1.12.2 (Quantitative Subject Expansion for Open CbNeed - Exponential), there exists (tight) type derivation

$$\Psi' \rhd_O \Gamma \vdash^{(|d'|_{\mathsf{m}},|d'|_{\mathsf{e}}+1,|q|_{\mathsf{nd}})} p : M$$

    Therefore, the statement follows by taking $\Phi \coloneqq \Psi'$, noting in particular that

$$(|d'|_{\mathsf{m}}, |d'|_{\mathsf{e}} + 1, |q|_{\mathsf{nd}}) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})$$

□

(Click here to go back to main chapter.)

13.5 Proofs of Chapter 8 (Useful Open CbNeed)

13.5.1 The usefulness criterion

For proving Proposition 8.1.3 (Usefulness of exponential steps), we first need to characterize the shape of exponential in a very practical way, in the specific terms required for proving Proposition 8.1.3. That is, we need the following

Lemma 13.5.1 (Characterization of exponential evaluation contexts)**.**
Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be an exponential evaluation context. Then P has one of the following three forms:

1. *$P = (\mathcal{H}, E)$, where*
   (a) *$\mathcal{H} = \mathcal{J}\langle\langle\cdot\rangle\, u\rangle$ is an applicative term context, for some term context $\mathcal{J}$ and $u \in \Lambda$.*
   (b) *$Q = (\mathcal{J}, E) \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$.*
   (c) *The derivation of $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ has no applications of rule $\mathsf{E_{NL}}$.*
2. *$P = (t, E\,[z{\leftarrow}\mathcal{H}]\,E')$, where*
   (a) *$\mathcal{H} = \mathcal{J}\langle\langle\cdot\rangle\, u\rangle$ is an applicative term context, for some term context $\mathcal{J}$ and $u \in \Lambda$.*
   (b) *$Q = (t, E\,[z{\leftarrow}\mathcal{J}]\,E') \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$.*
   (c) *The derivation of $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ has no applications of rule $\mathsf{E_{NL}}$.*
3. *$P = (t, E\,[z{\leftarrow}\langle\cdot\rangle]\,E')$, where*
   (a) *There exists exponential evaluation context $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $(t, E) = Q\langle z\rangle$ and such that for every $v \in \mathsf{Val}$, $R := Q@[z{\leftarrow}v]\,E' \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$.*
   (b) *The number of applications of rule $\mathsf{E_{NL}}$ in the derivation of $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ is the number of applications of rule $\mathsf{E_{NL}}$ in the derivation of $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ plus 1. The same relation holds between the derivations of $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ and of $R \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$.*

Proof.
By induction on $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. Cases:

- Base rule $\mathsf{E_{AX_1}}$: then we are in the first case.
- Base rule $\mathsf{E_{AX_2}}$: then we are in the second case.
- Inductive rules $\mathsf{E_I}$, $\mathsf{E_{VAR}}$, $\mathsf{E_{GC}}$, and $\mathsf{E_U}$: they simply preserve the case given by the *i.h.*.
- Inductive rule $\mathsf{E_{NL}}$: then $P = P_1\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle]$ and it is derived as follows:

$$\frac{P_1 \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad x \notin (\mathcal{U} \cup \mathcal{A})}{P_1\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E_{NL}}$$

By *i.h.*, P_1 has one of the 3 shapes in the statement. If it is the first or the second then we end up in the third case (details similar to the next case, that is the more interesting one). If it is the third case, then $P_1 = (t, E_1[z{\leftarrow}\langle\cdot\rangle]E_1')$ and $(t, E_1) = Q_1\langle z\rangle$ for some $Q_1 \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $R_1 := Q_1@([z{\leftarrow}v]E_1') \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ for every value v. Now note that by taking $Q := P_1$, $E := E_1[z{\leftarrow}x]E_1'$, and $E' := \epsilon$ we do satisfy the statement because we have

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad x \notin (\mathcal{U} \cup \mathcal{A})}{Q@[x{\leftarrow}v] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E_{GC}}$$

as required, and that the number of $\mathsf{E_{NL}}$ in the derivation of $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ is 1 plus the number in the derivation of $Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$.

□

Proposition 13.5.2 (Usefulness of exponential steps)**.**

Let $p = P\langle x\rangle \to_{\mathsf{ue}} P\langle v^\alpha\rangle = q$ with $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ and $P(x) = v$. Then there exists $r \in \mathcal{PR}$ and reduction sequence $d : q \to^{k}_{\mathsf{ue}}\to_{\mathsf{um}} r$ such that

1. *the evaluation context of each $\to_{\mathsf{ue}}$ steps in d is in $\mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$, and the one of $\to_{\mathsf{um}}$ is in $\mathcal{E}_{\mathcal{U},\mathcal{A}}$.*
2. *$k \geq 0$ is the number of $\mathsf{E_{NL}}$ rules in the derivation of $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$.*

Proof. (Click here to go back to main chapter.)

Let $p = P\langle x\rangle \to_{\mathsf{ue}} P\langle v^\alpha\rangle = q$ with $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ and $P(x) = v$. The proof is by induction on the derivation of $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. Cases:

- Rules $\mathsf{E_{AX_1}}$,$\mathsf{E_{AX_2}}$, and $\mathsf{E_{NL}}$: impossible, because if $P = (\mathcal{H}, \epsilon)$, $P = Q\langle x\rangle@[x\leftarrow\mathcal{H}]$, or $P = Q\langle x\rangle@[x\leftarrow\langle\cdot\rangle]$ then $P(x) = \bot$, against hypothesis.
- Rule $\mathsf{E_I}$: then $P = Q@[y\leftarrow t]$ and it is derived as follows:

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B}) \quad \mathsf{uinert}(t)}{Q@[y\leftarrow t] \in \mathcal{E}^{@}_{(\mathcal{V}\setminus\{y\})\cup\mathsf{u}(t),(\mathcal{B}\setminus\{y\})\cup\mathsf{a}(t)}}\ \mathsf{E_I}$$

  with $\mathcal{U} = (\mathcal{V}\setminus\{y\}) \cup \mathsf{u}(t)$ and $\mathcal{A} = (\mathcal{B}\setminus\{y\}) \cup \mathsf{a}(t)$. Note that $x \neq y$ because $P(x)$ is not an inert term. Let $q' := Q\langle x\rangle$. By *i.h.*, there exists $d' : q' \to^{k}_{\mathsf{ue}}\to_{\mathsf{um}} r'$ as in the statement. Let the evaluation context of each $\to_{\mathsf{ue}}$-step in d' be $Q_{\mathsf{ue},1}, ..., Q_{\mathsf{ue},k} \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$, and let the $\to_{\mathsf{um}}$-step in d' be $Q_{\mathsf{um}} \in \mathcal{E}_{\mathcal{V},\mathcal{B}}$. We can derive the evaluation contexts of each $\to_{\mathsf{ue}}$-step in d, say $P_{\mathsf{ue},1}, ..., P_{\mathsf{ue},1} \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$, and the evaluation context of the $\to_{\mathsf{um}}$ in d, P_{um} by applying the $\mathsf{E_I}$ rule obtaining a derivation $d' : q = q'@[x\leftarrow t] \to^{k}_{\mathsf{ue}}\to_{\mathsf{um}} r'@[x\leftarrow t]$ as in the statement.
- Rule $\mathsf{E_{VAR}}$: exactly as the previous one.
- Rule $\mathsf{E_{GC}}$: then $P = Q@[y\leftarrow t]$ and it is derived as follows:

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{Q@[y\leftarrow t] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E_{GC}}$$

  If $x \neq y$ then it goes as in the previous case. Otherwise, $t = v$ and $q = Q\langle v^\alpha\rangle@[x\leftarrow v]$. By Lemma 13.5.1, Q has one of the following three forms:

  1. $Q = (\mathcal{H}, E)$, where $\mathcal{H}$ is applicative, that is $\mathcal{H} = \mathcal{J}\langle\langle\cdot\rangle u\rangle$ for some term u, and $R := (\mathcal{J}, E) \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$. Then $q = Q\langle v^\alpha\rangle = R\langle v^\alpha t\rangle$ is a multiplicative redex in a useful multiplicative context, and so $q \to_{\mathsf{um}} r$ for some r. As required by the statement, the lemma gives also that there are no $\mathsf{E_{NL}}$ rules in Q, and thus in P.
  2. $Q = (t, E[z\leftarrow\mathcal{H}]E')$, where $\mathcal{H}$ is applicative. It goes as in the previous case.
  3. $Q = (t, E[z\leftarrow\langle\cdot\rangle]E')$ and $(t, E) = R\langle z\rangle$ for some $R \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ having one less $\mathsf{E_{NL}}$ rule than Q and such that $C''_1 := Q@([z\leftarrow w]E') \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ for every value w. Now, consider taking w as v^α and note that

$$q' := Q\langle v^\alpha\rangle = R\langle z\rangle@([z\leftarrow v^\alpha]E') \to_{\mathsf{ue}} R\langle v^\alpha\rangle@([z\leftarrow v^\alpha]E') =: q''$$

     By *i.h.*, there exists $d' : q'' \to^{k'}_{\mathsf{ue}}\to_{\mathsf{um}} r'$ for some r', where k' is the number of $\mathsf{E_{NL}}$ in Q and the evaluation contexts satisfy the statement. Then we can apply rule $\mathsf{E_{GC}}$ to all these evaluation contexts, and to Q as well, obtaining the following derivation, that satisfies the statement.

$$d : q = q'@[x\leftarrow v] \quad \to_{\mathsf{ue}} \quad q''@[x\leftarrow v] \quad \to^{k'}_{\mathsf{ue}}\to_{\mathsf{um}} \quad r'@[x\leftarrow v]$$

- Rule $\mathsf{E_U}$: exactly as the previous one.

□

(Click here to go back to main chapter.)

13.5.2 Characterizing Useful Open CbNeed-normal forms.

Lemma 13.5.3 (Unapplied, applied, and needed variables)**.**

1. Terms*:* $\mathsf{nv}(t) = \mathsf{u}(t) \cup \mathsf{a}(t)$ *for every* $t \in \Lambda$*.*
2. Programs*:* $\mathsf{nv}(p) = \mathsf{u}(p) \cup \mathsf{a}(p)$ *for every* $p \in \mathcal{PR}$*.*

Proof. (Click here to go back to main chapter.)

1. *Terms*: By induction on the shape of t:
   - Let $t = x$. Then $\mathsf{nv}(t) = \{x\} = \{x\} \cup \emptyset = \mathsf{u}(t) \cup \mathsf{a}(t)$.
   - Let $t = \lambda x.u$. Then $\mathsf{nv}(t) = \emptyset = \mathsf{u}(t) \cup \mathsf{a}(t)$.
   - Let $t = sm$. We do case analysis on the shape of s:
     - Let $s = x$. Then $\mathsf{nv}(t) = \mathsf{nv}(s) \cup \mathsf{nv}(m) = \{x\} \cup \mathsf{nv}(m) =^{i.h.} \{x\} \cup (\mathsf{u}(m) \cup \mathsf{a}(m)) = \mathsf{u}(m) \cup (\{x\} \cup \mathsf{a}(m)) = \mathsf{u}(t) \cup \mathsf{a}(t)$.
     - Let $s \neq x$. Then $\mathsf{nv}(t) = \mathsf{nv}(s) \cup \mathsf{nv}(m) =^{i.h.} (\mathsf{u}(s) \cup \mathsf{a}(s)) \cup (\mathsf{u}(m) \cup \mathsf{a}(m)) = (\mathsf{u}(s) \cup \mathsf{u}(m)) \cup (\mathsf{a}(s) \cup \mathsf{a}(m)) = \mathsf{u}(t) \cup \mathsf{a}(t)$.
2. *Programs*: Let $p = (t, E)$. We proceed by induction on the lenght of E:
   - Let $E = \epsilon$. By Lemma 8.1.1.1 (Unapplied, applied and needed variabless), we have that $\mathsf{nv}(t) = \mathsf{u}(t) \cup \mathsf{a}(t)$, and so $\mathsf{nv}(t, \epsilon) = \mathsf{nv}(t) = \mathsf{u}(t) \cup \mathsf{a}(t) = \mathsf{u}(t, \epsilon) \cup \mathsf{a}(t, \epsilon)$.
   - Let $E = E'[x{\leftarrow}u]$. We proceed by case analysis on $x \in \mathsf{nv}(t, E')$:
     - Let $x \in \mathsf{nv}(t, E')$. Then, $\mathsf{nv}(t, E'[x{\leftarrow}u]) = (\mathsf{nv}(t, E') \setminus \{x\}) \cup \mathsf{nv}(u)$, $\mathsf{nv}(t, E') = \mathsf{u}(t, E') \cup \mathsf{a}(t, E')$—by *i.h.*—and $\mathsf{nv}(u) = \mathsf{u}(u) \cup \mathsf{a}(u)$—by Lemma 8.1.1.1 (Unapplied, applied and needed variables). That is,

       $$\mathsf{nv}(t, E'[x{\leftarrow}u]) = ((\mathsf{u}(t, E') \cup \mathsf{a}(t, E')) \setminus \{x\}) \cup (\mathsf{u}(u) \cup \mathsf{a}(u))$$

       We proceed by case analysis on the shape of u and on how $x \in \mathsf{nv}(t, E') = \mathsf{u}(t, E') \cup \mathsf{a}(t, E')$:
       - Let $u = y$ and $x \in (\mathsf{u}(t, E') \setminus \mathsf{a}(t, E'))$. Then $x \notin \mathsf{a}(t, E')$ and $\mathsf{a}(u) = \emptyset$, so that

         $$\begin{array}{rl} & ((\mathsf{u}(t, E') \cup \mathsf{a}(t, E')) \setminus \{x\}) \cup (\mathsf{u}(u) \cup \mathsf{a}(u)) \\ = & ((\mathsf{u}(t, E') \setminus \{x\}) \cup \mathsf{u}(u)) \cup \mathsf{a}(t, E') \\ = & \mathsf{u}(t, E'[x{\leftarrow}u]) \cup \mathsf{a}(t, E'[x{\leftarrow}u]) \end{array}$$

       - Let $u = y$ and $x \in (\mathsf{a}(t, E') \setminus \mathsf{u}(t, E'))$. Then $x \notin \mathsf{u}(t, E')$, $\mathsf{u}(u) = \{y\}$, and $\mathsf{a}(u) = \emptyset$, so that

         $$\begin{array}{rl} & ((\mathsf{u}(t, E') \cup \mathsf{a}(t, E')) \setminus \{x\}) \cup (\mathsf{u}(u) \cup \mathsf{a}(u)) \\ = & \mathsf{u}(t, E') \cup ((\mathsf{a}(t, E') \setminus \{x\}) \cup \{y\}) \\ = & \mathsf{u}(t, E'[x{\leftarrow}u]) \cup \mathsf{a}(t, E'[x{\leftarrow}u]) \end{array}$$

       - Let $u = y$, $x \in \mathsf{u}(t, E')$ and $x \in \mathsf{a}(t, E')$. Then,

         $$\begin{array}{rl} & ((\mathsf{u}(t, E') \cup \mathsf{a}(t, E')) \setminus \{x\}) \cup (\mathsf{u}(u) \cup \mathsf{a}(u)) \\ = & (\mathsf{u}(t, E') \setminus \{x\}) \cup (\mathsf{a}(t, E') \setminus \{x\}) \cup \{y\} \\ = & ((\mathsf{u}(t, E') \setminus \{x\}) \cup \{y\}) \cup ((\mathsf{a}(t, E') \setminus \{x\}) \cup \{y\}) \\ = & \mathsf{u}(t, E'[x{\leftarrow}u]) \cup \mathsf{a}(t, E'[x{\leftarrow}u]) \end{array}$$

       - Let $u \neq y$. Then by definition $\mathsf{u}(t, E'[x{\leftarrow}u]) = (\mathsf{u}(t, E') \setminus \{x\}) \cup \mathsf{u}(u)$ and $\mathsf{a}(t, E'[x{\leftarrow}u]) = (\mathsf{a}(t, E') \setminus \{x\}) \cup \mathsf{a}(u)$. Thus,

         $$\begin{array}{rl} & ((\mathsf{u}(t, E') \cup \mathsf{a}(t, E')) \setminus \{x\}) \cup (\mathsf{u}(u) \cup \mathsf{a}(u)) \\ = & ((\mathsf{u}(t, E') \setminus \{x\}) \cup \mathsf{u}(u)) \cup ((\mathsf{a}(t, E') \setminus \{x\}) \cup \mathsf{a}(u)) \\ = & \mathsf{u}(t, E'[x{\leftarrow}u]) \cup \mathsf{a}(t, E'[x{\leftarrow}u]) \end{array}$$

– Let $x \notin \mathsf{nv}(t, E')$. The *i.h.* gives $\mathsf{nv}(t, E') = \mathsf{u}(t, E') \cup \mathsf{a}(t, E')$ and by definition $\mathsf{a}(t, E'[x \leftarrow u]) = \mathsf{a}(t, E')$ and $\mathsf{u}(t, E'[x \leftarrow u]) = \mathsf{u}(t, E')$. Hence,

$$\begin{array}{rcl} \mathsf{nv}(t, E'[x \leftarrow u]) & = & \mathsf{nv}(t, E') \\ & =_{i.h.} & \mathsf{u}(t, E') \cup \mathsf{a}(t, E') \\ & = & \mathsf{u}(t, E'[x \leftarrow u]) \cup \mathsf{a}(t, E'[x \leftarrow u]) \end{array}$$

□

(Click here to go back to main chapter.)

Lemma 13.5.4 (Term contexts: Unapplied, applied and needed variables)**.**
Let $\mathcal{H}$ be a term context. Then $\mathsf{nv}(\mathcal{H}) = \mathsf{u}(\mathcal{H}) \cup \mathsf{a}(\mathcal{H})$.

Proof. *(Click here to go back to main chapter.)*
By structural induction on $\mathcal{H}$: the base case is trivial, while the inductive ones easily follow from the *i.h.* and Lemma 8.1.1.1 (Unapplied, applied and needed variables).

□

(Click here to go back to main chapter.)

Lemma 13.5.5 (Disjointness of generalized variables, useful abstraction programs and useful inert programs)**.**
For every $p \in \mathcal{PR}$, at most one *of the following holds:*
i) $\mathsf{genVar}_{\#}(p)$,
ii) $\mathsf{uabs}(p)$, *or*
iii) $\mathsf{uinert}(p)$.

Proof. *(Click here to go back to main chapter.)*
Let $p = (t, E)$. We proceed by induction on $|E|$:

- *Base case*: Let $E \coloneqq \epsilon$. The statement follows by the fact that the base derivation rules for $\mathsf{genVar}_{\_}()$, $\mathsf{uabs}(.)$ and $\mathsf{uinert}(.)$ can only be applied to pairwise syntactically distinct Λ-terms.
- *Inductive case*: Let $E \coloneqq E'[y \leftarrow u]$. By *i.h.*, (t, E') can satisfy *at most* one of the three predicates. The proof proceeds by assuming that $\mathsf{genVar}_t(E'[y \leftarrow u],)$ (resp. $\mathsf{uabs}(t, E'[y \leftarrow u])$; $\mathsf{uinert}(t)E'[y \leftarrow u]$), and then proving that the other predicates cannot hold for $(t, E'[y \leftarrow u])$. We avoid the repetitive work here, since every case follows from the *i.h.* and the fact that each derivation rule for each of the predicates takes as premise the specific syntactical category of u.

□

(Click here to go back to main chapter.)

Lemma 13.5.6 (Properties of useful inert terms)**.**
Let i^+ be a non-variable inert term. Then $|i^+|_{\mathsf{nd}} \geq 1$ and $\mathsf{a}(i^+) \neq \emptyset$.

Proof. *(Click here to go back to main chapter.)*
The fact that $\mathsf{a}(i^+) \neq \emptyset$ can be easily proven by induction on the structure of i^+, while $|i^+|_{\mathsf{nd}} \geq 1$ simply follows by the fact that i^+ is an application Λ-term.

□

(Click here to go back to main chapter.)

Lemma 13.5.7 (Properties of generalized variables)**.**
Let $\mathsf{genVar}_x(p)$*. Then* $|p|_{\mathsf{nd}} = 0$*,* $\mathsf{nv}(p) = \mathsf{u}(p) = \{x\}$ *and* $\mathsf{a}(p) = \emptyset$*.*

Proof. (Click here to go back to main chapter.)
We proceed by induction on the derivation of $\mathsf{genVar}_x(p)$:
- If $p = (x, \epsilon)$ then the statement clearly follows.
- Let $\mathsf{genVar}_x(p)$ be derived as follows

$$\frac{\mathsf{genVar}_y(q)}{\mathsf{genVar}_x(q@[y{\leftarrow}x])}\ \mathsf{GV}_{\mathsf{HER}}$$

  By *i.h.* on q, $\mathsf{nv}(q) = \mathsf{u}(q) = \{y\}$, $\mathsf{a}(q) = \emptyset$ and $|q|_{\mathsf{nd}} = 0$. The statement follows easily.
- Let $\mathsf{genVar}_x(p)$ be derived as follows

$$\frac{\mathsf{genVar}_x(q) \quad y \neq x}{\mathsf{genVar}_x(q@[y{\leftarrow}t])}\ \mathsf{GV}_{\mathsf{GC}}$$

  By *i.h.* on q, $\mathsf{nv}(q) = \mathsf{u}(q) = \{x\}$, $\mathsf{a}(q) = \emptyset$ and $|q|_{\mathsf{nd}} = 0$. The statement follows easily.

□

(Click here to go back to main chapter.)

Lemma 13.5.8 (Properties of useful abstraction programs)**.**
Let $\mathsf{uabs}(p)$*. Then* $|p|_{\mathsf{nd}} = 0$ *and* $\mathsf{nv}(p) = \mathsf{u}(p) = \mathsf{a}(p) = \emptyset$*.*

Proof. (Click here to go back to main chapter.)
By induction on the derivation of $\mathsf{uabs}(p)$:
- The statement holds trivially if $p = (v, \epsilon)$.
- Let $\mathsf{uabs}(p)$ be derived as

$$\frac{\mathsf{genVar}_x(q)}{\mathsf{uabs}(q@[x{\leftarrow}v])}\ \mathsf{A}_{\mathsf{GV}}$$

  with $p = q@[x{\leftarrow}v]$. Note that $\mathsf{nv}(q@[x{\leftarrow}v]) = \mathsf{nv}(v)$—by Lemma 8.2.4 (Properties of generalized variables)—and so $|q@[x{\leftarrow}v]|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |v|_{\mathsf{nd}}$. We do case analysis on the last typing rule in Φ:
  - Let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} p : M$ be of the form

$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m',e',r')} q : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} v : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} q@[x{\leftarrow}v] : M}\ \mathrm{ES}$$

    where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. By application of the *i.h.* on Ψ, it must be that Φ is of the form

$$\frac{\Psi \rhd_U x : M \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} v : M}{\Delta \vdash^{(m'',e'',|q|_{\mathsf{nd}}+r'')} q@[x{\leftarrow}v] : M}\ \mathrm{ES}$$

    Since $M \in \mathsf{Tight}$ by hypothesis, we can apply the *i.h.* on Θ to obtain that $\Theta \rhd_U \Delta \vdash^{(0,0,|t|_{\mathsf{nd}})} v : M$, with $\mathsf{dom}(\Delta) = \mathsf{nv}(v)$ and $\mathsf{tight}(\Delta)$. Thus, Φ satisfies the statement.

– Suppose Φ is of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} q : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} q@[x{\leftarrow}v] : M}\ \mathrm{ES_{gc}}$$

However, an application of the *i.h.* on Ψ followed by an application of Lemma 8.2.4 (Properties of generalized variables) would yield that $x \in \mathsf{dom}(\Gamma)$, which makes this case absurd.

- Let $\mathsf{uabs}(p)$ be derived as

$$\frac{\mathsf{uabs}(q)}{\mathsf{uabs}(q@[x{\leftarrow}t])}\ \mathsf{GV_{GC}}$$

with $p = q@[x{\leftarrow}t]$. Note that $\mathsf{nv}(q@[x{\leftarrow}t]) = \mathsf{nv}(q)$ and so $|q@[x{\leftarrow}t]|_{\mathsf{nd}} = |q|_{\mathsf{nd}}$. We proceed by induction on the last typing rule in Φ:

– Suppose Φ is of the form

$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m',e',r')} q : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} q@[x{\leftarrow}t] : M}\ \mathrm{ES}$$

However, an application of the *i.h.* on Ψ together with an application of Lemma 8.2.5 (Properties of useful abstraction programs) would yield that $y \notin \mathsf{dom}(\Pi; x : N) = \emptyset$. This would mean that $N = \mathbf{0}$, making this case absurd.

– Let Φ be derived as

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} q : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} q@[x{\leftarrow}t] : M}\ \mathrm{ES_{gc}}$$

The statement then holds by *i.h.* on Ψ.

□

(Click here to go back to main chapter.)

Lemma 13.5.9 (Properties of useful inert programs)**.**
Let $\mathsf{uinert}(p)$*. Then* $|p|_{\mathsf{nd}} \geq 1$ *and* $\mathsf{a}(p) \neq \emptyset$*.*

Proof. (Click here to go back to main chapter.)
We proceed by induction on the derivation of $\mathsf{uinert}(p)$:

- If $\mathsf{uinert}(p)$ is derived as

$$\frac{}{\mathsf{uinert}(i^+, \epsilon)}\ \mathsf{I_{Lift}}$$

with $p = (i^+, \epsilon)$, then the statement follows by application of Lemma 8.2.3 (Properties of useful inert terms) on i^+.

- Let $\mathsf{uinert}(p)$ be derived as

$$\frac{\mathsf{uinert}(q) \quad x \in \mathsf{nv}(q)}{\mathsf{uinert}(q@[x{\leftarrow}i])}\ \mathsf{I_I}$$

with $p = q@[x{\leftarrow}i]$. By *i.h.* on q, $\mathsf{a}(q) \neq \emptyset$ and $|q|_{\mathsf{nd}} \geq 1$. Note that then $|p|_{\mathsf{nd}} \geq |q|_{\mathsf{nd}} \geq 1$. Note that if $i \in \mathsf{Var}$, then the statement follows easily. Let i be a non-variable inert Λ-term. Recall that $x \in \mathsf{nv}(q)$, and so by application of Lemma 8.2.3 (Properties of useful inert terms), we can conclude that $\mathsf{a}(p) = \mathsf{a}(q) \cup \mathsf{a}(i) \supseteq \mathsf{a}(i) \neq \emptyset$, $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |i|_{\mathsf{nd}} \geq |i|_{\mathsf{nd}} \geq 1$.

- Let $\mathsf{uinert}(p)$ be derived as
$$\frac{\mathsf{genVar}_x(q)}{\mathsf{uinert}(q@[x{\leftarrow}i^+])}\ \mathsf{I_{GV}}$$
with $p = q@[x{\leftarrow}i^+]$. By Lemma 8.2.4 (Properties of generalized variables) on q, $\mathsf{nv}(q) = \mathsf{u}(q) = \{x\}$, $\mathsf{a}(q) = \emptyset$ and $|q|_{\mathsf{nd}} = 0$. By Lemma 8.2.3 (Properties of useful inert terms) on i^+, $\mathsf{a}(i^+) \neq \emptyset$, $|i^+|_{\mathsf{nd}} \geq 1$, and i^+ is a non-variable inert Λ-term.
Therefore, the statement holds for p by noting that $\mathsf{a}(p) = (\mathsf{a}(q) \setminus \{x\}) \cup \mathsf{a}(t) \geq \mathsf{a}(t) \neq \emptyset$, $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |t|_{\mathsf{nd}} \geq |t|_{\mathsf{nd}} \geq 1$.
- Let $\mathsf{uinert}(p)$ be derived as
$$\frac{\mathsf{uinert}(q) \quad x \in \mathsf{ul}\,(q)}{\mathsf{uinert}(p@[x{\leftarrow}v])}\ \mathsf{I_U}$$
with $p = q@[x{\leftarrow}v]$. Note that $x \in \mathsf{nv}(q)$—by Lemma 8.1.1.2 (Unapplied, applied and needed variables). By *i.h.* on q, $\mathsf{a}(q) \neq \emptyset$, $|q|_{\mathsf{nd}} \geq 1$.
Therefore, we have that $\mathsf{a}(p) = (\mathsf{a}(q) \setminus \{x\}) \cup \mathsf{a}(v) = \mathsf{a}(q) \neq \emptyset$, $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |v|_{\mathsf{nd}} \geq |q|_{\mathsf{nd}} \geq 1$.
- Let $\mathsf{uinert}(p)$ be derived as
$$\frac{\mathsf{uinert}(q) \quad x \notin \mathsf{nv}(q)}{\mathsf{uinert}(q@[x{\leftarrow}t])}\ \mathsf{I_{GC}}$$
with $p = q@[x{\leftarrow}t]$. By *i.h.* on q, $\mathsf{a}(q) \neq \emptyset$ and $|q|_{\mathsf{nd}} \geq 1$. Thus, $\mathsf{a}(p) = \mathsf{a}(q) \neq \emptyset$, $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}} \geq 1$.

□

(Click here to go back to main chapter.)

Lemma 13.5.10 (Rewriting: term contexts)**.**

1. Focusing inert terms on unapplied variables*: Let i be an inert term and let $x \in \mathsf{u}(i)$. Then there exists term context $\mathcal{H}_x$ such that $\mathcal{H}_x\langle x\rangle = i$, with $x \notin \mathsf{u}(\mathcal{H}_x) \subset \mathsf{u}(i)$ and $\mathsf{a}(\mathcal{H}_x) \subseteq \mathsf{a}(i)$.*
2. Focusing term contexts on unapplied varables*: Let $\mathcal{H}$ be a term context and $x \in \mathsf{u}(\mathcal{H})$. Then for every $t \in \Lambda$ there exists a term context $\mathcal{H}_t$ such that $\mathcal{H}_t\langle x\rangle = \mathcal{H}\langle t\rangle$, with $x \notin \mathsf{u}(\mathcal{H}_t) \subset \mathsf{u}(\mathcal{H})$ and $x \notin \mathsf{a}(\mathcal{H}_t) \subseteq \mathsf{a}(\mathcal{H})$.*

Proof.

1. *Focusing inert terms on unapplied variables*: By structural induction on i:
   - *Variable*: The statement holds by taking $\mathcal{H} \coloneqq \langle\cdot\rangle$.
   - *Application*: Let $i = jn$. If $x \in j$ then the *i.h.* gives a term context $\mathcal{J}$ such that $\mathcal{J}\langle x\rangle = i$, with $x \notin \mathsf{u}(\mathcal{J}) \subset \mathsf{u}(j)$ and $x \notin \mathsf{a}(\mathcal{J}) \subseteq \mathsf{a}(j)$. The statement thus holds by taking $\mathcal{H} \coloneqq \mathcal{J}n$.
     If instead $x \notin j$, then $x \in n$. This means that n is itself an inert Λ-term, allowing us to apply the *i.h.* to get a term context $\mathcal{J}$ such that $\mathcal{J}\langle x\rangle = n$, with $x \notin \mathsf{u}(\mathcal{J}) \subset \mathsf{u}(i)$ and $x \notin \mathsf{a}(\mathcal{J}) \subseteq \mathsf{a}(i)$. The statement thus holds by taking $\mathcal{H} \coloneqq j\mathcal{J}$.
2. *Focusing term contexts on unapplied variables*: Let $t \in \Lambda$. We proceed by structural induction on $\mathcal{H}$:
   - *Empty context*: This case is impossible.
   - *Application left*: Let $\mathcal{H} \coloneqq \mathcal{J}u$. Then $x \in \mathsf{u}(\mathcal{J})$ and so application of the *i.h.* gives a term context $\mathcal{J}_t$ such that $\mathcal{J}_t\langle x\rangle = \mathcal{J}\langle t\rangle$, with $x \notin \mathsf{u}(\mathcal{J}_t) \subset \mathsf{u}(\mathcal{J})$ and $x \notin \mathsf{a}(\mathcal{J}_t) \subseteq \mathsf{a}(\mathcal{J})$. The statement then holds by taking $\mathcal{H}_t \coloneqq \mathcal{J}_t u$.
   - *Application right*: Let $\mathcal{H} \coloneqq i\mathcal{J}$. Case analysis on whether $x \in \mathsf{nv}(i)$:

– Let $x \in \mathsf{nv}(i)$. Then Lemma 13.5.10.1 (Focusing inert terms on unapplied variables) gives a term context $\mathcal{I}_x$ such that $\mathcal{I}_x\langle x\rangle = i$, with $x \notin \mathsf{u}(\mathcal{I}_x) \subset \mathsf{u}(i)$ and $x \notin \mathsf{a}(\mathcal{I}_x) \subseteq \mathsf{a}(i)$. The statement thus holds by taking $\mathcal{H}_t \coloneqq \mathcal{I}_x\mathcal{J}\langle t\rangle$, noting that $\mathcal{H}_t\langle x\rangle = \mathcal{I}_x\langle x\rangle\mathcal{J}\langle t\rangle = i\mathcal{J}\langle t\rangle = \mathcal{H}\langle t\rangle$, $x \notin \mathsf{u}(\mathcal{I}_x) \subset \mathsf{u}(\mathcal{H}_t)$, and $x \notin \mathsf{a}(\mathcal{I}_x) \subseteq \mathsf{a}(\mathcal{H}_t)$.
– Let $x \notin \mathsf{nv}(i)$. Then it must be that $x \in \mathsf{nv}(\mathcal{J})$, and so we can apply the *i.h.* to get a term context $\mathcal{J}_t$ such that $\mathcal{J}_t\langle x\rangle = \mathcal{J}\langle t\rangle$, with $x \notin \mathsf{u}(\mathcal{J}_t) \subset \mathsf{u}(\mathcal{J})$ and $x \notin \mathsf{a}(\mathcal{J}_t) \subseteq \mathsf{a}(\mathcal{J})$. The statement thus holds by taking $\mathcal{H}_t \coloneqq i\mathcal{J}_t$, noting that $\mathcal{H}_t\langle x\rangle = i\mathcal{J}_t\langle x\rangle = i\mathcal{J}\langle t\rangle = \mathcal{H}\langle t\rangle$, $x \notin \mathsf{u}(\mathcal{H}_t) = \mathsf{u}(i) \cup \mathsf{u}(\mathcal{J}_t) \subset \mathsf{u}(i) \cup \mathsf{u}(\mathcal{J}) = \mathsf{u}(\mathcal{H})$, and $x \notin \mathsf{a}(\mathcal{H}_t) = \mathsf{a}(i) \cup \mathsf{a}(\mathcal{J}_t) \subseteq \mathsf{a}(i) \cup \mathsf{a}(\mathcal{J}) = \mathsf{a}(\mathcal{H})$.

□

Lemma 13.5.11 (Rewriting: applicative term contexts)**.**

1. Applicative term contexts give applied variables*: Let $\mathcal{H}^@$ be an applicative term context and $x \in \mathsf{Var}$. Then $x \in \mathsf{a}(\mathcal{H}^@\langle x\rangle)$.*
2. Focusing useful inert terms on applied variables*: Let i^+ be a useful inert term and $x \in \mathsf{a}(i^+)$. Then there exists an applicative term context $\mathcal{H}^@_x$ such that $\mathcal{H}^@_x\langle x\rangle = i^+$, with $\mathsf{u}(\mathcal{H}^@_x) \subseteq \mathsf{u}(i^+)$ and $x \notin \mathsf{a}(\mathcal{H}^@_x) \subset \mathsf{a}(i^+)$.*
3. Focusing term contexts on applied variables*: Let $x \in \mathsf{a}(\mathcal{H})$. Then for every $t \in \Lambda$ there exists an applicative term context $\mathcal{H}^@_t$ such that $\mathcal{H}^@_t\langle x\rangle = \mathcal{H}\langle t\rangle$, with $\mathsf{u}(\mathcal{H}^@_t) \subseteq \mathsf{u}(\mathcal{H})$ and $x \notin \mathsf{a}(\mathcal{H}^@_t) \subset \mathsf{a}(\mathcal{H})$.*

Proof.

1. *Applicative term contexts give applied variables*: By structural induction on $\mathcal{H}^@$:
   - *Base case*: If $\mathcal{H}^@ = \langle\cdot\rangle t$ then $x \in (\{x\} \cup \mathsf{a}(t)) = \mathsf{a}(xt) = \mathsf{a}(\mathcal{H}^@\langle x\rangle)$.
   - *Application left*: Let $\mathcal{H}^@ \coloneqq \mathcal{J}^@t$. By *i.h.*, $x \in \mathsf{a}(\mathcal{J}^@\langle x\rangle)$ and so $x \in \mathsf{a}(\mathcal{J}^@\langle x\rangle) \cup \mathsf{a}(t) = \mathsf{a}(\mathcal{H}\langle x\rangle)$.
   - *Application right*: Let $\mathcal{H}^@ \coloneqq i\mathcal{J}^@$. By *i.h.*, $x \in \mathsf{a}(\mathcal{J}^@\langle x\rangle)$ and so $x \in \mathsf{a}(i) \cup \mathsf{a}(\mathcal{J}^@\langle x\rangle) = \mathsf{a}(\mathcal{H}\langle x\rangle)$.
2. *Focusing useful inert terms on applied variables*: By structural induction on $i^+ \coloneqq jn$. Note that if $j = x$ then the statement holds by taking $\mathcal{H}^@ \coloneqq \langle\cdot\rangle n$. Let j be a useful inert Λ-term. We proceed by case analysis on whether $x \in \mathsf{a}(i^+)$:
   - If $x \in \mathsf{a}(j)$, then j is a useful inert term. By application of the *i.h.*, we obtain an applicative term context $\mathcal{J}^@_x$ such that $j = \mathcal{J}^@_x\langle x\rangle$, with $\mathsf{u}(\mathcal{J}^@_x) \subseteq \mathsf{u}(j)$ and $x \notin \mathsf{a}(\mathcal{J}^@_x) \subset \mathsf{a}(j)$; thus, the statement holds by taking $\mathcal{H}^@ \coloneqq \mathcal{J}^@n$.
   - Let $x \notin \mathsf{a}(j)$. This means that $x \in \mathsf{a}(n)$, which in turn means that n is a useful inert term. By *i.h.*, there exists an applicative term context $\mathcal{J}^@$ such that $n = \mathcal{J}^@_x\langle x\rangle$, with $\mathsf{u}(\mathcal{J}^@_x) \subseteq \mathsf{u}(n)$ and $x \notin \mathsf{a}(\mathcal{J}^@_x) \subseteq \mathsf{a}(n)$; thus, the statement holds by taking $\mathcal{H}^@_x \coloneqq j\mathcal{J}^@_x$, since $\mathcal{H}^@_x\langle x\rangle = j\mathcal{J}^@_x\langle x\rangle = jn = i^+$, $\mathsf{u}(\mathcal{H}^@_x) = \mathsf{u}(j) \cup \mathsf{u}(\mathcal{J}^@_x) \subseteq \mathsf{u}(j) \cup \mathsf{u}(n) = \mathsf{u}(i^+)$, $x \notin \mathsf{a}(j) \cup \mathsf{a}(\mathcal{J}^@_x) = \mathsf{a}(\mathcal{H}^@_x) \subset \mathsf{a}(j) \cup \mathsf{a}(n) = \mathsf{a}(i^+)$.
3. *Focusing term contexts on applied variables*: Let $t \in \Lambda$. We proceed by structural induction on $\mathcal{H}$:
   - *Context hole*: Impossible.
   - *Application left*: Let $\mathcal{H} \coloneqq \mathcal{J}u$. Since then $x \in \mathsf{a}(\mathcal{H}) = \mathsf{a}(\mathcal{J})$, we can apply the *i.h.* and get an applicative term context $\mathcal{J}^@_t$ such that $\mathcal{J}^@_t\langle x\rangle = \mathcal{J}\langle t\rangle$, with $\mathsf{u}(\mathcal{J}^@_t) \subseteq \mathsf{u}(\mathcal{J})$ and $x \notin \mathsf{a}(\mathcal{J}^@_t) \subset \mathsf{a}(\mathcal{J})$. The statement holds by taking $\mathcal{H}^@ \coloneqq \mathcal{J}^@u$, and noting that $\mathcal{H}^@_t\langle x\rangle = \mathcal{J}^@_t\langle x\rangle t = \mathcal{J}\langle t\rangle u = \mathcal{H}\langle t\rangle$, $\mathsf{u}(\mathcal{H}^@_t) = \mathsf{u}(\mathcal{J}^@_t) \subseteq \mathsf{u}(\mathcal{J}) = \mathsf{u}(\mathcal{H})$, and $x \notin \mathsf{a}(\mathcal{J}^@_t) = \mathsf{a}(\mathcal{H}^@_t) \subset \mathsf{a}(\mathcal{J}) = \mathsf{a}(\mathcal{H})$.

- *Application right*: Let $\mathcal{H} \coloneqq i\mathcal{J}$. We proceed by case analysis on whether $x \in \mathsf{a}(i)$:
  - If $x \in \mathsf{a}(i)$, then i is a useful inert term. By application of Lemma 13.5.11.2 (Focusing useful inert terms on applied variables) on i, we get an applicative term context $\mathcal{J}_x^{@}$ such that $\mathcal{J}_x^{@}\langle x\rangle = i$, with $\mathsf{u}(\mathcal{J}_x^{@}) \subseteq \mathsf{u}(i)$ and $x \notin \mathsf{a}(\mathcal{J}_x^{@}) \subset \mathsf{a}(i)$. The statement holds by taking $\mathcal{H}_t^{@} \coloneqq \mathcal{J}_x^{@}\mathcal{J}\langle t\rangle$, since then $\mathcal{H}_t^{@}\langle x\rangle = (\mathcal{J}^{@}\langle x\rangle)\mathcal{J}\langle t\rangle = i\mathcal{J}\langle t\rangle = \mathcal{H}\langle t\rangle$, $\mathsf{u}(\mathcal{H}_t^{@}) = \mathsf{u}(\mathcal{J}_x^{@}) \subseteq \mathsf{u}(i) \subseteq \mathsf{u}(i) \cup \mathsf{u}(\mathcal{J}) = \mathsf{u}(\mathcal{H})$ and $x \notin \mathsf{a}(\mathcal{J}_x^{@}) = \mathsf{a}(\mathcal{H}_t^{@}) \subseteq \mathsf{a}(i) \subset \mathsf{a}(i) \cup \mathsf{a}(\mathcal{J}) = \mathsf{a}(\mathcal{H})$.
  - Let $x \notin \mathsf{a}(i)$. Then it must be that $x \in \mathsf{a}(\mathcal{J})$ and so the *i.h.* gives an applicative term context $\mathcal{J}_t^{@}$ such that $\mathcal{J}_t^{@}\langle x\rangle = \mathcal{J}\langle t\rangle$, with $\mathsf{u}(\mathcal{J}_t^{@}) \subseteq \mathsf{u}(\mathcal{J})$ and $x \notin \mathsf{a}(\mathcal{J}_t^{@}) \subset \mathsf{a}(\mathcal{J})$. The statement holds by taking $\mathcal{H}_t^{@} \coloneqq i\mathcal{J}_x^{@}$, since then $\mathcal{H}_t^{@}\langle x\rangle = i\mathcal{J}_x^{@}\langle x\rangle = i\mathcal{J}\langle t\rangle = \mathcal{H}\langle t\rangle$, $\mathsf{u}(\mathcal{H}_t^{@}) = \mathsf{u}(i) \cup \mathsf{u}(\mathcal{J}_x^{@}) \subseteq \mathsf{u}(i) \cup \mathsf{u}(\mathcal{J}) = \mathsf{u}(\mathcal{H})$ and $x \notin \mathsf{a}(i) \cup \mathsf{a}(\mathcal{J}_x^{@}) = \mathsf{a}(\mathcal{H}_t^{@}) \subset \mathsf{a}(i) \cup \mathsf{a}(\mathcal{J}) = \mathsf{a}(\mathcal{H})$.

□

Lemma 13.5.12 (Rewriting evaluation contexts: base cases)**.**

1. Multiplicative evaluation contexts give needed variables*: Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ and $x \notin \mathsf{dom}(P)$. Then $x \in \mathsf{nv}(P\langle x\rangle)$.*
2. Exponential evaluation contexts give applied variables*: Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ and $x \notin \mathsf{dom}(P)$. Then $x \in \mathsf{a}(P\langle x\rangle)$.*
3. Focusing multiplicative evaluation contexts on unapplied variables*: Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ and $x \in \mathcal{U}$. Then for every $t \in \Lambda$ there exists multiplicative evaluation context $P_t \in \mathcal{E}_{\mathcal{U}_t,\mathcal{A}_t}$ such that $P_t\langle x\rangle = P\langle t\rangle$, $x \notin \mathcal{U}_t \subset \mathcal{U}$, and $x \notin \mathcal{A}_t \subseteq \mathcal{A}$.*
4. Exponential evaluation contexts are multiplicative*: Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. Then $P \in \mathcal{E}_{\mathcal{V},\mathcal{B}}$, for some $\mathcal{V} \subseteq \mathcal{U}$ and $\mathcal{B} \subseteq \mathcal{A}$.*

Proof.

1. Easily provable by induction on $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. The only cases that are not provable by direct application of the *i.h.* are the following:
   - If $P = (\mathcal{H}^{@}, \epsilon) \in \mathcal{E}^{@}_{\mathsf{u}(\mathcal{H}^{@}),\mathsf{a}(\mathcal{H}^{@})}$, then the statement holds by application of Lemma 13.5.11.1 (Applicative term contexts give applied variables).
   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows
   $$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}] \in \mathcal{E}^{@}_{\mathcal{V}\cup\mathsf{u}(\mathcal{H}^{@}),\mathcal{B}\cup\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_2}$$
   where $P = Q\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}]$, $\mathcal{U} = \mathcal{V} \cup \mathsf{u}(\mathcal{H}^{@})$ and $\mathcal{A} = \mathcal{B} \cup \mathsf{a}(\mathcal{H}^{@})$. First, note that $y \in \mathsf{a}(Q\langle y\rangle)$—by *i.h.*—and so $y \in \mathsf{nv}(Q\langle y\rangle)$—by Lemma 8.1.1.2 (Unapplied, applied and needed variables). Moreover, note that $x \in \mathsf{a}(\mathcal{H}^{@}\langle x\rangle)$—by Lemma 13.5.11.1 (Applicative term contexts give applied variables). Therefore, Definition 36 (Applied variables) gives that $x \in \mathsf{a}(P\langle x\rangle) = (\mathsf{a}(Q\langle y\rangle) \setminus \{y\}) \cup \mathsf{a}(\mathcal{H}^{@}\langle x\rangle)$.
   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows
   $$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{A}} \quad x \notin \mathcal{A}}{Q\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{V}\setminus\{y\},\mathcal{A}}}\ \mathsf{E}_{\mathsf{NL}}$$
   where $P = Q\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle]$ and $\mathcal{U} = \mathcal{V} \setminus \{y\}$. By *i.h.*, $y \in \mathsf{a}(Q\langle y\rangle)$, and so $y \in \mathsf{nv}(Q\langle y\rangle)$—by Lemma 8.1.1.2 (Unapplied, applied and needed variables). Therefore, Definition 36 (Applied variables) gives that $x \in \mathsf{a}(P\langle x\rangle) = (\mathsf{a}(Q\langle y\rangle) \setminus \{y\}) \cup \{x\}$.

2. Let $x \in \mathcal{U}$ and $t \in \Lambda$. We proceed by case analysis on the last derivation rule in $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$:
   - Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{}{(\mathcal{H},\epsilon) \in \mathcal{E}_{\mathsf{u}(\mathcal{H}),\mathsf{a}(\mathcal{H})}}\ \mathsf{M_{AX}}$$

     with $P = (\mathcal{H},\epsilon)$, $\mathcal{U} = \mathsf{u}(\mathcal{H})$ and $\mathcal{A} = \mathsf{a}(\mathcal{H})$. Since $x \in \mathsf{u}(\mathcal{H}) \subseteq \mathsf{nv}(\mathcal{H})$—by Lemma 8.1.2 (Term contexts: Unapplied, applied and needed variables)—then by Lemma 13.5.10.2 (Focusing term contexts on unapplied variables) there exists $\mathcal{H}_t$ such that $\mathcal{H}_t\langle x\rangle = \mathcal{H}\langle t\rangle$, with $x \notin \mathsf{u}(\mathcal{H}_t) \subset \mathsf{u}(\mathcal{H})$ and $x \notin \mathsf{a}(\mathcal{H}_t) \subseteq \mathsf{a}(\mathcal{H})$. The statement then follows by taking $P_t = (\mathcal{H}_t,\epsilon)$.
   - Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}z] \in \mathcal{E}_{\mathsf{upd}(\mathcal{V},y,z),\mathsf{upd}(\mathcal{B},z,\tilde{x})}}\ \mathsf{M_{VAR}}$$

     with $P = Q@[y{\leftarrow}z]$, $\mathcal{U} = \mathsf{upd}(\mathcal{V},y,z)$ and $\mathcal{A} = \mathsf{upd}(\mathcal{B},y,z)$. The statement follows by *i.h.* on Q, yielding a $Q_t \in \mathcal{E}_{\mathcal{U}_t,\mathcal{A}_t}$ and finally deriving P_t by an application of $\mathsf{M_{VAR}}$ (resp. $\mathsf{M_{GC}}$) if $y \in (\mathcal{U}_t \cup \mathcal{A}_t)$ (resp. if $y \notin (\mathcal{U}_t \cup \mathcal{A}_t)$).
   - Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}i^+] \in \mathcal{E}_{(\mathcal{V}\setminus\{y\})\cup\mathsf{u}(i^+),(\mathcal{B}\setminus\{y\})\cup\mathsf{a}(i^+)}}\ \mathsf{M_I}$$

     with $P = Q@[y{\leftarrow}i^+]$, $\mathcal{U} = (\mathcal{V} \setminus \{y\}) \cup \mathsf{u}(i^+)$ and $\mathcal{A} = (\mathcal{B} \setminus \{y\}) \cup \mathsf{a}(i^+)$. The statement follows by *i.h.* on Q, yielding a $Q_t \in \mathcal{E}_{\mathcal{U}_t,\mathcal{A}_t}$ and finally deriving P_t by an application of $\mathsf{M_I}$ (resp. $\mathsf{M_{GC}}$) if $y \in (\mathcal{U}_t \cup \mathcal{A}_t)$ (resp. if $y \notin (\mathcal{U}_t \cup \mathcal{A}_t)$).
   - Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}s] \in \mathcal{E}_{\mathcal{U},\mathcal{A}}}\ \mathsf{M_{GC}}$$

     with $P = Q@[y{\leftarrow}s]$. The statement follows by *i.h.* on Q and finally deriving P_t by an application of $\mathsf{M_{GC}}$.
   - Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \setminus \mathcal{B})}{Q@[y{\leftarrow}v] \in \mathcal{E}_{\mathcal{V}\setminus\{y\},\mathcal{B}}}\ \mathsf{M_U}$$

     with $P = Q@[y{\leftarrow}v]$, $\mathcal{U} = \mathcal{V} \setminus \{y\}$ and $\mathcal{A} = \mathcal{B}$. The statement follows by *i.h.* on Q, yielding a $Q_t \in \mathcal{E}_{\mathcal{U}_t,\mathcal{A}_t}$ and finally deriving P_t by an application of $\mathsf{M_U}$ (resp. $\mathsf{M_{GC}}$) if $y \in (\mathcal{U}_t \setminus \mathcal{A}_t)$ (resp. if $y \notin (\mathcal{U}_t \setminus \mathcal{A}_t)$).
   - Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle z\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{V}\cup\mathsf{u}(\mathcal{H}),\mathcal{B}\cup\mathsf{a}(\mathcal{H})}}\ \mathsf{M_{HER}}$$

     with $P = Q\langle z\rangle@[y{\leftarrow}\mathcal{H}]$, $\mathcal{U} = \mathcal{V} \cup \mathsf{u}(\mathcal{H})$ and $\mathcal{A} = \mathcal{B} \cup \mathsf{a}(\mathcal{H})$. The statement follows by Lemma 13.5.10.2 (Focusing term contexts on unapplied variables) on $\mathcal{H}$, yielding a term context $\mathcal{H}_t$ and finally deriving P_t by an application of $\mathsf{M_{HER}}$ combining Q and $\mathcal{H}_t$.

3. *Exponential evaluation contexts are multiplicative*: By induction on the derivation of $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$:
   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{}{(\mathcal{H}^{@}, \epsilon) \in \mathcal{E}^{@}_{\mathsf{u}(\mathcal{H}^{@}),\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_1}$$

     with $P = (\mathcal{H}^{@}, \epsilon)$, $\mathcal{U} = \mathsf{u}(\mathcal{H}^{@})$ and $\mathcal{A} = \mathsf{a}(\mathcal{H}^{@})$. The statement simply follows by the fact that $\mathcal{H}^{@}$ is a term context.
   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{W},\mathcal{C}} \quad x \notin (\mathcal{W} \cup \mathcal{C})}{Q\langle x\rangle@[x{\leftarrow}\mathcal{H}^{@}] \in \mathcal{E}^{@}_{\mathcal{W}\cup\mathsf{u}(\mathcal{H}^{@}),\mathcal{C}\cup\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_2}$$

     where $P = Q\langle x\rangle@[x{\leftarrow}\mathcal{H}^{@}]$, $\mathcal{U} = \mathcal{W} \cup \mathsf{u}(\mathcal{H}^{@})$ and $\mathcal{A} = \mathcal{C} \cup \mathsf{a}(\mathcal{H}^{@})$. Since $\mathcal{H}^{@}$ is a term context, we get that

$$\frac{Q \in \mathcal{E}_{\mathcal{W},\mathcal{C}} \quad x \notin (\mathcal{W} \cup \mathcal{C})}{Q\langle x\rangle@[x{\leftarrow}\mathcal{H}^{@}] \in \mathcal{E}_{\mathcal{W}\cup\mathsf{u}(\mathcal{H}^{@}),\mathcal{C}\cup\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{M}_{\mathsf{HER}}$$

   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{W},\mathcal{C}} \quad x \in (\mathcal{W} \cup \mathcal{C})}{Q@[x{\leftarrow}y] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{W},x,y),\mathsf{upd}(\mathcal{C},x,y)}}\ \mathsf{E}_{\mathsf{VAR}}$$

     where $P = Q@[x{\leftarrow}y]$, $\mathcal{U} = \mathsf{upd}(\mathcal{W}, x, y)$ and $\mathcal{A} = \mathsf{upd}(\mathcal{C}, x, y)$. By *i.h.*, $Q \in \mathcal{E}_{\tilde{\mathcal{W}},\tilde{\mathcal{C}}}$ for some $\tilde{\mathcal{W}} \subseteq \mathcal{W}$ and $\tilde{\mathcal{C}} \subseteq \mathcal{C}$. Case analysis on whether $x \in (\tilde{\mathcal{W}} \cup \tilde{\mathcal{C}})$:
     - Let $x \in (\tilde{\mathcal{W}} \cup \tilde{\mathcal{C}})$. Then we can derive $Q@[x{\leftarrow}y] \in \mathcal{E}_{\mathsf{upd}(\tilde{\mathcal{W}},x,y),\mathsf{upd}(\tilde{\mathcal{C}},x,y)}$ via rule $\mathsf{M}_{\mathsf{VAR}}$.
     - Let $x \notin (\tilde{\mathcal{W}} \cup \tilde{\mathcal{C}})$. Then we can derive $Q@[x{\leftarrow}y] \in \mathcal{E}_{\tilde{\mathcal{W}},\tilde{\mathcal{C}}}$ via rule M_{GC}.
   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{W},\mathcal{C}} \quad x \in (\mathcal{W} \cup \mathcal{C})}{Q@[x{\leftarrow}i^{+}] \in \mathcal{E}^{@}_{(\mathcal{W}\setminus\{x\})\cup\mathsf{u}(i^{+}),(\mathcal{C}\setminus\{x\})\cup\mathsf{a}(i^{+})}}\ \mathsf{E}_{\mathsf{I}}$$

     where $P = Q@[x{\leftarrow}i^{+}]$, $\mathcal{U} = (\mathcal{W} \setminus \{x\}) \cup \mathsf{u}(i^{+})$ and $\mathcal{A} = (\mathcal{C} \setminus \{x\}) \cup \mathsf{a}(i^{+})$. By *i.h.*, $Q \in \mathcal{E}_{\tilde{\mathcal{W}},\tilde{\mathcal{C}}}$ for some $\tilde{\mathcal{W}} \subseteq \mathcal{W}$ and $\tilde{\mathcal{C}} \subseteq \mathcal{C}$. Case analysis on whether $x \in (\tilde{\mathcal{W}} \cup \tilde{\mathcal{C}})$:
     - Let $x \in (\tilde{\mathcal{W}} \cup \tilde{\mathcal{C}})$. Then we can derive $Q@[x{\leftarrow}i^{+}] \in \mathcal{E}_{(\tilde{\mathcal{V}}\setminus\{x\})\cup\mathsf{u}(i^{+}),(\tilde{\mathcal{B}}\setminus\{x\})\cup\mathsf{a}(i^{+})}$ via rule M_{I}.
     - Let $x \notin (\tilde{\mathcal{W}} \cup \tilde{\mathcal{C}})$. Then we can derive $Q@[x{\leftarrow}i^{+}] \in \mathcal{E}_{\tilde{\mathcal{W}},\tilde{\mathcal{C}}}$ via rule M_{GC}. Note that $\tilde{\mathcal{W}} = \tilde{\mathcal{W}} \setminus \{x\} \subseteq (\mathcal{W} \setminus \{x\}) \subseteq (\mathcal{W} \setminus \{x\}) \cup \mathsf{u}(i^{+}) = \mathcal{U}$ and, similarly, $\tilde{\mathcal{C}} \subseteq \mathcal{A}$.
   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad x \notin (\mathcal{U} \cup \mathcal{A})}{Q@[x{\leftarrow}t] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E}_{\mathsf{GC}}$$

     where $P = Q@[x{\leftarrow}t]$. By *i.h.*, $Q \in \mathcal{E}^{@}_{\tilde{\mathcal{U}},\tilde{\mathcal{A}}}$ for some $\tilde{\mathcal{U}} \subseteq \mathcal{U}$ and $\tilde{\mathcal{A}} \subseteq \mathcal{A}$, and we can derive $Q@[x{\leftarrow}t] \in \mathcal{E}_{\tilde{\mathcal{U}},\tilde{\mathcal{A}}}$ via rule M_{GC}.

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{W},\mathcal{C}} \quad x \in (\mathcal{W} \setminus \mathcal{C})}{Q@[x{\leftarrow}v] \in \mathcal{E}^{@}_{\mathcal{W}\setminus\{x\},\mathcal{C}}}\ \mathsf{E_U}$$

where $Q = P@[x{\leftarrow}v]$, $\mathcal{U} = \mathcal{W} \setminus \{x\}$ and $\mathcal{A} = \mathcal{C}$. By *i.h.*, $Q \in \mathcal{E}_{\tilde{\mathcal{W}},\tilde{\mathcal{C}}}$ for some $\tilde{\mathcal{W}} \subseteq \mathcal{W}$ and $\tilde{\mathcal{C}} \subseteq \mathcal{C}$. Case analysis on whether $x \in (\tilde{\mathcal{W}} \setminus \tilde{\mathcal{C}})$:
  - Let $x \in (\tilde{\mathcal{W}} \setminus \tilde{\mathcal{C}})$. Then we can derive $Q@[x{\leftarrow}t] \in \mathcal{E}_{\tilde{\mathcal{W}}\setminus\{x\},\tilde{\mathcal{C}}}$ via rule $\mathsf{M_U}$.
  - Let $x \notin (\tilde{\mathcal{W}} \setminus \tilde{\mathcal{C}})$. Then we can derive $Q@[x{\leftarrow}t] \in \mathcal{E}_{\tilde{\mathcal{W}},\tilde{\mathcal{C}}}$ via rule $\mathsf{M_{GC}}$.
- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{P \in \mathcal{E}^{@}_{\mathcal{W},\mathcal{C}} \quad x \notin \mathcal{C}}{P\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{W}\setminus\{x\},\mathcal{C}}}\ \mathsf{E_{NL}}$$

where $P = P\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle], \mathcal{U} = \mathcal{W} \setminus \{x\}$ and $\mathcal{A} = \mathcal{C}$. By *i.h.*, $P \in \mathcal{E}_{\tilde{\mathcal{W}},\tilde{\mathcal{C}}}$ for some $\tilde{\mathcal{W}} \subseteq \mathcal{W}$ and $\tilde{\mathcal{C}} \subseteq \mathcal{C}$. Case analysis on whether $x \in \tilde{\mathcal{W}}$:
  - Let $x \in \tilde{\mathcal{W}}$. By Lemma 13.5.12.3 (Focusing multiplicative evaluation contexts on unapplied variables), there exists $P_x \in \mathcal{E}_{\tilde{\mathcal{W}}',\tilde{\mathcal{C}}'}$ such that $P_x\langle x\rangle = P\langle x\rangle$, with $x \notin \tilde{\mathcal{W}}' \subset \tilde{\mathcal{W}}$ and $x \notin \tilde{\mathcal{C}}' \subseteq \tilde{\mathcal{C}}$. Hence, we can derive

$$\frac{P_x \in \mathcal{E}_{\tilde{\mathcal{W}}',\tilde{\mathcal{C}}'} \quad x \notin (\tilde{\mathcal{W}}' \cup \tilde{\mathcal{C}}')}{P_x\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}_{\tilde{\mathcal{W}}',\tilde{\mathcal{C}}'}}\ \mathsf{M_{HER}}$$

    noting in particular that $P_x\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle] = P\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle]$.
  - Let $x \notin \tilde{\mathcal{W}}$. Then we can derive $P\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}_{\tilde{\mathcal{W}},\tilde{\mathcal{C}}}$ via rule $\mathsf{M_{HER}}$.

□

Lemma 13.5.13 (Rewriting evaluation contexts)**.**

1. Focusing multiplicative evaluation contexts on applied variables*: Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ and $x \in \mathcal{A}$. Then for every normal term n, there exists exponential evaluation context $P_n \in \mathcal{E}^{@}_{\mathcal{U}_n,\mathcal{A}_n}$ such that $P_n\langle x\rangle = P\langle n\rangle$, with $\mathcal{U}_n \subseteq \mathcal{U}$ and $x \notin \mathcal{A}_n \subset \mathcal{A}$.*
2. Focusing exponential evaluation contexts on unapplied variables*: Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ and $x \in \mathcal{U}$. Then for every normal term n, there exists exponential evaluation context $P_n \in \mathcal{E}_{\mathcal{U}_n,\mathcal{A}_n}$ such that $P_n\langle x\rangle = P\langle n\rangle$, with $x \notin \mathcal{U}_n \subset \mathcal{U}$ and $\mathcal{A}_n \subseteq \mathcal{A}$.*
3. Focusing exponential evaluation contexts on applied variables*: Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ and $x \in \mathcal{A}$. Then for every normal term n, there exists exponential evaluation context $P_n \in \mathcal{E}^{@}_{\mathcal{U}_n,\mathcal{A}_n}$ such that $P_n\langle x\rangle = P\langle n\rangle$, with $\mathcal{U}_n \subseteq \mathcal{U}$ and $x \notin \mathcal{A}_n \subset \mathcal{A}$.*

Proof. All three statements are proven by mutual induction on the derivation of $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ or $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$:

1. *Focusing multiplicative evaluation contexts on applied variables*:
   Let $x \in \mathcal{A}$ and t be normal. We proceed by case analysis on the last derivation rule in $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$:
   - Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{}{(\mathcal{H},\epsilon) \in \mathcal{E}_{\mathsf{u}(\mathcal{H}),\mathsf{a}(\mathcal{H})}}\ \mathsf{M_{AX}}$$

with $P = (\mathcal{H}, \epsilon), \mathcal{U} = \mathsf{u}(\mathcal{H})$ and $\mathcal{A} = \mathsf{a}(\mathcal{H})$. By application of Lemma 13.5.11.3 (Focusing term contexts on applied variables) on $\mathcal{H}$, we get an applicative term context $\mathcal{J}_t^{@}$ such that $\mathcal{J}_t^{@}\langle x\rangle = \mathcal{H}\langle t\rangle$, with $\mathsf{u}(\mathcal{J}_t^{@}) \subseteq \mathsf{u}(\mathcal{H})$ and $x \notin \mathsf{a}(\mathcal{J}_t^{@}) \subset \mathsf{a}(\mathcal{H})$. The statement then follows by taking $P_t \coloneqq (\mathcal{J}_t^{@}, \epsilon)$.

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}z] \in \mathcal{E}_{\mathsf{upd}(\mathcal{V},y,z),\mathsf{upd}(\mathcal{B},y,z)}}\ \mathsf{M_{VAR}}$$

with $P = Q@[y{\leftarrow}z], \mathcal{U} = \mathsf{upd}(\mathcal{V}, y, z)$ and $\mathcal{A} = \mathsf{upd}(\mathcal{B}, y, z)$. We proceed by case analysis on whether $y \in \mathcal{B}$:

  - Let $y \notin \mathcal{B}$. Since then $\mathcal{B} = \mathsf{upd}(\mathcal{B}, y, z) \ni x$, application of the *i.h.* (Focusing multiplicative evaluation contexts on applied variables) gives us $Q_t \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$, with $\mathcal{V}_t \subseteq \mathcal{V}$ and $x \notin \mathcal{B}_t \subset \mathcal{B}$. We can then derive $P_t \coloneqq Q_t@[y{\leftarrow}z] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{V}_t,y,z),\mathsf{upd}(\mathcal{B}_t,y,z)}$ (resp., $P_t \coloneqq Q_t@[y{\leftarrow}z] \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$) via rule $\mathsf{E_{VAR}}$ (resp., rule $\mathsf{E_{GC}}$) if $y \in (\mathcal{V}_t \cup \mathcal{B}_t)$ (resp., if $y \notin (\mathcal{V}_t \cup \mathcal{B}_t)$).
  - Let $y \in \mathcal{B}$. Note that then then $\mathsf{upd}(\mathcal{B}, y, z) = (\mathcal{B} \setminus \{y\}) \cup \{z\}$. We do case analysis on whether $x \in \mathcal{B}$:
    - If $x \in \mathcal{B}$, then we can prove the statement similarly to how we proved it for when $y \notin \mathcal{B}$.
    - If $x \notin \mathcal{B}$, then it must be that $x = z$. By *i.h.* (Focusing multiplicative evaluation contexts on applied variables) with respect to $y \in \mathcal{B}$, there exists $Q_t \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle y\rangle = Q\langle t\rangle$, with $\mathcal{V}_t \subseteq \mathcal{V}$ and $y \notin \mathcal{B}_t \subset \mathcal{B}$. We can then derive $P_t \coloneqq Q_t\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{V}_t\setminus\{y\},\mathcal{B}_t}$ via rule $\mathsf{E_{NL}}$.

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}i^+] \in \mathcal{E}_{(\mathcal{V}\setminus\{y\})\cup\mathsf{u}(i^+),(\mathcal{B}\setminus\{y\})\cup\mathsf{a}(i^+)}}\ \mathsf{M_I}$$

with $P = Q@[y{\leftarrow}i^+], \mathcal{U} = (\mathcal{V} \setminus \{y\}) \cup \mathsf{u}(i^+)$ and $\mathcal{A} = (\mathcal{B} \setminus \{y\}) \cup \mathsf{a}(i^+)$. Case analysis on whether $x \in \mathcal{B}$:

  - Let $x \in \mathcal{B}$. By *i.h.* (Focusing multiplicative evaluation contexts on applied variables), there exists $Q_t \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$, with $\mathcal{V}_t \subseteq \mathcal{V}$ and $x \notin \mathcal{B}_t \subset \mathcal{B}$. First, note that if $y \notin (\mathcal{V}_t \cup \mathcal{B}_t)$, then we can derive $P_t \coloneqq Q_t@[y{\leftarrow}i^+] \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$ via rule $\mathsf{E_{GC}}$.
    Second, if $y \in (\mathcal{V}_t \cup \mathcal{B}_t)$ and $x \notin \mathsf{a}(i^+)$, then we can derive $P_t \coloneqq Q_t@[y{\leftarrow}i^+] \in \mathcal{E}^{@}_{(\mathcal{V}_t\setminus\{y\})\cup\mathsf{u}(i^+),(\mathcal{B}_t\setminus\{y\})\cup\mathsf{a}(i^+)}$ via rule $\mathsf{E_I}$.
    Finally, let us consider the case where $y \in (\mathcal{V}_t \cup \mathcal{B}_t)$ and $x \in \mathsf{a}(i^+)$. Notice that the latter implies—by Lemma 13.5.11.2 (Focusing useful inert terms on applied variables)—the existence of an applicative term context $\mathcal{H}_x^{@}$ such that $\mathcal{H}_x^{@}\langle x\rangle = i^+$, with $\mathsf{u}(\mathcal{H}_x^{@}) \subseteq \mathsf{u}(i^+)$ and $x \notin \mathsf{a}(\mathcal{H}_x^{@}) \subset \mathsf{a}(i^+)$. We now proceed by case analysis on how exactly $y \in (\mathcal{V}_t \cup \mathcal{B}_t)$:
    - Let $y \in \mathcal{V}_t$. By *i.h.* (Focusing exponential evaluation contexts on unapplied variables), there exists $R_x \in \mathcal{E}_{\mathcal{W}_x,\mathcal{C}_x}$ such that $R_x\langle y\rangle = Q_t\langle x\rangle = Q\langle t\rangle$, with $y \notin \mathcal{W}_x \subset \mathcal{V}_t \subseteq \mathcal{V}$ and $\mathcal{C}_x \subseteq \mathcal{B}_t \subset \mathcal{B}$. Case analysis on whether $y \in \mathcal{C}_x$:
      - If $y \notin \mathcal{C}_x$, then we can apply rule $\mathsf{E_{AX_2}}$ to derive

$$P_t \coloneqq R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}_x^{@}] \in \mathcal{E}^{@}_{\mathcal{W}_x\cup\mathsf{u}(\mathcal{H}_x^{@}),\mathcal{C}_x\cup\mathsf{a}(\mathcal{H}_x^{@})}$$

· Let $y \in \mathcal{C}_x$. By *i.h.* (Focusing multiplicative evaluation contexts on applied variables), there exists $S_y \in \mathcal{E}^{@}_{\mathcal{X}_y,\mathcal{D}_y}$ such that $S_y\langle y\rangle = R_x\langle y\rangle = Q_t\langle x\rangle = Q\langle t\rangle$, with $\mathcal{X}_y \subseteq \mathcal{W}_x \subset \mathcal{V}_t \subseteq \mathcal{V}$ and $y \notin \mathcal{D}_y \subset \mathcal{C}_x \subseteq \mathcal{B}_t \subset \mathcal{B}$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $S_y \in \mathcal{E}_{\tilde{\mathcal{X}}_y,\tilde{\mathcal{D}}_y}$ for some $\tilde{\mathcal{X}}_y \subseteq \mathcal{X}_y$ and $\tilde{\mathcal{D}}_y \subseteq \mathcal{D}_y$. We can then derive $P_t \coloneqq S_y\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\tilde{\mathcal{X}}_y \cup \mathsf{u}(\mathcal{H}^{@}_x),\tilde{\mathcal{D}}_y \cup \mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E_{NL}}$.

∗ Let $y \notin \mathcal{V}_t$. Then $y \in \mathcal{B}_t$, and by *i.h.* (Focusing exponential evaluation contexts on applied variables) there exists $R_x \in \mathcal{E}^{@}_{\mathcal{W}_x,\mathcal{C}_x}$ such that $R_x\langle y\rangle = Q_t\langle x\rangle = Q\langle t\rangle$, with $\mathcal{W}_x \subseteq \mathcal{V}_t \subseteq \mathcal{V}$ and $y \notin \mathcal{C}_x \subset \mathcal{B}_t \subset \mathcal{B}$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $R_x \in \mathcal{E}_{\tilde{\mathcal{W}}_x,\tilde{\mathcal{C}}_x}$ for some $\tilde{\mathcal{W}}_x \subseteq \mathcal{W}_x$ and $\tilde{\mathcal{C}}_x \subseteq \mathcal{C}_x$. We can then derive $P_t \coloneqq R_x@[y{\leftarrow}\mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\tilde{\mathcal{W}}_x \cup \mathsf{u}(\mathcal{H}^{@}_x),\tilde{\mathcal{C}}_x \cup \mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E_{AX_2}}$.

– Let $x \notin \mathcal{B}$. Then it must be that $x \in \mathsf{a}(i^+)$, which implies—by Lemma 13.5.11.2 (Focusing useful inert terms on applied variables)—the existence of an applicative term context $\mathcal{H}^{@}_x$ such that $\mathcal{H}^{@}_x\langle x\rangle = i^+$, with $\mathsf{u}(\mathcal{H}^{@}_x) \subseteq \mathsf{u}(i^+)$ and $x \notin \mathsf{a}(\mathcal{H}^{@}_x) \subset \mathsf{a}(i^+)$. We proceed by case analysis on whether $y \in \mathcal{V}$:

∗ If $y \in \mathcal{V}$, then *i.h.* (Focusing multiplicative evaluation contexts on applied varibales) gives the existence of $Q_t \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle y\rangle = Q\langle t\rangle$ such that $x \notin \mathcal{V}_t \subset \mathcal{V}$ and $\mathcal{B}_t \subseteq \mathcal{B}$. Case analyss on whether $y \in \mathcal{B}_t$:

· If $y \notin \mathcal{B}_t$, then application of the *i.h.* (Focusing multiplicative evaluation contexts on applied variables) gives $R_y \in \mathcal{E}^{@}_{\mathcal{W}_y,\mathcal{C}_y}$ such that $R_y\langle y\rangle = Q_t\langle y\rangle = Q\langle t\rangle$, with $\mathcal{W}_y \subseteq \mathcal{V}_t \subset \mathcal{V}$ and $y \notin \mathcal{C}_y \subset \mathcal{B}_t \subseteq \mathcal{B}$. In addition, $R_y \in \mathcal{E}_{\tilde{\mathcal{W}}_y,\tilde{\mathcal{C}}_y}$ for some $\tilde{\mathcal{W}}_y \subseteq \mathcal{W}_y$ and $\tilde{\mathcal{C}}_y \subseteq \mathcal{C}_y$. We can then derive $P_t \coloneqq R_y\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\tilde{\mathcal{W}}_y \cup \mathsf{u}(\mathcal{H}^{@}_x),\tilde{\mathcal{C}}_y \cup \mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E_{AX_2}}$.

· If $y \in \mathcal{B}_t$, then we can derive $P_t \coloneqq Q_t\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\mathcal{V}_t \cup \mathsf{u}(\mathcal{H}^{@}_x),\mathcal{B}_t \cup \mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E_{AX_2}}$.

∗ Let $y \notin \mathcal{V}$. Then it must be that $y \in \mathcal{B}$, and so applying the *i.h.* gives $R_t \in \mathcal{E}^{@}_{\mathcal{W}_t,\mathcal{C}_t}$ such that $R_t\langle y\rangle = Q\langle t\rangle$, with $\mathcal{W}_t \subseteq \mathcal{V}$ and $y \notin \mathcal{C}_t \subset \mathcal{B}$. Moreover, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) proves that $R_t \in \mathcal{E}_{\mathcal{X},\mathcal{D}}$ such that $\mathcal{X} \subseteq \mathcal{W}_t$ and $\mathcal{D} \subseteq \mathcal{C}_t$. Thus, we can derive $P_t \coloneqq R_t\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\mathcal{X} \cup \mathsf{u}(\mathcal{H}^{@}_x),\mathcal{D} \cup \mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E_{AX_2}}$.

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{Q@[y{\leftarrow}u] \in \mathcal{E}_{\mathcal{U},\mathcal{A}}}\ \mathsf{M_{GC}}$$

with $P = Q@[y{\leftarrow}u]$. By *i.h.* (Focusing multiplicative evaluation contexts on applied variables), there exists $Q_t \in \mathcal{E}^{@}_{\mathcal{U}_t,\mathcal{A}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$, with $\mathcal{U}_t \subseteq \mathcal{U}$ and $x \notin \mathcal{A}_t \subset \mathcal{A}$. We can then derive $P_t \coloneqq Q_t@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathcal{U}_t,\mathcal{A}_t}$ via rule $\mathsf{E_{GC}}$.

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \setminus \mathcal{B})}{Q@[y{\leftarrow}v] \in \mathcal{E}_{\mathcal{V}\setminus\{y\},\mathcal{B}}}\ \mathsf{M_U}$$

with $P = Q@[y{\leftarrow}v], \mathcal{U} = \mathcal{V}\setminus\{y\}$ and $\mathcal{A} = \mathcal{B}$. By *i.h.* (Focusing multiplicative evaluation contexts on applied variables), there exists $Q_t \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$, with

$\mathcal{V}_t \subseteq \mathcal{V}$ and $x \notin \mathcal{B}_t \subset \mathcal{B}$. We can then derive $P_t \coloneqq Q_t@[y{\leftarrow}t] \in \mathcal{E}^{@}_{\mathcal{V}_t \setminus \{y\},\mathcal{B}_t}$ (resp. $P_t \coloneqq Q_t@[y{\leftarrow}t] \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$) via rule $\mathsf{E_U}$ (resp. rule $\mathsf{E_{GC}}$) if $y \in \mathcal{V}_t$ (resp. if $y \notin \mathcal{V}_t$).

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{Q\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}_{\mathcal{U},\mathcal{A}}}\ \mathsf{M_{HER}}$$

with $P = Q\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle]$. By *i.h.* (Focusing multiplicative evaluation contexts on applied variables), there exists $Q_y \in \mathcal{E}^{@}_{\mathcal{U}_y,\mathcal{A}_y}$ such that $Q_y\langle x\rangle = Q\langle y\rangle$, with $\mathcal{U}_y \subseteq \mathcal{U}$ and $x \notin \mathcal{A}_y \subset \mathcal{A}$. We can then derive $P_t \coloneqq Q_y@[y{\leftarrow}t] \in \mathcal{E}^{@}_{\mathcal{U}_y,\mathcal{A}_t}$ via rule $\mathsf{E_{GC}}$.

2. *Focusing exponential evaluation contexts on unapplied variables*: Let $x \in \mathcal{U}$ and $t \in \Lambda$ be normal. We proceed by case analysis on the last derivation rule in $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$:
   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{}{(\mathcal{H}^{@},\epsilon) \in \mathcal{E}^{@}_{\mathsf{u}(\mathcal{H}^{@}),\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E_{AX_1}}$$

   with $P = (\mathcal{H}^{@},\epsilon)$, $\mathcal{U} = \mathsf{u}(\mathcal{H}^{@})$ and $\mathcal{A} = \mathsf{a}(\mathcal{H}^{@})$. The statement is given by applying Lemma 13.5.10.2 (Focusing terms contexts on unapplied variables) on $\mathcal{H}^{@}$, yielding a term context $\mathcal{H}_t$ such that $\mathcal{H}_t\langle x\rangle = \mathcal{H}^{@}\langle t\rangle$, with $x \notin \mathsf{u}(\mathcal{H}_t) \subset \mathsf{u}(\mathcal{H}^{@})$ and $\mathsf{a}(\mathcal{H}_t) \subseteq \mathsf{a}(\mathcal{H}^{@})$, and finally taking $P_t \coloneqq (\mathcal{H}_t,\epsilon)$.
   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}] \in \mathcal{E}^{@}_{\mathcal{V}\cup\mathsf{u}(\mathcal{H}^{@}),\mathcal{B}\cup\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E_{AX_2}}$$

   with $P = Q\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}]$, $\mathcal{U} = \mathcal{V} \cup \mathsf{u}(\mathcal{H}^{@})$ and $\mathcal{A} = \mathcal{B} \cup \mathsf{a}(\mathcal{H}^{@})$. Case analysis on whether $x \in \mathcal{V}$:
     - Let $x \in \mathcal{V}$. By Lemma 13.5.12.3 (Focusing multiplicative evaluation contexts on unapplied variables), there exists $Q_y \in \mathcal{E}_{\mathcal{V}_y,\mathcal{B}_y}$ such that $Q_y\langle x\rangle = Q\langle y\rangle$, with $x \notin \mathcal{V}_y \subset \mathcal{V}$ and $\mathcal{B}_y \subseteq \mathcal{B}$. We can then derive $P_t \in \mathcal{E}_{\mathcal{U}_t,\mathcal{A}_t}$ as follows

$$\frac{Q_y \in \mathcal{E}_{\mathcal{V}_y,\mathcal{B}_y} \quad y \notin (\mathcal{V}_y \cup \mathcal{B}_y)}{Q_y@[y{\leftarrow}\mathcal{H}^{@}\langle t\rangle] \in \mathcal{E}_{\mathcal{V}_y,\mathcal{B}_y}}\ \mathsf{M_{GC}}$$

     - Let $x \notin \mathcal{V}$. Then $x \in \mathsf{u}(\mathcal{H}^{@})$. By Lemma 13.5.10.2 (Focusing term contexts on unapplied variables), there exists term context $\mathcal{H}_t$ such that $\mathcal{H}_t\langle x\rangle = \mathcal{H}^{@}\langle t\rangle$, with $x \notin \mathsf{u}(\mathcal{H}_t) \subset \mathsf{u}(\mathcal{H}^{@})$ and $\mathsf{a}(\mathcal{H}_t) \subseteq \mathsf{a}(\mathcal{H}^{@})$. We can then derive $P_t \in \mathcal{E}_{\mathcal{U}_t,\mathcal{A}_t}$ as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle y\rangle@[y{\leftarrow}\mathcal{H}_t] \in \mathcal{E}^{@}_{\mathcal{V}\cup\mathsf{u}(\mathcal{H}_t),\mathcal{B}\cup\mathsf{a}(\mathcal{H}_t)}}\ \mathsf{E_{AX_2}}$$

   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}z] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{V},y,z),\mathsf{upd}(\mathcal{B},y,z)}}\ \mathsf{E_{VAR}}$$

   with $P = Q@[y{\leftarrow}z]$, $\mathcal{U} = \mathsf{upd}(\mathcal{V},y,z)$ and $\mathcal{A} = \mathsf{upd}(\mathcal{B},y,z)$. We proceed by case analysis on whether $y \in \mathcal{V}$:

– Let $y \notin \mathcal{V}$. Then $\mathcal{V} = \mathsf{upd}(\mathcal{V}, y, z) \ni x$ and so application of the *i.h.* (Focusing exponential evaluation contexts on unapplied variables) yields multiplicative evaluation context $Q_t \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$, with $x \notin \mathcal{V}_t \subset \mathcal{V}$ and $\mathcal{B}_t \subseteq \mathcal{B}$. Case analysis on whether $y \in \mathcal{B}_t$:
  * Let $y \in \mathcal{B}_t$. Then we can derive $P_t \coloneqq Q_t@[y{\leftarrow}z] \in \mathcal{E}_{\mathsf{upd}(\mathcal{V}_t,y,z),\mathsf{upd}(\mathcal{B}_t,y,z)}$ via rule $\mathsf{E_{VAR}}$.
  * Let $y \notin \mathcal{B}_t$. Then we can derive $P_t \coloneqq Q_t@[y{\leftarrow}z] \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ via rule $\mathsf{E_{VAR}}$.
– Let $y \in \mathcal{V}$. Then $\mathcal{U} = (\mathcal{V} \setminus \{y\}) \cup \{z\}$. Case analysis on whether $x \in (\mathcal{V} \setminus \{y\})$ and on whether $x = z$:
  * Let $x \neq z$. Then $x \in (\mathcal{V}\setminus\{y\})$. By *i.h.* (Focusing exponential evaluation contexts on unapplied variables), there exists $Q_t \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$, with $x \notin \mathcal{V}_t \subset \mathcal{V}$ and $\mathcal{B}_t \subseteq \mathcal{B}$.
    Finally, if $y \in (\mathcal{V}_t \cup \mathcal{B}_t)$, then we can derive $P_t \coloneqq Q_t@[y{\leftarrow}z] \in \mathcal{E}_{\mathsf{upd}(\mathcal{V}_t,y,z),\mathsf{upd}(\mathcal{B}_t,y,z)}$ via rule $\mathsf{M_{VAR}}$. If instead $y \notin (\mathcal{V}_t \cup \mathcal{B}_t)$, then we can derive $P_t \coloneqq Q_t@[y{\leftarrow}z] \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$.
  * Let $x = z$ and $x \notin (\mathcal{V} \setminus \{y\})$. By *i.h.* (Focusing exponential evaluation contexts on unapplied variables), there exists $Q_t \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle y\rangle = Q\langle t\rangle$, with $y \notin \mathcal{V}_t \subset \mathcal{V}$ and $\mathcal{B}_t \subseteq \mathcal{B}$. Case analysis on whether $y \in \mathcal{B}_t$:
    · Let $y \in \mathcal{B}_t$. By *i.h.* (Focusing multiplicative evaluation contexts on applied variables), there exists $R_y \in \mathcal{E}^{@}_{\mathcal{W}_y,\mathcal{C}_y}$ such that $R_y\langle y\rangle = Q_t\langle x\rangle = Q\langle t\rangle$, with $\mathcal{W}_y \subseteq \mathcal{V}_t \subset \mathcal{V}$ and $y \notin \mathcal{C}_y \subset \mathcal{B}_t \subseteq \mathcal{B}$. Moreover, by Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative), $R_y \in \mathcal{E}_{\tilde{\mathcal{W}}_y,\tilde{\mathcal{C}}_y}$ for some $\tilde{\mathcal{W}}_y \subseteq \mathcal{W}_y$ and $\tilde{\mathcal{C}}_y \subseteq \mathcal{C}_y$. We can finally derive $P_t \coloneqq R_y\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}_{\tilde{\mathcal{W}}_y\setminus\{y\},\tilde{\mathcal{C}}_y}$ via rule $\mathsf{M_{HER}}$.
    · Let $y \notin \mathcal{B}_t$. Then we can derive $P_t \coloneqq Q_t\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}_{\mathcal{V}_t\setminus\{y\},\mathcal{B}_t}$ via rule $\mathsf{M_{HER}}$.
  * Let $x = z$ and $x \in (\mathcal{V} \setminus \{y\})$. By *i.h.* (Focusing exponential evaluation contexts on unapplied variables), there exists $Q_t \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$, with $x \notin \mathcal{V}_t \subset \mathcal{V}$ and $\mathcal{B}_t \subseteq \mathcal{B}$. Cases analysis on whether $y \in \mathcal{B}_t$:
    · Let $y \in \mathcal{B}_t$. By *i.h.* (Focusing multiplicative evaluation contexts on applied variables), there exists $R_x \in \mathcal{E}^{@}_{\mathcal{W}_x,\mathcal{C}_x}$ such that $R_x\langle y\rangle = Q_t\langle x\rangle = Q\langle t\rangle$, with $\mathcal{W}_x \subseteq \mathcal{V}_t \subset \mathcal{V}$ and $y \notin \mathcal{C}_x \subset \mathcal{B}_t \subseteq \mathcal{B}$. Moreover, by Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative), $R_x \in \mathcal{E}_{\tilde{\mathcal{V}}_x,\tilde{\mathcal{B}}_x}$ for some $\tilde{\mathcal{V}}_x \subseteq \mathcal{W}_x$ and $\tilde{\mathcal{B}}_x \subseteq \mathcal{C}_x$.
      Finally, if $y \in \tilde{\mathcal{V}}_x$, then Lemma 13.5.12.3 (Focusing multiplicative evaluation contexts on unapplied variables) gives multiplicative evaluation context $S_y \in \mathcal{E}_{\mathcal{X}_y,\mathcal{D}_y}$ such that $S_y\langle y\rangle = R_x\langle y\rangle = Q_t\langle x\rangle = Q\langle t\rangle$, with $y \notin \mathcal{X}_y \subset \mathcal{W}_x \subseteq \mathcal{V}_t \subset \mathcal{V}$ and $\mathcal{D}_y \subseteq \mathcal{C}_x \subset \mathcal{B}_t \subseteq \mathcal{B}$; we can derive $P_t \coloneqq S_y\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}_{\mathcal{X}_y\setminus\{y\},\mathcal{D}_y}$ via rule $\mathsf{M_{HER}}$. If instead $y \notin \tilde{\mathcal{V}}_x$, then we can derive $P_t \coloneqq R_x\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}_{\tilde{\mathcal{V}}_x\setminus\{y\},\tilde{\mathcal{B}}_x}$ via rule $\mathsf{M_{HER}}$.
    · Let $y \notin \mathcal{B}_t$. If $y \notin \mathcal{V}_t$, then we can simply derive $Q_t@[y{\leftarrow}z] \in \mathcal{E}_{\mathcal{V}_t\setminus\{y\},\mathcal{B}_t}$ via rule $\mathsf{M_{HER}}$. If $y \in \mathcal{V}_t$ instead, then application of Lemma 13.5.12.3 (Focusing mutliplicative evaluation contexts on unapplied varibales) gives the existence of $R_x \in \mathcal{E}_{\mathcal{W}_x,\mathcal{C}_x}$ such that $R_x\langle y\rangle = Q_t\langle x\rangle = Q\langle t\rangle$, with $y \notin \mathcal{W}_x \subset \mathcal{V}_t \subset \mathcal{V}$ and $\mathcal{C}_x \subseteq \mathcal{B}_t \subseteq \mathcal{B}$; we can derive $P_t \coloneqq R_x\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}_{\mathcal{W}_x\setminus\{y\},\mathcal{C}_x}$ via rule $\mathsf{M_{HER}}$.

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}i^+] \in \mathcal{E}^{@}_{(\mathcal{V}\setminus\{y\})\cup \mathsf{u}(i^+),(\mathcal{B}\setminus\{y\})\cup \mathsf{a}(i^+)}}\ \mathsf{E_I}$$

with $P = Q@[y{\leftarrow}i^+]$, $\mathcal{U} = (\mathcal{V} \setminus \{y\}) \cup \mathsf{u}(i^+)$ and $\mathcal{A} = (\mathcal{B} \setminus \{y\}) \cup \mathsf{a}(i^+)$. Case analysis on whether $x \in \mathcal{V}$:

  - Let $x \in \mathcal{V}$. By *i.h.* (Focusing exponential evaluation contexts on unapplied variables), there exists $Q_t \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$, with $x \notin \mathcal{V}_t \subset \mathcal{V}$ and $\mathcal{B}_t \subseteq \mathcal{B}$. Case analysis on whether $x \in \mathsf{u}(i^+)$:
    - Let $x \notin \mathsf{u}(i^+)$. If $y \in (\mathcal{V}_t \cup \mathcal{B}_t)$, then we can derive $P \coloneqq Q_t@[y{\leftarrow}i^+] \in \mathcal{E}_{(\mathcal{V}_t\setminus\{y\})\cup \mathsf{u}(i^+),(\mathcal{B}_t\setminus\{y\})\cup \mathsf{a}(i^+)}$ via rule $\mathsf{M_I}$. If $y \notin (\mathcal{V}_t \cup \mathcal{B}_t)$, then we can derive $P_t \coloneqq Q_t@[y{\leftarrow}i^+] \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ via rule $\mathsf{M_{GC}}$.
    - Let $x \in \mathsf{u}(i^+)$. By Lemma 13.5.10.2 (Focusing inert terms on unapplied variables) gives the existence of a term context $\mathcal{H}_x$ such that $\mathcal{H}_x\langle x\rangle = i^+$, with $x \notin \mathsf{u}(\mathcal{H}_x) \subset \mathsf{u}(i^+)$ and $\mathsf{a}(\mathcal{H}_x) \subseteq \mathsf{a}(i^+)$. Case analysis on whether $y \in \mathcal{V}_t$ and on whether $y \in \mathcal{B}_t$:
      - Let $y \notin (\mathcal{V}_t \cup \mathcal{B}_t)$. Then we can derive $P_t \coloneqq Q_t@[y{\leftarrow}i^+] \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ via rule $\mathsf{M_{GC}}$.
      - Let $y \in \mathcal{V}_t$ and $y \notin \mathcal{B}_t$. By Lemma 13.5.12.3 (Focusing multiplicative evaluation context on unapplied variables), there exists $R_x \in \mathcal{E}_{\mathcal{W}_x,\mathcal{C}_x}$ such that $R_x\langle y\rangle = Q_t\langle x\rangle = Q\langle t\rangle$, with $y \notin \mathcal{W}_x \subset \mathcal{V}_t \subseteq \mathcal{V}$ and $\mathcal{C}_x \subseteq \mathcal{B}_t \subseteq \mathcal{B}$. We can finally derive $P_t \coloneqq R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\mathcal{W}_x\cup \mathsf{u}(\mathcal{H}_x),\mathcal{C}_x\cup \mathsf{a}(\mathcal{H}_x)}$ via rule $\mathsf{M_{HER}}$.
      - Let $y \in \mathcal{B}_t$. By *i.h.* (Focusing multiplicative evaluation contexts on applied variables), there exists $R_x \in \mathcal{E}^{@}_{\mathcal{W}_x,\mathcal{C}_x}$ such that $R_x\langle y\rangle = Q_t\langle y\rangle = Q\langle t\rangle$, with $\mathcal{W}_x \subseteq \mathcal{V}_t \subset \mathcal{V}$ and $y \notin \mathcal{C}_x \subset \mathcal{B}_t \subseteq \mathcal{B}$. Moreover, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $R_x \in \mathcal{E}_{\tilde{\mathcal{W}}_x,\tilde{\mathcal{C}}_x}$ for some $\tilde{\mathcal{W}}_x \subseteq \mathcal{W}_x$ and $\tilde{\mathcal{C}}_x \subseteq \mathcal{C}_x$.
        Finally, if $y \notin \tilde{\mathcal{W}}_x$, then we can derive $P_t \coloneqq R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\tilde{\mathcal{W}}_x\cup \mathsf{u}(\mathcal{H}_x),\tilde{\mathcal{C}}_x\cup \mathsf{a}(\mathcal{H}_x)}$ via rule $\mathsf{M_{HER}}$. If $y \in \tilde{\mathcal{W}}_x$ instead, then Lemma 13.5.12.3 (Focusing multiplicative evaluation contexts on unapplied variables) gives $S_y \in \mathcal{E}_{\mathcal{X}_y,\mathcal{D}_y}$ such that $S_y\langle y\rangle = R_x\langle x\rangle = Q_t\langle x\rangle = Q\langle t\rangle$, with $y \notin \mathcal{X}_y \subset \tilde{\mathcal{W}}_x \subseteq \mathcal{W}_x \subseteq \mathcal{V}_t \subset \mathcal{V}$ and $\mathcal{D}_y \subseteq \tilde{\mathcal{C}}_x \subset \mathcal{B}_t \subseteq \mathcal{V}$; we can then derive $P_t \coloneqq S_y\langle y\rangle@[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\mathcal{X}_y\cup \mathsf{u}(\mathcal{H}_x),\mathcal{D}_y\cup \mathsf{a}(\mathcal{H}_x)}$ via rule $\mathsf{M_{HER}}$.
  - Let $x \notin \mathcal{V}$. Then $x \in \mathsf{u}(i^+)$. Note that Lemma 13.5.10.1 (Focusing inert terms on unapplied variables) gives the existence of term context $\mathcal{H}_x$ such that $\mathcal{H}_x\langle x\rangle = i^+$, with $x \notin \mathsf{u}(\mathcal{H}_x) \subset \mathsf{u}(i^+)$ and $\mathsf{a}(\mathcal{H}_x) \subseteq \mathsf{a}(i^+)$. Case analysis on whether $y \in \mathcal{V}$ and on whether $y \in \mathcal{B}$:
    - Let $y \in \mathcal{V}$ and $y \notin \mathcal{B}$. By *i.h.* (Focusing exponential evaluation contexts on unapplied variables), there exists $Q_t \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $y \notin \mathcal{V}_t \subset \mathcal{V}$ and $\mathcal{B}_t \subseteq \mathcal{B}$. We can then derive $P_t \coloneqq Q_t\langle y\rangle@[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\mathcal{V}_t\cup \mathsf{u}(\mathcal{H}_x),\mathcal{B}\cup \mathsf{a}(\mathcal{H}_x)}$ via rule $\mathsf{M_{HER}}$.
    - Let $y \notin \mathcal{V}$ and $y \in \mathcal{B}$. By *i.h.* (Focusing exponential evaluation contexts on applied variables), there exists $Q_t \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle y\rangle = Q\langle t\rangle$, with $\mathcal{V}_t \subseteq \mathcal{V}$ and $y \notin \mathcal{B}_t \subset \mathcal{B}$. Moreover, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $Q_t \in \mathcal{E}_{\tilde{\mathcal{V}}_t,\tilde{\mathcal{B}}_t}$ for some $\tilde{\mathcal{V}}_t \subseteq \mathcal{V}_t$ and $\tilde{\mathcal{B}}_t \subseteq \mathcal{B}$. We can finally derive $P_t \coloneqq Q_t\langle y\rangle@[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\tilde{\mathcal{V}}_t\cup \mathsf{u}(\mathcal{H}_x),\tilde{\mathcal{B}}_t\cup \mathsf{a}(\mathcal{H}_x)}$ via rule $\mathsf{M_{HER}}$.
    - Let $y \in \mathcal{V}$ and $y \in \mathcal{B}$. By *i.h.* (Focusing exponential evaluation contexts on

applied variables), there exists $Q_t \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle y\rangle = Q\langle t\rangle$, with $\mathcal{V}_t \subseteq \mathcal{V}$ and $y \notin \mathcal{B}_t \subset \mathcal{B}$. Case analysis on whether $x \in \mathcal{V}_t$:
  - Let $y \in \mathcal{V}_t$. By *i.h.* (Focusing exponential evaluation contexts on unapplied variables), there exists $R_y \in \mathcal{E}_{\mathcal{W}_y,\mathcal{C}_y}$ such that $R_y\langle y\rangle = Q_t\langle y\rangle = Q\langle t\rangle$, with $y \notin \mathcal{W}_y \subset \mathcal{V}_t \subseteq \mathcal{V}$ and $\mathcal{C}_y \subseteq \mathcal{B}_t \subset \mathcal{B}$. We can then derive $P_t \coloneqq R_y\langle y\rangle @[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\mathcal{W}_y \cup \mathsf{u}(\mathcal{H}_x),\mathcal{C}_y \cup \mathsf{a}(\mathcal{H}_x)}$ via rule $\mathsf{M_{HER}}$.
  - Let $y \notin \mathcal{V}_t$. By Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $Q_t \in \mathcal{E}_{\tilde{\mathcal{V}}_t,\tilde{\mathcal{B}}_t}$ for some $\tilde{\mathcal{V}}_t \subseteq \mathcal{V}_t$ and $\tilde{\mathcal{B}}_t \subseteq \mathcal{B}$. We can then derive $P_t \coloneqq Q_t\langle y\rangle @[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\tilde{\mathcal{V}}_t \cup \mathsf{u}(\mathcal{H}_x),\tilde{\mathcal{B}}_t \cup \mathsf{a}(\mathcal{H}_x)}$.
- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{Q@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E_{GC}}$$

  with $P = Q@[y{\leftarrow}u]$. By *i.h.* (Focusing exponential evaluation contexts on unapplied variables), there exists $Q_t \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$, $x \notin \mathcal{U}_t \subset \mathcal{U}$ and $\mathcal{A}_t \subseteq \mathcal{A}$. We can then derive $P_t \coloneqq Q_t@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathcal{U}_t,\mathcal{A}_t}$ via rule $\mathsf{E_{GC}}$.
- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \setminus \mathcal{B})}{Q@[y{\leftarrow}v] \in \mathcal{E}^{@}_{\mathcal{V}\setminus\{y\},\mathcal{B}}}\ \mathsf{E_U}$$

  with $P = Q@[y{\leftarrow}v]$, $\mathcal{U} = \mathcal{V} \setminus \{y\}$ and $\mathcal{A} = \mathcal{B}$. By *i.h.* (Focusing exponential evaluation contexts on unapplied variables), there exists $Q_t \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$, with $x \notin \mathcal{V}_t \subset \mathcal{V}$ and $\mathcal{B}_t \subseteq \mathcal{B}$.
  We can then derive $P_t \coloneqq Q_t@[y{\leftarrow}v] \in \mathcal{E}_{\mathcal{V}_t\setminus\{y\},\mathcal{B}_t}$ (resp. $P_t \coloneqq Q_t@[y{\leftarrow}v] \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$) via rule $\mathsf{M_U}$ (resp. via rule $\mathsf{M_{GC}}$) if $y \in \mathcal{V}_t$ (resp. if $y \notin \mathcal{V}_t$).
- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \notin \mathcal{B}}{Q\langle y\rangle @[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{V}\setminus\{y\},\mathcal{B}}}\ \mathsf{E_{NL}}$$

  with $P = Q\langle y\rangle @[y{\leftarrow}\langle\cdot\rangle]$, $\mathcal{U} = \mathcal{V} \setminus \{y\}$ and $\mathcal{A} = \mathcal{B}$. By *i.h.* (Focusing exponential evaluation contexts on unapplied variables) on x and Q, there exists $Q_y \in \mathcal{E}_{\mathcal{V}_y,\mathcal{B}_y}$ such that $Q_y\langle x\rangle = Q\langle y\rangle$, with $x \notin \mathcal{V}_y \subset \mathcal{V}$ and $\mathcal{B}_y \subseteq \mathcal{B}$.
  If $y \notin \mathcal{V}_t$, then we can derive $P_t \coloneqq Q_y@[y{\leftarrow}t] \in \mathcal{E}^{@}_{\mathcal{V}_y,\mathcal{B}_t}$ via rule $\mathsf{M_{GC}}$.
  Else, we proceed by case analysis on the shape of t:
  - Let $t \in \mathsf{Var}$. Then we can derive $P_t \coloneqq Q_t@[y{\leftarrow}t] \in \mathcal{E}_{\mathsf{upd}(\mathcal{V}_t,y,t),\mathsf{upd}(\mathcal{B}_t,y,t)}$ via rule $\mathsf{M_{VAR}}$.
  - Let t be a useful inert term. Then we apply rule $\mathsf{M_I}$ to derive

$$P_t \coloneqq Q_t@[y{\leftarrow}t] \in \mathcal{E}_{(\mathcal{V}_t\setminus\{y\})\cup\mathsf{u}(t),(\mathcal{B}_t\setminus\{y\})\cup\mathsf{a}(t)}$$

  - Let $t \in \mathsf{Val}$. Then we can derive $P_t \coloneqq Q_t@[y{\leftarrow}t] \in \mathcal{E}_{\mathcal{V}_t\setminus\{y\}),\mathcal{B}_t}$ via rule $\mathsf{M_U}$.

3. *Focusing exponential evaluation contexts on applied variables*:
   Let $x \in \mathcal{A}$ and $t \in \Lambda$ be normal. We proceed by case analysis on the last derivation rule in $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$:

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{}{(\mathcal{H}^{@}, \epsilon) \in \mathcal{E}^{@}_{\mathsf{u}(\mathcal{H}^{@}),\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_1}$$

  with $P = (\mathcal{H}^{@}, \epsilon)$, $\mathcal{U} = \mathsf{u}(\mathcal{H}^{@})$ and $\mathcal{A} = \mathsf{a}(\mathcal{H}^{@})$. The statement is given by applying Lemma 13.5.11.3 (Focusing term contexts on applied variables) on $\mathcal{H}^{@}$, yielding an applicative term context $\mathcal{J}^{@}_t$ such that $\mathcal{J}^{@}_t\langle x\rangle = \mathcal{H}^{@}\langle t\rangle$, with $\mathsf{u}(\mathcal{J}^{@}_t) \subseteq \mathsf{u}(\mathcal{H}^{@})$ and $x \notin \mathsf{a}(\mathcal{J}^{@}_t) \subset \mathsf{a}(\mathcal{H}^{@})$, and finally taking $P_t \coloneqq (\mathcal{J}^{@}_t, \epsilon)$.
- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}] \in \mathcal{E}^{@}_{\mathcal{V}\cup\mathsf{u}(\mathcal{H}^{@}),\mathcal{B}\cup\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_2}$$

  with $P = Q\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}]$, $\mathcal{U} = \mathcal{V} \cup \mathsf{u}(\mathcal{H}^{@})$ and $\mathcal{A} = \mathcal{B} \cup \mathsf{a}(\mathcal{H}^{@})$. Case analyis on whether $x \in \mathcal{B}$:
  - Let $x \in \mathcal{B}$. By *i.h.* (Focusing multiplicative evaluation contexts on applied variables), there exists $Q_y \in \mathcal{E}^{@}_{\mathcal{V}_y,\mathcal{B}_y}$ such that $Q_y\langle x\rangle = Q\langle y\rangle$, with $\mathcal{V}_y \subseteq \mathcal{V}$ and $x \notin \mathcal{B} \subset \mathcal{B}_y$. We can then derive $P_t \coloneqq Q_y@[y{\leftarrow}t] \in \mathcal{E}^{@}_{\mathcal{V}_y,\mathcal{B}_y}$ via rule E_{GC}.
  - Let $x \notin \mathcal{B}$. Then it must be that $x \in \mathsf{a}(\mathcal{H}^{@})$. By Lemma 13.5.11.3 (Focusing term contexts on applied variables), there exists an applicative term context $\mathcal{H}^{@}_t$ such that $\mathcal{H}^{@}_t\langle x\rangle = \mathcal{H}^{@}\langle t\rangle$, with $\mathsf{u}(\mathcal{H}^{@}_t) \subseteq \mathsf{u}(\mathcal{H}^{@})$ and $x \notin \mathsf{a}(\mathcal{H}^{@}_t) \subset \mathsf{a}(\mathcal{H}^{@})$. We can then derive $P_t \coloneqq Q\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_t] \in \mathcal{E}^{@}_{\mathcal{V}\cup\mathsf{u}(\mathcal{H}^{@}_t),\mathcal{B}_y\cup\mathsf{a}(\mathcal{H}^{@}_t)}$ via rule $\mathsf{E}_{\mathsf{AX}_2}$.
- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}z] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{V},y,z),\mathsf{upd}(\mathcal{B},y,z)}}\ \mathsf{E}_{\mathsf{VAR}}$$

  with $P = Q@[y{\leftarrow}z]$, $\mathcal{U} = \mathsf{upd}(\mathcal{V}, y, z)$ and $\mathcal{A} = \mathsf{upd}(\mathcal{B}, y, z)$. We proceed by case analysis on whether $y \in \mathcal{B}$:
  - Let $y \notin \mathcal{B}$. Since then $\mathcal{B} = \mathsf{upd}(\mathcal{B}, y, z) \ni x$, then we can apply the *i.h.* (Focusing exponential evaluation contexts on applied variables) to obtain a $Q_t \in \mathcal{E}^{@}_{\mathcal{B}_t,\mathcal{V}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$, with $\mathcal{V}_t \subseteq \mathcal{V}$ and $x \notin \mathcal{B}_t \subset \mathcal{B}$. Then we can derive $P_t \coloneqq Q_t@[y{\leftarrow}z] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{V},y,z),\mathsf{upd}(\mathcal{B},y,z)}$ (resp., $P_t \coloneqq Q_t@[y{\leftarrow}z] \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$) via rule $\mathsf{E}_{\mathsf{VAR}}$ (resp., E_{GC}) if $y \in (\mathcal{B}_t \cup \mathcal{V}_t)$ (resp., $y \notin (\mathcal{B}_t \cup \mathcal{V}_t)$).
  - Let $y \in \mathcal{B}$. Note that then $\mathsf{upd}(\mathcal{B}, y, z) = (\mathcal{B} \setminus \{y\}) \cup \{z\}$. We do case analysis on whether $x \in \mathcal{B}$:
    - If $x \in (\mathcal{B} \setminus \{y\})$, then we can prove the statement similarly to how we proved it in the case above where $y \notin \mathcal{B}$.
    - If $x \notin \mathcal{B}$, then it must be that $x = z$. By *i.h.* (Focusing exponential evaluation contexts on applied variables) with respect to y and Q, there exists $Q_t \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle y\rangle = Q\langle t\rangle$, with $\mathcal{V}_t \subseteq \mathcal{V}$ and $y \notin \mathcal{B}_t \subset \mathcal{B}$. Moreover, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $Q_t \in \mathcal{E}_{\mathcal{W},\mathcal{C}}$ for some $\mathcal{W} \subseteq \mathcal{V}_t$ and $\mathcal{C} \subseteq \mathcal{B}_t$. We can then derive $P_t \coloneqq Q_t\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{W},\mathcal{C}}$ via rule $\mathsf{E}_{\mathsf{AX}_2}$.
- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}i^{+}] \in \mathcal{E}^{@}_{(\mathcal{V}\setminus\{y\})\cup\mathsf{u}(i^{+}),(\mathcal{B}\setminus\{y\})\cup\mathsf{a}(i^{+})}}\ \mathsf{E}_{\mathsf{I}}$$

with $P = Q@[y{\leftarrow}i^+]$, $\mathcal{U} = (\mathcal{V} \setminus \{y\}) \cup \mathsf{u}(i^+)$ and $\mathcal{A} = (\mathcal{B} \setminus \{y\}) \cup \mathsf{a}(i^+)$. Case analysis on whether $x \in \mathcal{B}$:

- Let $x \in \mathcal{B}$. By *i.h.* (Focusing exponential evaluation contexts on applied variables), there exists $Q_t \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$, with $\mathcal{V}_t \subseteq \mathcal{V}$ and $x \notin \mathcal{B}_t \subset \mathcal{B}$. Case analysis on whether $x \in \mathsf{a}(i^+)$:
  - Let $x \notin \mathsf{a}(i^+)$. We can then derive

    $$P_t \coloneqq Q_t@[y{\leftarrow}i^+] \in \mathcal{E}^{@}_{(\mathcal{V}_t\setminus\{y\})\cup\mathsf{u}(i^+),(\mathcal{B}_t\setminus\{y\})\cup\mathsf{a}(i^+)}$$

    (resp. $P_t \coloneqq Q_t@[y{\leftarrow}i^+] \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$) via rule $\mathsf{E_I}$ (resp. $\mathsf{E_{GC}}$) if $y \in (\mathcal{V}_t \cup \mathcal{B}_t)$ (resp. if $y \notin (\mathcal{V}_t \cup \mathcal{B}_t)$).
  - Let $x \in \mathsf{a}(i^+)$. Note that if $y \notin (\mathcal{V}_t \cup \mathcal{B}_t)$ then we can derive $P_t \coloneqq Q_t@[y{\leftarrow}i^+] \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$ via rule $\mathsf{E_{GC}}$. Let us now consider the case of $y \in (\mathcal{V}_t \cup \mathcal{B}_t)$, proceeding by case analysis:
    - Let $y \in \mathcal{V}_t$ and $y \notin \mathcal{B}_t$. By *i.h.* (Focusing exponential evaluation contexts on unapplied variables), there exists $R_x \in \mathcal{E}_{\mathcal{W}_x,\mathcal{C}_x}$ such that $R_x\langle y\rangle = Q_t\langle x\rangle$, with $y \notin \mathcal{W}_x \subset \mathcal{V}_t \subseteq \mathcal{V}$ and $\mathcal{C}_x \subseteq \mathcal{B}_t \subseteq \mathcal{B}$. We can then derive $P_t \coloneqq R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\mathcal{W}_x\cup\mathsf{u}(\mathcal{H}^{@}_x),\mathcal{C}_x\cup\mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E_{GC}}$.
    - Let $y \in \mathcal{B}_t$. By *i.h.* (Focusing exponential evaluation contexts on applied variables), there exists $R_x \in \mathcal{E}^{@}_{\mathcal{W}_x,\mathcal{C}_x}$ such that $R_x\langle y\rangle = Q_t\langle t\rangle = Q\langle t\rangle$, with $\mathcal{W}_x \subseteq \mathcal{V}_t \subseteq \mathcal{V}$ and $y \notin \mathcal{C}_x \subset \mathcal{B}_t \subset \mathcal{B}$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $R_x \in \mathcal{E}_{\tilde{\mathcal{W}}_x,\tilde{\mathcal{C}}_x}$ for some $\tilde{\mathcal{W}}_x \subseteq \mathcal{W}_x$ and $\tilde{\mathcal{C}}_x \subseteq \mathcal{C}_x$.
      Finally, if $y \notin \tilde{\mathcal{W}}_x$, then we can derive $P_t \coloneqq R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_x]$ via rule $\mathsf{E_{AX_2}}$. If $y \in \tilde{\mathcal{W}}_x$ instead, then Lemma 13.5.12.3 (Focusing multiplicative evaluation contexts on unapplied variables) gives $S_y \in \mathcal{E}_{\mathcal{X}_y,\mathcal{D}_y}$ such that $S_y\langle y\rangle = R_x\langle y\rangle = Q_t\langle x\rangle = Q\langle t\rangle$, with $y \notin \mathcal{X}_y \subset \mathcal{W}_x \subseteq \mathcal{V}_t \subseteq \mathcal{V}$ and $\mathcal{D}_y \subseteq \mathcal{C}_x \subset \mathcal{B}_t \subset \mathcal{B}$; we can then derive $P_t \coloneqq S_y\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\mathcal{X}_y\cup\mathsf{u}(\mathcal{H}^{@}_x),\mathcal{D}_y\cup\mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E_{AX_2}}$.
- Let $x \notin \mathcal{B}$. Then $x \in \mathsf{a}(i^+)$. By Lemma 13.5.11.2 (Focusing useful inert terms on applied variables), there exists applicative term context $\mathcal{H}^{@}_x$ such that $\mathcal{H}^{@}_x\langle x\rangle = i^+$, with $\mathsf{u}(\mathcal{H}^{@}_x) \subseteq \mathsf{u}(i^+)$ and $x \notin \mathsf{a}(\mathcal{H}^{@}_x) \subset \mathsf{a}(i^+)$. Case analysis on whether $y \in \mathcal{V}$ and on whether $y \in \mathcal{B}$:
  - Let $y \in \mathcal{V}$ and $y \notin \mathcal{B}$. By *i.h.* (Focusing exponential evaluation contexts on unapplied variables), there exists $Q_t \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle y\rangle = Q\langle t\rangle$, with $y \notin \mathcal{V}_t \subset \mathcal{V}$ and $\mathcal{B}_t \subseteq \mathcal{B}$. We can then derive $P_t \coloneqq Q_t\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\mathcal{V}_t\cup\mathsf{u}(\mathcal{H}^{@}_x),\mathcal{B}_t\cup\mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E_{AX_2}}$.
  - Let $y \notin \mathcal{V}$ and $y \in \mathcal{B}$. By *i.h.* (Focusing exponential evaluation contexts on applied variables), there exists $Q_t \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle y\rangle = Q\langle t\rangle$, with $\mathcal{V}_t \subseteq \mathcal{V}$ and $y \notin \mathcal{B}_t \subset \mathcal{B}$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $Q_t \in \mathcal{E}_{\tilde{\mathcal{V}}_t,\tilde{\mathcal{B}}_t}$, for some $\tilde{\mathcal{V}}_t \subseteq \mathcal{V}_t$ and $\tilde{\mathcal{B}}_t \subseteq \mathcal{B}_t$. We can then derive $P_t \coloneqq Q_t\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\tilde{\mathcal{V}}_t\cup\mathsf{u}(\mathcal{H}^{@}_x),\tilde{\mathcal{B}}_t\cup\mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E_{AX_2}}$.
  - Let $y \in \mathcal{V}$ and $y \in \mathcal{B}$. By *i.h.* (Focusing exponential evaluation contexts on unapplied variables), there exists $Q_t \in \mathcal{E}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle y\rangle = Q\langle t\rangle$, with $y \notin \mathcal{V}_t \subset \mathcal{V}$ and $\mathcal{B}_t \subseteq \mathcal{B}$. Case analysis on whether $y \in \mathcal{B}_t$:
    - If $y \notin \mathcal{B}_t$, then we can derived $P_t \coloneqq Q_t\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\mathcal{V}_t\cup\mathsf{u}(\mathcal{H}^{@}_x),\mathcal{B}_t\cup\mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E_{AX_2}}$.

· Let $y \in \mathcal{B}_t$. By *i.h.* (Focusing multiplicative evaluation contexts on applied variables), there exists $R_y \in \mathcal{E}^{@}_{\mathcal{W}_y,\mathcal{C}_y}$ such that $R_y\langle y\rangle = Q_t\langle y\rangle = Q\langle t\rangle$, with $\mathcal{W}_y \subseteq \mathcal{V}_t \subset \mathcal{V}$ and $y \notin \mathcal{C}_y \subset \mathcal{B}_t \subseteq \mathcal{B}$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $R_y \in \mathcal{E}_{\tilde{\mathcal{W}}_y,\tilde{\mathcal{C}}_y}$, for some $\tilde{\mathcal{W}}_y \subseteq \mathcal{V}_t$ and $\tilde{\mathcal{C}}_y \subseteq \mathcal{B}_t$. We can then derive $P \coloneqq R_y\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\tilde{\mathcal{W}}_y \cup \mathsf{u}(\mathcal{H}^{@}_x),\tilde{\mathcal{C}}_y \cup \mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E}_{\mathsf{AX}_2}$.

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{Q@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E}_{\mathsf{GC}}$$

  with $P = Q@[y{\leftarrow}u]$. By *i.h.* (Focusing exponential evaluation contexts on applied variables), there exists $Q_t \in \mathcal{E}^{@}_{\mathcal{U}_t,\mathcal{A}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$. We can then derive $P_t \coloneqq Q_t@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathcal{U}_t,\mathcal{A}_t}$ via rule E_{GC}.
- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \setminus \mathcal{B})}{Q@[y{\leftarrow}v] \in \mathcal{E}^{@}_{\mathcal{V}\setminus\{y\},\mathcal{B}}}\ \mathsf{E}_{\mathsf{U}}$$

  with $P = Q@[y{\leftarrow}v]$, $\mathcal{U} = \mathcal{V} \setminus \{y\}$ and $\mathcal{A} = \mathcal{B}$. By *i.h.* (Focusing exponential evaluation contexts on applied variables), there exists $Q_t \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$ such that $Q_t\langle x\rangle = Q\langle t\rangle$, with $\mathcal{V}_t \subseteq \mathcal{V}$ and $x \notin \mathcal{B}_t \subset \mathcal{B}$. We can then derive $P_t \coloneqq Q_t@[y{\leftarrow}v] \in \mathcal{E}^{@}_{\mathcal{V}_t\setminus\{y\},\mathcal{B}_t}$ (resp. $P_t \coloneqq Q_t@[y{\leftarrow}v] \in \mathcal{E}^{@}_{\mathcal{V}_t,\mathcal{B}_t}$) via rule E_{U} (resp. E_{GC}) if $y \in \mathcal{V}_t$ (resp. if $y \notin \mathcal{V}_t$).
- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as follows

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \notin \mathcal{B}}{Q\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{V}\setminus\{y\},\mathcal{B}}}\ \mathsf{E}_{\mathsf{NL}}$$

  with $P = Q\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle]$, $\mathcal{U} = \mathcal{V} \setminus \{y\}$ and $\mathcal{A} = \mathcal{B}$. By *i.h.* (Focusing exponential evaluation contexts on applied variables) on x and Q, there exists $Q_y \in \mathcal{E}^{@}_{\mathcal{V}_y,\mathcal{B}_y}$ such that $Q_y\langle x\rangle = Q\langle y\rangle$, with $\mathcal{V}_y \subseteq \mathcal{V}$ and $x \notin \mathcal{B}_y \subset \mathcal{B}$. Case analysis on whether $y \in \mathcal{V}_y$:
  - Let $y \in \mathcal{V}_y$. Case analysis on the shape of t:
    - Let $t \in \mathsf{Var}$. We can then use $\mathsf{E}_{\mathsf{VAR}}$ to derive

$$P \coloneqq Q_y@[y{\leftarrow}t] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{V}_y,y,t),\mathsf{upd}(\mathcal{B}_y,y,t)}$$

    - Let t be a useful inert term. We can then apply rule E_{I} to derive

$$P \coloneqq Q_y@[y{\leftarrow}t] \in \mathcal{E}^{@}_{(\mathcal{V}_y\setminus\{y\})\cup\mathsf{u}(t),(\mathcal{B}_y\setminus\{y\})\cup\mathsf{a}(t)}$$

    - Let $t \in \mathsf{Val}$. We can then derive $P \coloneqq Q_y@[y{\leftarrow}t] \in \mathcal{E}^{@}_{\mathcal{V}_y\setminus\{y\},\mathcal{B}_y}$ via rule E_{U}.
  - Let $y \notin \mathcal{V}_y$. We can then derive $P_t \coloneqq Q_y@[y{\leftarrow}t] \in \mathcal{E}^{@}_{\mathcal{V}_y,\mathcal{B}_y}$ via rule E_{GC}.

□

Lemma 13.5.14 (Focusing for Useful Open CbNeed-normal forms)**.**
Let $p \in \mathcal{PR}$ be in $\to_{\mathsf{und}}$-normal form.

1. Focusing Useful Open CbNeed-normal forms on unapplied variables*: Let* $x \in \mathsf{u}(p)$*. Then there exists* $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ *such that* $p = P\langle x\rangle$*, with* $x \notin \mathcal{U} \subset \mathsf{u}(p)$ *and* $\mathcal{A} \subseteq \mathsf{u}(p)$*.*
2. Focusing Useful Open CbNeed-normal forms on applied variables*: Let* $x \in \mathsf{a}(p)$*. Then there exists* $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ *such that* $p = P\langle x\rangle$*, with* $\mathcal{U} \subseteq \mathsf{u}(p)$ *and* $x \notin \mathcal{A} \subset \mathsf{u}(p)$*.*

Proof.

Both statements are proven simultaneously by mutual induction:

1. *Focusing* $\to_{\mathsf{und}}$*-normal forms on unapplied variables*: Let $p = (t, E)$ and $x \in \mathsf{u}(p)$. We proceed by induction on $|E|$:
   - Let $p = (t, \epsilon)$. Then $\mathsf{u}(t)$. Suppose t is not a normal term. Then there would exist term context $\mathcal{H}$ and $(\lambda z.s)m \in \Lambda$ such that $t = \mathcal{H}\langle(\lambda z.s)m\rangle$—by Lemma 8.2.2 (Redex in non-normal terms). But then we would be able to derive $(\mathcal{H}, \epsilon) \in \mathcal{E}_{\mathsf{u}(\mathcal{H}),\mathsf{a}(\mathcal{H})}$, getting that $p = (\mathcal{H}, \epsilon)\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} (\mathcal{H}, \epsilon)\langle s, [z{\leftarrow}m]\rangle$, which is absurd. Therefore, t is a normal term. Moreover, if $t \in \mathsf{Val}$ then $\mathsf{u}(t) = \emptyset$; absurd as well.
     Hence, t is an inert term. By Lemma 13.5.10.1 (Focusing inert terms on unapplied variables), there exists term context $\mathcal{H}_x$ such that $\mathcal{H}_x\langle x\rangle = t$, with $x \notin \mathsf{u}(\mathcal{H}_x) \subset \mathsf{u}(t)$ and $\mathsf{a}(\mathcal{H}_x) \subseteq \mathsf{a}(t)$. We can then derive $P \coloneqq (\mathcal{H}_x, \epsilon)$ via rule $\mathsf{M_{AX}}$.
   - Let $p = (t, E')@[y{\leftarrow}u]$. Case analysis on whether $x \in \mathsf{u}(t, E')$:
     - Let $x \in \mathsf{u}(t, E')$. By *i.h.* (Focusing $\to_{\mathsf{und}}$-normal forms on unapplied variables), there exists $Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}}$ such that $Q\langle x\rangle = (t, E')$, with $x \notin \mathcal{V} \subset \mathsf{u}(t, E')$ and $\mathcal{B} \subseteq \mathsf{a}(t, E')$.
       Note that if $y \notin (\mathcal{V} \cup \mathcal{B})$, then we can derive $P \coloneqq Q@[y{\leftarrow}u] \in \mathcal{E}_{\mathcal{V},\mathcal{B}}$ via rule $\mathsf{M_{GC}}$.
       Let us now consider the case where $y \in (\mathcal{V} \cup \mathcal{B})$, and proceed by case analysis on whether $y \in \mathcal{V}$.
       - Let $y \in \mathcal{V}$. By Lemma 13.5.12.3 (Focusing multiplicative evaluation contexts on unapplied variables), there exists $R_x \in \mathcal{E}_{\mathcal{W}_x,\mathcal{C}_x}$ such that $R_x\langle y\rangle = Q\langle x\rangle$, with $y \notin \mathcal{W}_x \subset \mathsf{u}(t, E')$ and $\mathcal{C}_x \subseteq \mathsf{a}(t, E')$.
         Now, suppose u were not a normal term. By Lemma 8.2.2 (Redex in non-normal terms), there would exist term context $\mathcal{H}$ and $(\lambda z.s)m \in \Lambda$ such that $u = \mathcal{H}\langle(\lambda z.s)m\rangle$. Note that if $u \notin \mathcal{C}_x$, then we would be able to derive $R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}]$ via rule $\mathsf{M_{GC}}$ and get that

$$\begin{array}{rcl} p & = & (t, E')@[y{\leftarrow}\mathcal{H}] \\ & = & (R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \\ & \to_{\mathsf{um}} & (R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle \end{array}$$

         which is absurd. If $u \in \mathcal{C}_x$ instead, then there would exist $S_y \in \mathcal{E}^{@}_{\mathcal{X}_y,\mathcal{D}_y}$ such that $S_y\langle y\rangle = R_x\langle y\rangle$, with $\mathcal{X}_y \subseteq \mathcal{W}_x$ and $y \notin \mathcal{D}_y \subset \mathcal{C}_x$—by Lemma 13.5.13.1 (Focusing multiplicative evaluation contexts on applied variables). In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) would give that $S_y \in \mathcal{E}_{\tilde{\mathcal{X}}_y,\tilde{\mathcal{D}}_y}$, for some $\tilde{\mathcal{X}}_y \subseteq \mathcal{X}_y$ and $\tilde{\mathcal{D}}_y \subseteq \mathcal{D}_y$. But then we would be able to derive $S_y@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\tilde{\mathcal{X}}_y \cup \mathsf{u}(\mathcal{H}),\tilde{\mathcal{D}}_y \cup \mathsf{a}(\mathcal{H})}$ and get that

$$\begin{array}{rcl} p & = & (t, E')@[y{\leftarrow}u] \\ & = & (R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \\ & \to_{\mathsf{um}} & (R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle \end{array}$$

         which is also absurd.
         Therefore, u must be a normal term. We proceed by case analysis on the shape of u as a normal term, and on whether $x \in \mathsf{u}(u)$:

· Let $x = u \in \mathsf{Var}$. If $y \notin \mathcal{C}_x$, then we can derive $P \coloneqq R_x\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}_{\mathcal{W}_x,\mathcal{C}_x}$ via rule $\mathsf{M_{HER}}$. If $y \in \mathcal{C}_x$ instead, then Lemma 13.5.13.1 (Focusing multiplicative evaluation contexts on applied variables) gives the existence of $S_y \in \mathcal{E}^{@}_{\mathcal{X}_y,\mathcal{D}_y}$ such that $S_y\langle y\rangle = R_x\langle y\rangle$, with $\mathcal{X}_y \subseteq \mathcal{W}_x$ and $y \notin \mathcal{D}_y \subset \mathcal{C}_x$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $S_y \in \mathcal{E}_{\tilde{\mathcal{X}}_y,\tilde{\mathcal{D}}_y}$, for some $\tilde{\mathcal{X}}_y \subseteq \mathcal{X}_y$ and $\tilde{\mathcal{D}}_y \subseteq \mathcal{D}_y$. We can then derive $P \coloneqq S_y\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}_{\tilde{\mathcal{X}}_y,\tilde{\mathcal{D}}_y}$ via rule $\mathsf{M_{HER}}$.

· Let $x \neq u \in \mathsf{Var}$. Then we can derive

$$P \coloneqq R_x\langle y\rangle@[y{\leftarrow}u] \in \mathcal{E}_{\mathsf{upd}(\mathcal{W}_x,y,u),\mathsf{upd}(\mathcal{C}_x,y,u)}$$

via rule $\mathsf{M_{VAR}}$.

· Let u be an inert term, with $x \in \mathsf{u}(u)$. By Lemma 13.5.10.1 (Focusing inert terms on unapplied variables), there exists term context $\mathcal{H}_x$ such that $\mathcal{H}_x\langle x\rangle = u$, with $x \notin \mathsf{u}(\mathcal{H}_x) \subset \mathsf{u}(u)$ and $\mathsf{a}(\mathcal{H}_x) \subseteq \mathsf{a}(u)$.
Now, if $u \notin \mathcal{C}_x$, then we can derive

$$P \coloneqq R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\mathcal{W}_x\cup\mathsf{u}(\mathcal{H}_x),\mathcal{C}_x\cup\mathsf{a}(\mathcal{H}_x)}$$

via rule $\mathsf{M_{HER}}$. If $y \in \mathcal{C}_x$ instead, then Lemma 13.5.13.1 (Focusing multiplicative evaluation contexts on applied variables) gives the existence of $S_y \in \mathcal{E}^{@}_{\mathcal{X}_y,\mathcal{D}_y}$ such that $S_y\langle y\rangle = R_x\langle y\rangle$, with $\mathcal{X}_y \subseteq \mathcal{W}_x$ and $y \notin \mathcal{D}_y \subset \mathcal{C}_x$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $S_y \in \mathcal{E}_{\tilde{\mathcal{X}}_y,\tilde{\mathcal{D}}_y}$, for some $\tilde{\mathcal{X}}_y \subseteq \mathcal{X}_y$ and $\tilde{\mathcal{D}}_y \subseteq \mathcal{D}_y$. We can then derive $P \coloneqq S_y\langle y\rangle@[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\tilde{\mathcal{X}}_y\cup\mathsf{u}(\mathcal{H}_x),\tilde{\mathcal{D}}_y\cup\mathsf{a}(\mathcal{H}_x)}$ via rule $\mathsf{M_{HER}}$.

· Let u be an inert term, with $x \notin \mathsf{u}(u)$. We can then derive $P \coloneqq Q@[y{\leftarrow}u] \in \mathcal{E}_{(\mathcal{V}\setminus\{y\})\cup\mathsf{u}(u),(\mathcal{B}\setminus\{y\})\cup\mathsf{a}(u)}$ via rule $\mathsf{M_I}$.

· Let $u \in \mathsf{Val}$. Suppose that $y \in \mathcal{C}_x$. By Lemma 13.5.13.1 (Focusing multiplicative evaluation contexts on applied variables), there would exist $S_y \in \mathcal{E}^{@}_{\mathcal{X}_y,\mathcal{D}_y}$ such that $S_y\langle y\rangle = R_x\langle y\rangle$, with $\mathcal{X}_y \subseteq \mathcal{W}_x$ and $y \notin \mathcal{D}_y \subset \mathcal{C}_x$. But then we would be able to derive $S_y\langle y\rangle@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathcal{X}_y,\mathcal{D}_y}$ via rule $\mathsf{M_{GC}}$ and get that

$$P = (t, E')@[y{\leftarrow}u] = (S_y@[y{\leftarrow}u])\langle y\rangle \to_{\mathsf{ue}} (S_y@[y{\leftarrow}u])\langle u^\alpha\rangle$$

which is absurd.
Therefore, $y \notin \mathcal{C}_x \subseteq \mathcal{B}$. Since $y \in \mathcal{V}$, then we can then derive $P \coloneqq Q@[y{\leftarrow}u] \in \mathcal{E}_{\mathcal{V},\mathcal{B}}$ via rule $\mathsf{M_U}$.

∗ Let $y \notin \mathcal{V}$. Then $y \in \mathcal{B}$. Following an analogous reasoning to the one above—where $y \in \mathcal{V}$—we can conclude that u must be a normal term. Similarly, it must be that $u \notin \mathsf{Val}$. The statement follows by proceeding by case analysis on the shape of u, and on whether $x \in \mathsf{u}(u)$, analogously to what we did in the case above—where $y \in \mathcal{V}$.

– Let $x \notin \mathsf{u}(t, E')$. Then $x \in \mathsf{u}(u)$, and—by Definition 37 (Unapplied variables)—it must be that either $y \in \mathsf{u}(t, E')$ or ($y \in \mathsf{nv}(t, E')$ and $u \notin \mathsf{Var}$)—note that these are not mutually exclusive proposition.
Moreover, following a similar reasoning to the case above—where $x \in \mathsf{u}(t, E')$—we can see that u must be an inert term. By Lemma 13.5.10.1 (Focusing inert terms on unapplied variables), there exists term context $\mathcal{H}_x$ such that $\mathcal{H}_x\langle x\rangle = u$, with $x \notin \mathsf{u}(\mathcal{H}_x) \subset \mathsf{u}(u)$ and $\mathsf{a}(\mathcal{H}_x) \subseteq \mathsf{a}(u)$.
Case analysis on whether $y \notin \mathsf{u}(t, E')$.

* Let $y \in \mathsf{u}(t, E')$. By *i.h.* (Focusing $\to_{\mathsf{und}}$-normal forms on unapplied varibles), there exists $Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}}$ such that $Q\langle y\rangle = (t, E')$, with $y \notin \mathcal{V} \subset \mathsf{u}(t, E')$ and $\mathcal{B} \subseteq \mathsf{a}(t, E')$.
  Now, if $y \notin \mathcal{B}$, then we can derive $P \coloneqq Q\langle y\rangle@[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\mathcal{V}\cup\mathsf{u}(\mathcal{H}_x),\mathcal{B}\cup\mathsf{a}(\mathcal{H}_x)}$ via rule $\mathsf{M_{HER}}$. If $y \in \mathcal{B}$ instead, then Lemma 13.5.13.1 (Focusing multiplicative evaluation contexts on applied variables), there exists $R_y \in \mathcal{E}^{@}_{\mathcal{W}_y,\mathcal{C}_y}$ such that $R_y\langle y\rangle = Q\langle y\rangle$, with $\mathcal{W}_y \subseteq \mathcal{V}$ and $\mathcal{C}_y \subseteq \mathcal{B}$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $R_y \in \mathcal{E}_{\tilde{\mathcal{W}}_y,\tilde{\mathcal{C}}_y}$, for some $\tilde{\mathcal{W}}_y \subseteq \mathcal{W}_y$ and $\tilde{\mathcal{C}}_y \subseteq \mathcal{C}_y$. We can finally derive $P \coloneqq R_y\langle y\rangle@[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\tilde{\mathcal{W}}_y\cup\mathsf{u}(\mathcal{H}_x),\tilde{\mathcal{C}}_y\cup\mathsf{a}(\mathcal{H}_x)}$ via rule $\mathsf{M_{HER}}$.
* Let $y \notin \mathsf{u}(t, E')$. Then $u \notin \mathsf{Var}$ and $y \in \mathsf{nv}(t, E')$, the latter implying that $y \in \mathsf{a}(t, E')$—by Lemma 8.1.1.2 (Unapplied, applied and needed variables). By *i.h.* (Focusing $\to_{\mathsf{und}}$-normal forms on applied variables), there exists $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $Q\langle y\rangle = (t, E')$, with $\mathcal{V} \subseteq \mathsf{u}(t, E')$ and $y \notin \mathcal{B} \subset \mathsf{a}(t, E')$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $Q \in \mathcal{E}_{\tilde{\mathcal{V}},\tilde{\mathcal{B}}}$, for some $\tilde{\mathcal{V}} \subseteq \mathcal{V}$ and $\tilde{\mathcal{B}} \subseteq \mathcal{B}$. Note that $y \notin \mathsf{u}(t, E') \supseteq \mathcal{V} \supseteq \tilde{\mathcal{V}}$. Hence, we can derive $P \coloneqq Q\langle y\rangle@[y{\leftarrow}\mathcal{H}_x] \in \mathcal{E}_{\tilde{\mathcal{V}}\cup\mathsf{u}(\mathcal{H}_x),\tilde{\mathcal{B}}\cup\mathsf{a}(\mathcal{H}_x)}$ via rule $\mathsf{M_{HER}}$.

2. *Focusing $\to_{\mathsf{und}}$-normal forms on applied variables*: Let $p = (t, E)$ and $x \in \mathsf{a}(p)$. We proceed by induction on $|E|$:
   - Let $p = (t, \epsilon)$. Note that if $t \in (\mathsf{Var} \cup \mathsf{Val})$ then $\mathsf{a}(P) = \emptyset$—absurd. Thus, if t is not a useful inert Λ-term, then it is not a normal term either and so Lemma 8.2.2 (Redex in non-normal terms) gives a term context $\mathcal{H}$ and $(\lambda z.s)m \in \Lambda$ such that $t = \mathcal{H}\langle(\lambda z.s)m\rangle$. Since all this implies that $p = (\mathcal{H}, \epsilon)\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} (\mathcal{H}, \epsilon)\langle s, [z{\leftarrow}m]\rangle$—which is absurd—we can conclude that t is a useful inert term. Therefore, by Lemma 13.5.11.2 (Focusing useful inert terms on applied variables) there exists an applicative term context $\mathcal{H}^{@}_x$ such that $t = \mathcal{H}_x\langle x\rangle$, with $\mathsf{u}(\mathcal{H}^{@}_x) \subseteq \mathsf{u}(t)$ and $x \notin \mathsf{a}(\mathcal{H}^{@}_x) \subset \mathsf{a}(t)$. The statement then follows by taking $P \coloneqq (\mathcal{H}^{@}_x, \epsilon) \in \mathcal{E}_{\mathsf{u}(\mathcal{H}^{@}_x),\mathsf{a}(\mathcal{H}^{@}_x)}$.
   - Let $p = (t, E')@[y{\leftarrow}u]$. Case analysis on whether $x \in \mathsf{a}(t, E')$:
     – Let $x \in \mathsf{a}(t, E')$. By *i.h.* (Focusing $\to_{\mathsf{und}}$-normal forms on applied variables), there exists $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $Q\langle t\rangle = (t, E')$, with $\mathcal{V} \subseteq \mathsf{u}(t, E')$ and $x \notin \mathcal{B} \subset \mathsf{a}(t, E')$.
     Note that if $y \notin (\mathcal{V} \cup \mathcal{B})$, then we can derive $P \coloneqq Q@[y{\leftarrow}u]$ via rule $\mathsf{E_{GC}}$. Let us assume now that $y \in (\mathcal{V} \cup \mathcal{B})$, and proceed by case analysis on whether $y \in \mathcal{B}$:
     * Let $y \in \mathcal{B}$. By Lemma 13.5.13.3 (Focusing exponential evaluation contexts on applied variables), there exists $R_x \in \mathcal{E}^{@}_{\mathcal{W}_x,\mathcal{C}_x}$ such that $R_x\langle y\rangle = Q\langle x\rangle$, with $\mathcal{W}_x \subseteq \mathcal{V}$ and $y \notin \mathcal{C}_x \subset \mathcal{B}$.
       Suppose now that u were not a normal term. By Lemma 8.2.2 (Redex in non-normal terms), there would exist term context $\mathcal{H}$ and $(\lambda z.s)m \in \Lambda$ such that $u = \mathcal{H}\langle(\lambda z.s)m\rangle$. Case analysis on whether $y \in \mathcal{W}_x$:
       · Let $y \notin \mathcal{W}_x$. Then note that $R_x \in \mathcal{E}_{\tilde{\mathcal{W}}_x,\tilde{\mathcal{C}}_x}$, for some $\tilde{\mathcal{W}}_x \subseteq \mathcal{W}_x$ and $\tilde{\mathcal{C}}_x \subseteq \mathcal{C}_x$—by Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative—and so we would be able to derive $R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\tilde{\mathcal{W}}_x\cup\mathsf{u}(\mathcal{H}),\tilde{\mathcal{C}}_x\cup\mathsf{a}(\mathcal{H})}$ via rule $\mathsf{E_{AX_2}}$ and get that

$$\begin{array}{rcl} p & = & (t, E')@[y{\leftarrow}u] \\ & = & (R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \\ & \to_{\mathsf{um}} & (R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle \end{array}$$

       which is absurd.

· Let $y \in \mathcal{W}_x$. By Lemma 13.5.13.2 (Focusing exponential evaluation contexts on unapplied variables), there exists $S_y \in \mathcal{E}_{\mathcal{X}_y,\mathcal{D}_x}$ such that $S_y\langle y\rangle = R_x\langle y\rangle$, with $y \notin \mathcal{X}_y \subset \mathcal{W}_x$ and $\mathcal{D}_y \subseteq \mathcal{C}_x$. But then we would be able to derive $S_y\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{X}_y\cup\mathsf{u}(\mathcal{H}),\mathcal{D}_y\cup\mathsf{a}(\mathcal{H})}$ via rule $\mathsf{E}_{\mathsf{AX}_2}$ and get that

$$\begin{array}{rcl} p & = & (t, E')@[y{\leftarrow}u] \\ & = & (S_y\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \\ & \to_{\mathsf{um}} & (S_y\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle \end{array}$$

which is absurd.

Therefore, u must be a normal term. Moreover, note that if $u \in \mathsf{Val}$, then we would be able to derive $R_x@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathcal{W}_x\setminus\{y\},\mathcal{B}}$ via rule E_{U} and get that

$$\begin{array}{rcl} p & = & (t, E')@[y{\leftarrow}u] \\ & = & (R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle y\rangle \\ & \to_{\mathsf{ue}} & (R_z\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle u^{\alpha}\rangle \end{array}$$

which is absurd as well. Therefore, u must be an inert term.

Finally, we proceed by case analysis on the shape of u:

· Let $u \in \mathsf{Var}$. If $u \neq x$, then we can derive $P \coloneqq Q@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{V},y,u),\mathsf{upd}(\mathcal{B},y,u)}$ via rule $\mathsf{E}_{\mathsf{VAR}}$. If $u = x$ instead, then we can derive $P \coloneqq R_x\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{W}_x\setminus\{y\},\mathcal{C}_x}$ via rule E_{NL}.

· Let $u \notin \mathsf{Var}$. Then t is a useful inert term. First, note if $x \notin \mathsf{a}(u)$, then we can simply derive $P \coloneqq Q@[y{\leftarrow}u] \in \mathcal{E}^{@}_{(\mathcal{V}\setminus\{y\})\cup\mathsf{u}(u),(\mathcal{B}\setminus\{y\})\cup\mathsf{a}(u)}$ via rule E_{I}.

Let us consider now the case where $x \in \mathsf{a}(u)$. Note that there exists applicative term context $\mathcal{H}^{@}_x$ such that $\mathcal{H}^{@}_x\langle x\rangle = u$, with $\mathsf{u}(\mathcal{H}^{@}_x) \subseteq \mathsf{u}(u)$ and $x \notin \mathsf{a}(\mathcal{H}^{@}_x) \subset \mathsf{a}(u)$—by Lemma 13.5.11.2 (Focusing useful inert terms on applied variables). Moreover, by Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative), it happens that $R_x \in \mathcal{E}_{\tilde{\mathcal{W}}_x,\tilde{\mathcal{C}}_x}$, for some $\tilde{\mathcal{W}}_x \subseteq \mathcal{X}_x$ and $\tilde{\mathcal{C}}_x \subseteq \mathcal{D}_x$.

Now, if $y \notin \tilde{\mathcal{W}}_x$, then we can simply apply rule $\mathsf{E}_{\mathsf{AX}_2}$ to derive

$$P \coloneqq R_x\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\mathcal{W}_x\cup\mathsf{u}(\mathcal{H}^{@}_x),\mathcal{C}_x\cup\mathsf{a}(\mathcal{H}^{@}_x)}$$

If $y \in \tilde{\mathcal{W}}_x$ instead, then Lemma 13.5.12.3 (Focusing multiplicative evaluation contexts on unapplied varibales) gives $S_y \in \mathcal{E}_{\mathcal{X}_y,\mathcal{D}_y}$ such that $S_y\langle y\rangle = R_x\langle x\rangle$, with $y \notin \mathcal{X}_y \subset \mathcal{W}_x$ and $\mathcal{D}_y \subseteq \mathcal{C}_x$; we can finally derive $P \coloneqq S_y\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\mathcal{X}_y\cup\mathsf{u}(\mathcal{H}^{@}_x),\mathcal{D}_y\cup\mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E}_{\mathsf{AX}_2}$.

∗ Let $y \notin \mathcal{B}$. Then $y \in \mathcal{V}$. By Lemma 13.5.13.2 (Focusing exponential evaluation contexts on unapplied variables), there exists $R_x \in \mathcal{E}_{\mathcal{W}_x,\mathcal{C}_x}$ such that $R_x\langle y\rangle = Q\langle x\rangle$, with $y \notin \mathcal{W}_x \subset \mathcal{V}$ and $\mathcal{C}_x \subseteq \mathcal{B}$.

We can now proceed by an analogous reasoning to the case above—where $y \in \mathcal{B}$—to obtain that u must be an inert term. Moreover, we can proceed by case analysis on the shape of u—following an analogous reasoning to the case above as well—to derive P.

– Let $x \notin \mathsf{a}(t, E')$. By Definition 36 (Applied variables), there are two (mutually exclusive) possibilities, namely $x = u \in \mathsf{Var}$ or $x \in \mathsf{a}(u)$:

* Let $x = u \in \mathsf{Var}$, with $y \in \mathsf{nv}(t, E')$ and $y \in \mathsf{a}(t, E')$—by Definition 36 (Applied variables). Hence, we can apply the *i.h.* (Focusing $\to_{\mathsf{und}}$-normal forms on applied variables) to get $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $Q\langle y\rangle = (t, E')$, with $\mathcal{V} \subseteq \mathsf{u}(t, E')$ and $y \notin \mathcal{B} \subset \mathsf{a}(t, E')$. We can finally derive $P \coloneqq Q\langle y\rangle @ [y \leftarrow \langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{V}\setminus\{y\},\mathcal{B}}$ via rule E_{NL}.
* Let $x \in \mathsf{a}(u)$, with $y \in \mathsf{nv}(t, E')$—by Definition 36 (Applied variables).
  Note that $x \notin (\mathsf{Var} \cup \mathsf{Val})$. We can infer that u must be a useful inert term, based on what is given by Lemma 8.2.2 (Redex in non-normal terms) and following an analogous reasoning to the ones given above. Therefore, there exists applicative term context $\mathcal{H}^{@}_x$ such that $\mathcal{H}^{@}_x\langle x\rangle = u$, with $\mathsf{u}(\mathcal{H}^{@}_x) \subseteq \mathsf{u}(u)$ and $x \notin \mathsf{a}(\mathcal{H}^{@}_x) \subset \mathsf{a}(u)$—by Lemma 13.5.11.2 (Focusing useful inert terms on applied variables).
  Moreover, note that $y \in \mathsf{nv}(t, E') = \mathsf{u}(t, E') \cup \mathsf{a}(t, E')$—by Lemma 8.1.1 (Unapplied, applied and needed variables). We proceed by case analysis on whether $y \in \mathsf{u}(t, E')$:
  · Let $y \in \mathsf{u}(t, E')$. By *i.h.* (Focusing $\to_{\mathsf{und}}$-normal forms on unapplied variables), there exists $Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}}$ such that $Q\langle y\rangle = (t, E')$, with $y \notin \mathcal{V} \subset \mathsf{u}(t, E')$ and $\mathcal{B} \subseteq \mathsf{a}(t, E')$.
    Now, if $y \notin \mathcal{B}$, then we can derive $P \coloneqq Q\langle y\rangle @ [y \leftarrow \mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\mathcal{V}\cup\mathsf{u}(\mathcal{H}^{@}_x),\mathcal{B}\cup\mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E}_{\mathsf{AX}_2}$. If $y \in \mathcal{B}$ instead, then there exists $R_y \in \mathcal{E}^{@}_{\mathcal{W}_y,\mathcal{C}_y}$ such that $R_y\langle y\rangle = Q\langle y\rangle$, with $\mathcal{W}_y \subseteq \mathcal{V}$ and $y \notin \mathcal{C}_y \subset \mathcal{B}$—by Lemma 13.5.13.1 (Focusing multiplicative evaluation contexts on applied variables); we can then apply Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) to get that $R_y \in \mathcal{E}_{\tilde{\mathcal{W}}_x,\tilde{\mathcal{C}}_x}$ for some $\tilde{\mathcal{W}}_x \subseteq \mathcal{W}_x$ and $\tilde{\mathcal{C}}_x \subseteq \mathcal{C}_x$, and finally derive $P \coloneqq R_y\langle y\rangle @ [y \leftarrow \mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\tilde{\mathcal{W}}_y\cup\mathsf{u}(\mathcal{H}^{@}_x),\tilde{\mathcal{C}}_y\cup\mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E}_{\mathsf{AX}_2}$.
  · Let $y \notin \mathsf{u}(t, E')$. Then it must be that $y \in \mathsf{a}(t, E')$. By *i.h.* (Focusing $\to_{\mathsf{und}}$-normal forms on applied variables), there exists $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $Q\langle y\rangle = (t, E')$, with $\mathcal{V} \subseteq \mathsf{u}(t, E')$ and $y \notin \mathcal{B} \subset \mathsf{a}(t, E')$. By Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative), $Q \in \mathcal{E}_{\tilde{\mathcal{V}},\tilde{\mathcal{B}}}$, for some $\tilde{\mathcal{V}} \subseteq \mathcal{V}$ and $\tilde{\mathcal{B}} \subseteq \mathcal{B}$.
    Now, if $y \notin \tilde{\mathcal{V}}$, then we can derive $P \coloneqq Q\langle y\rangle @ [y \leftarrow \mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\mathcal{V}\cup\mathsf{u}(\mathcal{H}^{@}_x),\mathcal{B}\cup\mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E}_{\mathsf{AX}_2}$. If $y \in \tilde{\mathcal{V}}$ instead, then Lemma 13.5.12.3 (Focusing multiplicative evaluation contexts on unapplied variables) gives $R_y \in \mathcal{E}_{\mathcal{W}_y,\mathcal{C}_y}$ such that $R_y\langle y\rangle = Q\langle y\rangle$, with $y \notin \mathcal{W}_y \subset \tilde{\mathcal{V}}$ and $\mathcal{C}_y \subseteq \tilde{\mathcal{B}}$; we can then derive $P \coloneqq R_y\langle y\rangle @ [y \leftarrow \mathcal{H}^{@}_x] \in \mathcal{E}^{@}_{\mathcal{W}_y\cup\mathsf{u}(\mathcal{H}^{@}_x),\mathcal{C}_y\cup\mathsf{a}(\mathcal{H}^{@}_x)}$ via rule $\mathsf{E}_{\mathsf{AX}_2}$.

□

Lemma 13.5.15 (Properties of Useful Open CbNeed-normal forms and ESs)**.**

1. Removing ESs does not create $\to_{\mathsf{und}}$-redexes*: if $(t, E[y \leftarrow u])$ is in $\to_{\mathsf{und}}$-normal form, then (t, E) is in $\to_{\mathsf{und}}$-normal form.*
2. Appending ESs that do not create $\to_{\mathsf{und}}$-redexes*: Let (t, E) be a $\to_{\mathsf{und}}$-normal form such that if $y \in \mathsf{nv}(t, E)$ then u is a normal term, and, moreover, if $y \in \mathsf{a}(t, E)$ then u is an inert Λ-term.*
   Then $(t, E[y \leftarrow u])$ is in $\to_{\mathsf{und}}$-normal form.

Proof.

1. *Removing ESs does not create $\to_{\mathsf{und}}$-redexes*: We prove the contrapositive statement:

$$\begin{array}{c}\text{If } (t,E) \text{ is not in } \to_{\mathsf{und}}\text{-normal form}\\ \text{then}\\ (t,E[y{\leftarrow}u]) \text{ is not in } \to_{\mathsf{und}}\text{-normal form}\end{array}$$

Case analysis on the kind of redex in (t,E):

- Let $(t,E) = P\langle(\lambda x.s)m\rangle \to_{\mathsf{um}} P\langle s,[x{\leftarrow}m]\rangle$, with $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$. Note that if $y \notin (\mathcal{U}\cup\mathcal{A})$, then we can derive $P@[y{\leftarrow}u] \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ via rule $\mathsf{M_{GC}}$ and get that

  $$(t,E[y{\leftarrow}u]) = (P@[y{\leftarrow}u])\langle(\lambda x.s)m\rangle \to_{\mathsf{um}} (P@[y{\leftarrow}u])\langle s,[x{\leftarrow}m]\rangle$$

  Let us now consider the case where $y \in (\mathcal{U}\cup\mathcal{A})$ and proceed by case analysis on the shape of u:
  - Let $u \in \mathsf{Var}$. We can then derive $P@[y{\leftarrow}u] \in \mathcal{E}_{\mathsf{upd}(\mathcal{U},y,u),\mathsf{upd}(\mathcal{A},y,u)}$ via rule $\mathsf{M_{VAR}}$ and get that

    $$(t,E[y{\leftarrow}u]) = (P@[y{\leftarrow}u])\langle(\lambda x.s)m\rangle \to_{\mathsf{um}} (P@[y{\leftarrow}u])\langle s,[x{\leftarrow}m]\rangle$$

  - Let u be a useful inert term. We can then derive $P@[y{\leftarrow}u] \in \mathcal{E}_{(\mathcal{U}\setminus\{y\})\cup\mathsf{u}(u),(\mathcal{A}\setminus\{y\})\cup\mathsf{a}(u)}$ via rule $\mathsf{M_I}$ and get that

    $$(t,E[y{\leftarrow}u]) = (P@[y{\leftarrow}u])\langle(\lambda x.s)m\rangle \to_{\mathsf{um}} (P@[y{\leftarrow}u])\langle s,[x{\leftarrow}m]\rangle$$

  - Let $u \in \mathsf{Val}$. Case analysis on whether $y \in \mathcal{A}$:
    - Let $y \in \mathcal{A}$. By Lemma 13.5.13.1 (Focusing multiplicative evaluation contexts on applied variables), there exists $Q_y \in \mathcal{E}^{@}_{\mathcal{V}_y,\mathcal{B}_y}$ such that $Q_y\langle y\rangle = P\langle(\lambda x.s)m\rangle$, with $\mathcal{V}_y \subseteq \mathcal{U}$ and $y \notin \mathcal{B}_y \subset \mathcal{A}$. Finally, case analysis on whether $y \in \mathcal{V}_y$
      - Let $y \in \mathcal{V}_y$. We can then derive $Q_y@[y{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}_y\setminus\{y\},\mathcal{B}}$ via rule $\mathsf{E_U}$ and get that

        $$(t,E[y{\leftarrow}u]) = (Q_y@[y{\leftarrow}u])\langle y\rangle \to_{\mathsf{ue}} (Q_y@[y{\leftarrow}u])\langle u^{\alpha}\rangle$$

      - Let $y \notin \mathcal{V}_y$. We can then derive $Q_y@[y{\leftarrow}u] \in \mathcal{E}_{\mathcal{V}_y,\mathcal{B}}$ via rule $\mathsf{E_{GC}}$ and get that

        $$(t,E[y{\leftarrow}u]) = (Q_y@[y{\leftarrow}u])\langle y\rangle \to_{\mathsf{ue}} (Q_y@[y{\leftarrow}u])\langle u^{\alpha}\rangle$$

    - Let $y \notin \mathcal{A}$. Then it must be that $y \in \mathcal{U}$. We can then derive $P@[y{\leftarrow}u] \in \mathcal{E}_{\mathcal{U}\setminus\{y\},\mathcal{A}}$ via rule $\mathsf{M_U}$ and get that

      $$(t,E[y{\leftarrow}u]) = (P@[y{\leftarrow}u])\langle(\lambda x.s)m\rangle \to_{\mathsf{um}} (P@[y{\leftarrow}u])\langle s,[x{\leftarrow}m]\rangle$$

  - Let u be a non-normal term. By Lemma 8.2.2 (Redex in non-normal terms), there exist term context $\mathcal{J}$ and $(\lambda\tilde{x}.\tilde{u})\tilde{s} \in \Lambda$ such that $u = \mathcal{H}\langle(\lambda\tilde{x}.\tilde{u})\tilde{s}\rangle$. Case analysis on whether $y \in \mathcal{U}$:
    - Let $y \in \mathcal{U}$. By Lemma 13.5.12.3 (Focusing multiplicative evaluation contexts on unapplied variables), there exists $Q_y \in \mathcal{E}_{\mathcal{V}_y,\mathcal{B}_y}$ such that $Q_y\langle y\rangle = P\langle(\lambda x.s)m\rangle$, with $y \notin \mathcal{V}_y \subset \mathcal{U}$ and $\mathcal{B}_y \subseteq \mathcal{A}$.
      Now, if $y \notin \mathcal{B}_y$, then we can derive $Q_y@[y{\leftarrow}\mathcal{J}] \in \mathcal{E}_{\mathcal{V}_y\cup\mathsf{u}(\mathcal{J}),\mathcal{B}_y\cup\mathsf{a}(\mathcal{J})}$ via rule $\mathsf{M_{HER}}$ and get that

      $$(t,E)@[y{\leftarrow}u] = (Q_y\langle y\rangle@[y{\leftarrow}\mathcal{J}])\langle(\lambda\tilde{x}.\tilde{u})\tilde{s}\rangle \to_{\mathsf{um}} (Q_y\langle y\rangle@[y{\leftarrow}\mathcal{J}])\langle\tilde{u},[\tilde{x}{\leftarrow}\tilde{s}]\rangle$$

Let $y \in \mathcal{B}_y$ instead. By Lemma 13.5.13.1 (Focusing multiplicative evaluation contexts on applied variables), there exists $R_y \in \mathcal{E}^{@}_{\mathcal{W}_y,\mathcal{C}_y}$ such that $R_y\langle y\rangle = Q_y\langle y\rangle$, with $\mathcal{W}_y \subseteq \mathcal{V}_y$ and $y \notin \mathcal{C}_y \subset \mathcal{B}_y$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $R_y \in \mathcal{E}_{\tilde{\mathcal{W}}_y,\tilde{\mathcal{C}}_y}$, for some $\tilde{\mathcal{W}}_y \subseteq \mathcal{W}_y$ and $\tilde{\mathcal{C}}_y \subseteq \mathcal{C}_y$. We can then derive $R_y\langle y\rangle@[y{\leftarrow}\mathcal{J}] \in \mathcal{E}_{\tilde{\mathcal{W}}_y \cup \mathsf{u}(\mathcal{J}),\tilde{\mathcal{C}}_y \cup \mathsf{a}(\mathcal{J})}$ such that

$$(t, E)@[y{\leftarrow}u] = (R_y\langle y\rangle@[y{\leftarrow}\mathcal{J}])\langle(\lambda\tilde{x}.\tilde{u})\tilde{s}\rangle \to_{\mathsf{um}} (R_y\langle y\rangle@[y{\leftarrow}\mathcal{J}])\langle\tilde{u}, [\tilde{x}{\leftarrow}\tilde{s}]\rangle$$

∗ Let $y \notin \mathcal{U}$. Then it must be that $y \in \mathcal{A}$. By Lemma 13.5.13.1 (Focusing multiplicative evaluation contexts on applied variables), there exists $Q_y \in \mathcal{E}^{@}_{\mathcal{V}_y,\mathcal{B}_y}$ such that $Q_y\langle y\rangle = P\langle(\lambda x.s)m\rangle$, with $\mathcal{V}_y \subseteq \mathcal{U}$ and $y \notin \mathcal{B}_y \subset \mathcal{A}$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $Q_y \in \mathcal{E}_{\tilde{\mathcal{V}}_y,\tilde{\mathcal{B}}_y}$, for some $\tilde{\mathcal{V}}_y \subseteq \mathcal{V}_y$ and $\tilde{\mathcal{B}}_y \subseteq \mathcal{B}_y$. We can then derive $Q_y\langle y\rangle@[y{\leftarrow}\mathcal{J}] \in \mathcal{E}_{\tilde{\mathcal{V}}_y \cup \mathsf{u}(\mathcal{J}),\tilde{\mathcal{B}}_y \cup \mathsf{a}(\mathcal{J})}$ such that

$$(t, E)@[y{\leftarrow}u] = (Q_y\langle y\rangle@[y{\leftarrow}\mathcal{J}])\langle(\lambda\tilde{x}.\tilde{u})\tilde{s}\rangle \to_{\mathsf{um}} (Q_y\langle y\rangle@[y{\leftarrow}\mathcal{J}])\langle\tilde{u}, [\tilde{x}{\leftarrow}\tilde{s}]\rangle$$

- Let $(t, E) = P\langle x\rangle \to_{\mathsf{ue}} P\langle s^{\alpha}\rangle$, with $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. Note then $x \neq y$—because $x \in \mathsf{dom}(t, E)$—and that if $y \notin (\mathcal{U} \cup \mathcal{A})$, then we can derive $P@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ via rule $\mathsf{E_{GC}}$ and get that

$$(t, E[y{\leftarrow}u]) = (P@[y{\leftarrow}u])\langle x\rangle \to_{\mathsf{um}} (P@[y{\leftarrow}u])\langle s^{\alpha}\rangle$$

Let us now consider the case where $y \in (\mathcal{U} \cup \mathcal{A})$ and proceed by case analysis on the shape of u:

– Let $u \in \mathsf{Var}$. We can then derive $P@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{U},y,u),\mathsf{upd}(\mathcal{A},y,u)}$ via rule $\mathsf{E_{VAR}}$ and get that

$$(t, E[y{\leftarrow}u]) = (P@[y{\leftarrow}u])\langle x\rangle \to_{\mathsf{um}} (P@[y{\leftarrow}u])\langle s^{\alpha}\rangle$$

– Let u be a useful inert term. We can then derive $P@[y{\leftarrow}u] \in \mathcal{E}^{@}_{(\mathcal{U}\setminus\{y\})\cup\mathsf{u}(u),(\mathcal{A}\setminus\{y\})\cup\mathsf{a}(u)}$ via rule $\mathsf{E_I}$ and get that

$$(t, E[y{\leftarrow}u]) = (P@[y{\leftarrow}u])\langle x\rangle \to_{\mathsf{um}} (P@[y{\leftarrow}u])\langle s^{\alpha}\rangle$$

– Let $u \in \mathsf{Val}$. If $x \notin \mathcal{A}$, we can then derive $P \coloneqq P@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ via rule $\mathsf{E_{GC}}$ and get that

$$(t, E[y{\leftarrow}u]) = (P@[y{\leftarrow}u])\langle x\rangle \to_{\mathsf{um}} (P@[y{\leftarrow}u])\langle s^{\alpha}\rangle$$

Let $x \in \mathcal{A}$ instead. By Lemma 13.5.13.3 (Focusing exponential evaluation contexts on applied variables), there exists $Q_y \in \mathcal{E}^{@}_{\mathcal{V}_y,\mathcal{B}_y}$ such that $Q_y\langle y\rangle = P\langle x\rangle$, with $\mathcal{V}_y \subseteq \mathcal{U}$ and $y \notin \mathcal{B}_y \subset \mathcal{A}$.
Now, if $y \notin \mathcal{V}_y$, then we can derive $Q_y@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathcal{V}_y,\mathcal{B}_y}$ via rule $\mathsf{E_{GC}}$ and get that

$$(t, E[y{\leftarrow}u]) = (Q_y@[y{\leftarrow}u])\langle y\rangle \to_{\mathsf{um}} (Q_y@[y{\leftarrow}u])\langle u^{\alpha}\rangle$$

Finally, if $y \in \mathcal{V}_y$ instead, then we can derive $Q_y@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathcal{V}_y\setminus\{y\},\mathcal{B}_y}$ via rule $\mathsf{E_U}$ and get that

$$(t, E[y{\leftarrow}u]) = (Q_y@[y{\leftarrow}u])\langle y\rangle \to_{\mathsf{um}} (Q_y@[y{\leftarrow}u])\langle u^{\alpha}\rangle$$

– Let u be a non-normal term. By Lemma 8.2.2 (Redex in non-normal terms), there exist term context $\mathcal{H}$ and $(\lambda z.s)m \in \Lambda$ such that $u = \mathcal{H}\langle(\lambda z.s)m\rangle$. We proceed by case analysis on whether $y \in \mathcal{U}$:

∗ Let $y \in \mathcal{U}$. By Lemma 13.5.13.2 (Focusing exponential evaluation contexts on unapplied variables), there exists $Q_y \in \mathcal{E}_{\mathcal{V}_y,\mathcal{B}_y}$ such that $Q_y\langle y\rangle = P\langle x\rangle$, with $y \notin \mathcal{V}_y \subset \mathcal{U}$ and $\mathcal{B}_u \subseteq \mathcal{A}$.
Now, if $y \notin \mathcal{B}_y$, then we can derive $Q_y\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}^{@}_{Q_y\langle y\rangle,[y\leftarrow\mathcal{H}]}$ via rule $\mathsf{M_{HER}}$ and get that

$$(t, E[y{\leftarrow}u]) = (Q_y\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} (Q_y\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle$$

Let $y \in \mathcal{B}_y$ instead. By Lemma 13.5.13.1 (Focusing multiplicative evaluation contexts on applied variables), there exists $R_y \in \mathcal{E}^{@}_{\mathcal{W}_y,\mathcal{C}_y}$ such that $R_y\langle y\rangle = Q_y\langle y\rangle$, with $\mathcal{W}_y \subseteq \mathcal{V}_y$ and $y \notin \mathcal{C}_y \subset \mathcal{B}_y$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $R_y \in \mathcal{E}_{\tilde{\mathcal{W}}_y,\tilde{\mathcal{C}}_y}$, for some $\tilde{\mathcal{W}}_y \subseteq \mathcal{W}_y$ and $\tilde{\mathcal{C}}_y \subseteq \mathcal{C}_y$. We can then derive $R_y\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\tilde{\mathcal{W}}_y\cup\mathsf{u}(\mathcal{H}),\tilde{\mathcal{C}}_y\cup\mathsf{a}(\mathcal{H})}$ via rule $\mathsf{M_{HER}}$ and get that

$$(t, E[y{\leftarrow}u]) = (Q_y\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} (Q_y\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle$$

∗ Let $y \notin \mathcal{U}$. Then it must be that $y \in \mathcal{A}$. By Lemma 13.5.13.3 (Focusing exponential evaluatioun contexts on applied variables), there exists $Q_y \in \mathcal{E}^{@}_{\mathcal{V}_y,\mathcal{B}_y}$ such that $Q_y\langle y\rangle = P\langle x\rangle$, with $\mathcal{V}_y \subseteq \mathcal{U}$ and $y \notin \mathcal{B}_y \subset \mathcal{A}$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $Q_y \in \mathcal{E}_{\tilde{\mathcal{V}}_y,\tilde{\mathcal{B}}_y}$, for some $\tilde{\mathcal{V}}_y \subseteq \mathcal{V}_y$ and $\tilde{\mathcal{B}}_y \subseteq \mathcal{B}_y$.
Now, if $y \notin \tilde{\mathcal{V}}_y$ then we can derive $Q_y\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\tilde{\mathcal{V}}_y\cup\mathsf{u}(\mathcal{H}),\tilde{\mathcal{B}}_y\cup\mathsf{a}(\mathcal{H})}$ via rule $\mathsf{M_{HER}}$ and get that

$$(t, E[y{\leftarrow}u]) = (Q_y\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} (Q_y\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle$$

Finally, let $y \in \tilde{\mathcal{V}}_y$ instead. By Lemma 13.5.12.3 (Focusing multiplicative evaluation contexts on unapplied variables), there exists $R_y \in \mathcal{E}_{\mathcal{W}_y,\mathcal{C}_y}$ such that $R_y\langle y\rangle = Q_y\langle y\rangle$, with $y \notin \mathcal{W}_y \subset \tilde{\mathcal{V}}_y$ and $\mathcal{C}_y \subseteq \tilde{\mathcal{B}}_y$. In addition, Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative) gives that $Q_y \in \mathcal{E}_{\tilde{\mathcal{W}}_y,\tilde{\mathcal{C}}_y}$, for some $\tilde{\mathcal{W}}_y \subseteq \mathcal{W}_y$ and $\tilde{\mathcal{C}}_y \subseteq \mathcal{C}_y$. We can derive $Q_y\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\tilde{\mathcal{W}}_y\cup\mathsf{u}(\mathcal{H}),\tilde{\mathcal{C}}_y\cup\mathsf{a}(\mathcal{H})}$ via rule $\mathsf{M_{HER}}$ and get that

$$(t, E[y{\leftarrow}u]) = (Q_y\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} (Q_y\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle s, [z{\leftarrow}m]\rangle$$

2. *Appending ESs that do not create $\to_{\mathsf{und}}$-redexes*: We prove the contrapositive statement:

Let u be such that if $y \in \mathsf{nv}(t, E)$ then u is a normal term and if $y \in \mathsf{a}(t, E)$ then u is an inert term. Moreover, let $(t, E[y{\leftarrow}u])$ not be in $\to_{\mathsf{und}}$-normal form
then
(t, E) is not in $\to_{\mathsf{und}}$-normal form

Case analysis on the kind of redex in $(t, E[y{\leftarrow}u])$:

- Let $(t, E[y{\leftarrow}u]) = P\langle(\lambda x.s)m\rangle$, with $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$. We proceed by case analysis on the last derivation rule in $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$:
  - *Rule* $\mathsf{M_{AX}}$: The case where $P = (\mathcal{H}, \epsilon)$ is impossible.

– *Rule* $\mathsf{M_{VAR}}$: Let

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y \leftarrow u] \in \mathcal{E}_{\mathsf{upd}(\mathcal{V},y,u),\mathsf{upd}(\mathcal{B},y,u)}}\ \mathsf{M_{VAR}}$$

where $P = Q@[y \leftarrow u]$, $u \in \mathsf{Var}$, $\mathcal{U} = \mathsf{upd}(\mathcal{V}, y, u)$ and $\mathcal{A} = \mathsf{upd}(\mathcal{B}, y, u)$. Then $(t, E) = Q\langle(\lambda x.s)m\rangle \to_{\mathsf{um}} Q\langle s, [x \leftarrow m]\rangle$.

– *Rule* $\mathsf{M_I}$: Let

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y \leftarrow u] \in \mathcal{E}_{(\mathcal{V}\setminus\{y\})\cup\mathsf{u}(u),(\mathcal{B}\setminus\{y\})\cup\mathsf{a}(u)}}\ \mathsf{M_I}$$

where $P = Q@[y \leftarrow u]$, u is a useful inert term, $\mathcal{U} = (\mathcal{V} \setminus \{y\}) \cup \mathsf{u}(u)$ and $\mathcal{A} = (\mathcal{B} \setminus \{y\}) \cup \mathsf{a}(u)$. Then $(t, E) = Q\langle(\lambda x.s)m\rangle \to_{\mathsf{um}} Q\langle s, [x \leftarrow m]\rangle$.

– *Rule* $\mathsf{M_{GC}}$: Let

$$\frac{Q \in \mathcal{E}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{Q@[y \leftarrow u] \in \mathcal{E}_{\mathcal{U},\mathcal{A}}}\ \mathsf{M_{GC}}$$

where $P = Q@[y \leftarrow u]$. Then $(t, E) = Q\langle(\lambda x.s)m\rangle \to_{\mathsf{um}} P\langle s, [x \leftarrow m]\rangle$.

– *Rule* $\mathsf{M_U}$: Let

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \setminus \mathcal{B})}{Q@[y \leftarrow u] \in \mathcal{E}_{\mathcal{V}\setminus\{y\},\mathcal{B}}}\ \mathsf{M_U}$$

where $P = Q@[y \leftarrow u]$, $u \in \mathsf{Val}$, $\mathcal{U} = \mathcal{V} \setminus \{y\}$ and $\mathcal{A} = \mathcal{B}$. Then $(t, E) = Q\langle(\lambda x.s)m\rangle \to_{\mathsf{um}} Q\langle s, [x \leftarrow m]\rangle$.

– *Rule* $\mathsf{M_{HER}}$: Suppose

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle y\rangle@[y \leftarrow \mathcal{H}] \in \mathcal{E}_{\mathcal{V}\cup\mathsf{u}(\mathcal{H}),\mathcal{B}\cup\mathsf{a}(\mathcal{H})}}\ \mathsf{M_{HER}}$$

with $P = Q\langle y\rangle@[y \leftarrow \mathcal{H}]$, $u = \mathcal{H}\langle(\lambda x.s)m\rangle$, $\mathcal{U} = \mathcal{V} \cup \mathsf{u}(\mathcal{H})$ and $\mathcal{A} = \mathcal{B} \cup \mathsf{a}(\mathcal{H})$. Since u is not normal—by Lemma 8.2.2 (Redex in non-normal terms)—then it would have to be that $y \notin \mathsf{nv}(t, E) = \mathsf{nv}(Q\langle y\rangle)$, by hypothesis. But Lemma 13.5.12.1 (Multiplicative evaluation contexts give needed variables) gives that $y \in \mathsf{nv}(Q\langle y\rangle)$. Hence, this case is impossible.

- Let $(t, E) = P\langle x\rangle$, with $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. We proceed by case analysis on the last derivation rule in $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$:
  – *Rule* $\mathsf{E_{AX_1}}$: The case where $P = (\mathcal{H}^{@}, \epsilon)$ is impossible.
  – *Rule* $\mathsf{E_{AX_2}}$: The case where $P = Q\langle y\rangle@[y \leftarrow \mathcal{H}^{@}]$, for some applicative term context $\mathcal{H}^{@}$, is impossible, as it would then be that $x \notin \mathsf{dom}(P)$; absurd.
  – *Rule* $\mathsf{E_{VAR}}$: Let

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y \leftarrow u] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{V},y,u),\mathsf{upd}(\mathcal{B},y,u)}}\ \mathsf{E_{VAR}}$$

where $P = Q@[y \leftarrow u]$, $u \in \mathsf{Var}$, $\mathcal{U} = \mathsf{upd}(\mathcal{V}, y, u)$ and $\mathcal{A} = \mathsf{upd}(\mathcal{B}, y, u)$. Note that $x \neq y$, because $u \notin \mathsf{Val}$. Hence, $x \in \mathsf{dom}(Q)$, and so we have that $(t, E) = Q\langle x\rangle \to_{\mathsf{ue}} Q\langle u^{\alpha}\rangle$.

  – *Rule* $\mathsf{E_I}$: Let

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y \leftarrow u] \in \mathcal{E}^{@}_{(\mathcal{V}\setminus\{y\})\cup\mathsf{u}(u),(\mathcal{B}\setminus\{y\})\cup\mathsf{a}(u)}}\ \mathsf{E_I}$$

where $P = Q@[y \leftarrow u]$, u is a useful inert term, $\mathcal{U} = (\mathcal{V} \setminus \{y\}) \cup \mathsf{u}(u)$ and $\mathcal{A} = (\mathcal{B} \setminus \{y\}) \cup \mathsf{a}(u)$. Note that $x \neq y$, because $u \notin \mathsf{Val}$. Hence, $x \in \mathsf{dom}(Q)$, and so we have that $(t, E) = Q\langle x \rangle \to_{\mathsf{ue}} Q\langle u^{\alpha} \rangle$.

– *Rule* $\mathsf{E_{GC}}$: Let

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{Q@[y \leftarrow u] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E_{GC}}$$

where $P = Q@[y \leftarrow u]$. Note that if $x \neq y$, then $x \in \mathsf{dom}(Q)$, giving that $(t, E) = Q\langle x \rangle \to_{\mathsf{ue}} Q\langle u^{\alpha} \rangle$. Suppose $x = y$ instead. Then it would have to be that $u \in \mathsf{Val}$ and that $(t, E) = Q\langle y \rangle$—by Definition 43 (Useful Open CbNeed evaluation strategy). But then Lemma 13.5.12.2 (Exponential evaluation contexts give applied variables) gives that $y \in \mathsf{a}(Q\langle y \rangle)$ and so it should be that u is an inert term, by hypothesis; absurd.

– *Rule* $\mathsf{E_U}$: Let

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \setminus \mathcal{B})}{Q@[y \leftarrow u] \in \mathcal{E}^{@}_{\mathcal{V} \setminus \{y\},\mathcal{B}}}\ \mathsf{E_U}$$

where $P = Q@[y \leftarrow u]$, $u \in \mathsf{Val}$, $\mathcal{U} = \mathcal{V} \setminus \{y\}$ and $\mathcal{A} = \mathcal{B}$. Note that if $x \neq y$, then $x \in \mathsf{dom}(Q)$, giving that $(t, E) = Q\langle x \rangle \to_{\mathsf{ue}} Q\langle u^{\alpha} \rangle$. Suppose $x = y$ instead. Then it would have to be that $u \in \mathsf{Val}$ and that $(t, E) = Q\langle y \rangle$. But then Lemma 13.5.12.2 (Exponential evaluation contexts give applied variables) gives that $y \in \mathsf{a}(Q\langle y \rangle)$ and so it should be that u is an inert term, by hypothesis; absurd.

– *Rule* $\mathsf{E_{NL}}$: The case where $P = Q\langle y \rangle@[y \leftarrow \langle \cdot \rangle]$ is impossible: it would then be that $x \notin \mathsf{dom}(P)$; absurd.

□

Proposition 13.5.16 (Syntactic characterization of Useful Open CbNeed-normal forms)**.** *Let $p \in \mathcal{PR}$. Then p is in $\to_{\mathsf{und}}$-normal form if and only if* $\mathsf{unorm}(p)$.

Proof. (Click here to go back to main chapter.)

⇒ : Let $p = (t, E)$ be in $\to_{\mathsf{und}}$-normal form. By induction on the length of E:

– Let $E = \epsilon$. Note that if $t \in \mathsf{Var}$ or $t \in \mathsf{Val}$ then $\mathsf{unorm}(t, \epsilon)$.

Let $t = t_1 t_2$. Case analysis on the shape of t_1:

∗ Suppose $t_1 \in \mathsf{Val}$, say $t_1 = \lambda x.u$. But then we would get that

$$p = (\langle \cdot \rangle, \epsilon)\langle (\lambda x.u) t_2 \rangle \to_{\mathsf{um}} (\langle \cdot \rangle, \epsilon)\langle u, [x \leftarrow t_2] \rangle$$

which is absurd.

∗ Let t_1 be an inert term. Case analysis on whether t_2 is a normal term:

· Suppose t_2 is not a normal term. By Lemma 8.2.2 (Redex in non-normal terms), there exist term context $\mathcal{H}$ and $(\lambda y.s)m \in \Lambda$ such that $t_2 = \mathcal{H}\langle (\lambda y.s)m \rangle$. But then $(t_1\mathcal{H}, \epsilon) \in \mathcal{E}_{\mathsf{u}(t_1\mathcal{H}),\mathsf{a}(t_1\mathcal{H})}$, and so

$$p = (t_1\mathcal{H}_1, \epsilon)\langle (\lambda y.s)m \rangle \to_{\mathsf{um}} (t_1\mathcal{H}_1, \epsilon)\langle s, [y \leftarrow m] \rangle$$

which is absurd.

· Let t_2 be a normal term. Then $t_1 t_2$ is a useful inert term, giving that

$$\dfrac{\dfrac{}{\mathsf{uinert}(t_1 t_2, \epsilon)}\,\mathsf{I_{AX}}}{\mathsf{unorm}(t_1 t_2, \epsilon)}\,\mathsf{unorm}_P$$

∗ Suppose t_1 is not a normal term. By Lemma 8.2.2 (Redex in non-normal terms), there exist term context $\mathcal{H}$ and $(\lambda y.s)m \in \Lambda$ such that $t_1 = \mathcal{H}\langle(\lambda y.s)m\rangle$. But then $(t_1\mathcal{H}, \epsilon) \in \mathcal{E}_{\mathsf{u}(t_1\mathcal{H}),\mathsf{a}(t_1\mathcal{H})}$, and so

$$p = (\mathcal{H}, \epsilon)\langle(\lambda y.s)m\rangle \to_{\mathsf{um}} (\mathcal{H}, \epsilon)\langle s, [y{\leftarrow}m]\rangle$$

which is absurd.

– Let $E = E'[y{\leftarrow}u]$—*i.e.*, $(t, E) = (t, E'[y{\leftarrow}u])$. By Lemma 13.5.15.1 (Removing ESs does not create $\to_{\mathsf{und}}$-redexes), (t, E') is in $\to_{\mathsf{und}}$-normal form. By *i.h.*, $\mathsf{unorm}(t, E')$. We proceed by case analysis on the predicate derivation of $\mathsf{unorm}(t, E)$:

∗ Let $\mathsf{genVar}_x(t, E')$. Case analysis on the last whether $y = x$:

· Let $y \neq x$. We can then derive

$$\dfrac{\dfrac{\mathsf{genVar}_x(t, E') \quad y \neq x}{\mathsf{genVar}_x(t, E'[y{\leftarrow}u])}\,\mathsf{GV_{GC}}}{\mathsf{unorm}(t, E'[y{\leftarrow}u])}\,\mathsf{unorm}_P$$

· Let $y = x$.
Now, if $u \in \mathsf{Var}$, then we can derive

$$\dfrac{\dfrac{\mathsf{genVar}_x(t, E')}{\mathsf{genVar}_u(t, E'[y{\leftarrow}u])}\,\mathsf{GV_{HER}}}{\mathsf{unorm}(u, E'[y{\leftarrow}u])}\,\mathsf{unorm}_P$$

If u is a useful inert term instead, then we can derive

$$\dfrac{\dfrac{\mathsf{genVar}_x(t, E')}{\mathsf{uinert}(u, t)E'[y{\leftarrow}u]}\,\mathsf{I_{GV}}}{\mathsf{unorm}(u, E'[y{\leftarrow}u])}\,\mathsf{unorm}_P$$

Next, if $u \in \mathsf{Val}$, then we can derive

$$\dfrac{\dfrac{\mathsf{genVar}_x(t, E')}{\mathsf{uinert}(u, t)E'[y{\leftarrow}u]}\,\mathsf{A_{GV}}}{\mathsf{unorm}(u, E'[y{\leftarrow}u])}\,\mathsf{unorm}_P$$

Finally, suppose u is not a normal term. By Lemma 8.2.2 (Redex in non-normal terms), there would exist term context $\mathcal{H}$ and $(\lambda y.s)m \in \Lambda$ such that $u = \mathcal{H}\langle(\lambda y.s)m\rangle$. In addition, note that $x \in \mathsf{nv}(t, E')$—by Lemma 8.2.4 (Properties of generalized variables). Thus, $x \in (\mathsf{u}(t, E') \cup \mathsf{a}(t, E'))$—by Lemma 8.1.1.2 (Unapplied, applied and needed variables). Case analysis on whether $x \in \mathsf{u}(t, E')$:
If $x \in \mathsf{u}(t, E')$, then there would exist $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ such that $P\langle x\rangle = (t, E')$, with $x \notin \mathcal{U} \subset \mathsf{u}(t, E')$ and $\mathcal{A} \subseteq \mathsf{a}(t, E')$. However, if $x \notin \mathcal{A}$, then we would be able to derive $P\langle x\rangle@[x{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{U}\cup\mathsf{u}(\mathcal{H}),\mathcal{A}\cup\mathsf{a}(\mathcal{H})}$ via rule $\mathsf{M_{HER}}$ and get that

$$(t, E'[y{\leftarrow}u]) = (P\langle x\rangle@[x{\leftarrow}\mathcal{H}])\langle(\lambda y.s)m\rangle \to_{\mathsf{um}} (P\langle x\rangle@[x{\leftarrow}\mathcal{H}])\langle s, [y{\leftarrow}m]\rangle$$

which is absurd. But if $x \in \mathcal{A}$ instead, then there would exist—by Lemma 13.5.13.1 (Focusing multiplicative evaluation contexts on applied variables)—$Q_x \in \mathcal{E}^{@}_{\mathcal{V}_x,\mathcal{B}_x}$ such that $Q_x\langle x\rangle = P\langle x\rangle$, with $\mathcal{V}_x \subseteq \mathcal{U}$ and $x \notin \mathcal{B}_x \subset \mathcal{A}$. In addition, $Q_x \in \mathcal{E}_{\tilde{\mathcal{V}}_x,\tilde{\mathcal{B}}_x}$ for some $\tilde{\mathcal{V}}_x \subseteq \mathcal{V}_x$ and $\tilde{\mathcal{B}}_x \subseteq \mathcal{B}_x$. However, we would then be able to derive $Q_x\langle x\rangle@[x{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\tilde{\mathcal{V}}_x \cup \mathsf{u}(\mathcal{H}),\tilde{\mathcal{B}}_x \cup \mathsf{a}(\mathcal{H})}$ via rule $\mathsf{M_{HER}}$ and get that

$$(t, E'[y{\leftarrow}u]) = (P\langle x\rangle@[x{\leftarrow}\mathcal{H}])\langle(\lambda y.s)m\rangle \rightarrow_{\mathsf{um}} (P\langle x\rangle@[x{\leftarrow}\mathcal{H}])\langle s, [y{\leftarrow}m]\rangle$$

which is also absurd.

Therefore, it would have to be that $x \notin \mathsf{u}(t, E')$ instead, forcing that $x \notin \mathsf{a}(t, E')$. Then, by Lemma 13.5.14.1 (Focusing Useful Open CbNeed-normal forms on applied variables), there would exist $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ such that $P\langle x\rangle = (t, E')$, with $\mathcal{U} \subseteq \mathsf{u}(t, E')$ and $x \notin \mathcal{A} \subset \mathsf{a}(t, E')$. In addition, $P \in \mathcal{E}_{\tilde{\mathcal{U}},\tilde{\mathcal{A}}}$, for some $\tilde{\mathcal{U}} \subseteq \mathcal{U}$ and $\tilde{\mathcal{A}} \subseteq \mathcal{A}$. However, we would then be able to derive $P\langle x\rangle@[x{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\tilde{\mathcal{U}} \cup \mathsf{u}(\mathcal{H}),\tilde{\mathcal{A}} \cup \mathsf{a}(\mathcal{H})}$ via rule $\mathsf{M_{HER}}$ and get that

$$(t, E'[y{\leftarrow}u]) = (P\langle x\rangle@[x{\leftarrow}\mathcal{H}])\langle(\lambda y.s)m\rangle \rightarrow_{\mathsf{um}} (P\langle x\rangle@[x{\leftarrow}\mathcal{H}])\langle s, [y{\leftarrow}m]\rangle$$

which is also absurd.

Therefore, the case where u is not a normal term is impossible.

∗ Let $\mathsf{uabs}(t, E')$. Note that $\mathsf{nv}(t, E') = \emptyset$—by Lemma 8.2.5 (Properties of useful abstraction programs)—and so we can derive $\mathsf{unorm}(t, E'[y{\leftarrow}u])$ as follows:

$$\dfrac{\dfrac{\mathsf{uabs}(t, E')}{\mathsf{uabs}(t, E'[y{\leftarrow}u])}\,\mathsf{A_{GC}}}{\mathsf{unorm}(t, E'[y{\leftarrow}u])}\,\mathsf{unorm}_P$$

∗ Let $\mathsf{uinert}(t, E')$. If $y \notin \mathsf{nv}(t, E')$ then we can derive $\mathsf{unorm}(t, E'[y{\leftarrow}u])$ as follows:

$$\dfrac{\dfrac{\mathsf{uinert}(t, E') \quad y \notin \mathsf{nv}(t, E')}{\mathsf{uinert}(t, E'[y{\leftarrow}u])}\,\mathsf{I_{GC}}}{\mathsf{unorm}(t, E'[y{\leftarrow}u])}\,\mathsf{unorm}_P$$

Let $y \in \mathsf{nv}(t, E') = (\mathsf{u}(t, E') \cup \mathsf{a}(t, E'))$—the last equality given by Lemma 8.1.1.2 (Unapplied, applied and needed variables). We proceed by case analysis on the shape of u:

· First, note that if u is an inert term then we can simply derive $\mathsf{unorm}(t, E'[y{\leftarrow}u])$ as follows:

$$\dfrac{\dfrac{\mathsf{uinert}(t, E') \quad y \in \mathsf{nv}(t, E')}{\mathsf{uinert}(t, E'[y{\leftarrow}u])}\,\mathsf{I_I}}{\mathsf{unorm}(t, E'[y{\leftarrow}u])}\,\mathsf{unorm}_P$$

· Let $u \in \mathsf{Val}$. On the one hand, note that if $y \notin \mathsf{a}(t, E')$ then it must be that $y \in \mathsf{u}(t, E')$ and so we can derive $\mathsf{unorm}(t, E'[y{\leftarrow}u])$ as follows:

$$\dfrac{\dfrac{\mathsf{uinert}(t, E') \quad y \in \mathsf{ul}\,(t, E')}{\mathsf{uinert}(t, E'[y{\leftarrow}u])}\,\mathsf{I_U}}{\mathsf{unorm}(t, E'[y{\leftarrow}u])}\,\mathsf{unorm}_P$$

On the other hand, let us suppose that $y \in \mathsf{a}(t, E')$. Then there would exist—by Lemma 13.5.14.2 (Focusing Useful Open CbNeed-normal forms on applied variables)—$P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ such that $P\langle y\rangle = (t, E')$, with $\mathcal{U} \subseteq \mathsf{u}(t, E')$ and $y \notin \mathcal{A} \subset \mathsf{a}(t, E')$. However, this leads to a contradiction. To see why, note that if $y \notin \mathcal{U}$ then would be able to derive

$$\frac{P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{P@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E_{GC}}$$

and if $y \in \mathcal{U}$ then we would be able to derive

$$\frac{P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad y \in (\mathcal{U} \setminus \mathcal{A})}{P@[y{\leftarrow}u] \in \mathcal{E}^{@}_{\mathcal{U}\setminus\{y\},\mathcal{A}}}\ \mathsf{E_{GC}}$$

having in both cases that

$$(t, E') = (P@[y{\leftarrow}u])\langle y\rangle \to_{\mathsf{ue}} (P@[y{\leftarrow}u])\langle u^{\alpha}\rangle$$

which is absurd. Therefore, the case where $y \in \mathsf{a}(t, E')$ is impossible.

· Finally, suppose u is not a normal term. By Lemma 8.2.2 (Redex in non-normal terms), there would exist term context $\mathcal{H}$ and $(\lambda z.s)m \in \Lambda$ such that $u = \mathcal{H}\langle(\lambda x.s)m\rangle$. Moreover, note that $y \in \mathsf{nv}(t, E') = (\mathsf{u}(t, E') \cup \mathsf{a}(t, E'))$—the equality given by Lemma 8.1.1.2 (Unapplied, applied and needed variables). Case analysis on whether $y \in \mathsf{u}(t, E')$:
Let $y \in \mathsf{u}(t, E')$. By Lemma 13.5.14.1 (Focusing Useful Open CbNeed-normal forms on unapplied variables), there would exist $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ such that $P\langle x\rangle = (t, E')$, with $x \notin \mathcal{U} \subset \mathsf{u}(t, E')$ and $\mathcal{A} \subset \mathsf{a}(t, E')$. But if $x \notin \mathcal{A}$, then we would be able to derive $P\langle x\rangle@[x{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{U}\cup\mathsf{u}(\mathcal{H}),\mathcal{A}\cup\mathsf{a}(\mathcal{H})}$ via rule $\mathsf{M_{HER}}$ and get that

$$(t, E'[x{\leftarrow}u]) = (P\langle x\rangle@[x{\leftarrow}\mathcal{H}])\langle(\lambda x.s)m\rangle \to_{\mathsf{um}} (P\langle x\rangle@[x{\leftarrow}\mathcal{H}])\langle s, [x{\leftarrow}m]\rangle$$

which is absurd. Moreover, if $x \in \mathcal{A}$, then Lemma 13.5.13.1 (Focusing multiplicative evaluation contexts on applied variables) would give $Q_x \in \mathcal{E}^{@}_{\mathcal{V}_x,\mathcal{B}_x}$ such that $Q_x\langle x\rangle = P\langle x\rangle$, with $\mathcal{V}_x \subseteq \mathcal{U}$ and $x \notin \mathcal{B}_x \subset \mathcal{A}$; by Lemma 13.5.12.4 (Exponential evaluation contexts are multiplicative), $Q_x \in \mathcal{E}_{\tilde{\mathcal{V}}_x,\tilde{\mathcal{B}}_x}$, for some $\tilde{\mathcal{V}}_x \subseteq \mathcal{V}_x$ and $\tilde{\mathcal{B}}_x \subseteq \mathcal{B}_x$. However, we would then be able to derive $Q_x\langle x\rangle@[x{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\tilde{\mathcal{V}}_x\cup\mathsf{u}(\mathcal{H}),\tilde{\mathcal{B}}_x\cup\mathsf{a}(\mathcal{H})}$ via rule $\mathsf{M_{HER}}$ and get that

$$(t, E'[x{\leftarrow}u]) = (Q\langle x\rangle@[x{\leftarrow}\mathcal{H}])\langle(\lambda x.s)m\rangle \to_{\mathsf{um}} (Q\langle x\rangle@[x{\leftarrow}\mathcal{H}])\langle s, [x{\leftarrow}m]\rangle$$

which is absurd.
Let $y \notin \mathsf{u}(t, E')$ instead, forcing that $y \in \mathsf{a}(t, E')$. It is easy to see that this case also leads to a contradiction, wrongfully proving that $(t, E'[x{\leftarrow}u]) \to_{\mathsf{um}}$-reduces. Therefore, the case where u is not a normal term is impossible.

$\Leftarrow$: We proceed by case analysis on the derivation of $\mathsf{unorm}(p)$. We first prove the statement for $\mathsf{genVar}_{\#}(p)$, and then prove it for $\mathsf{uinert}(p)$ and $\mathsf{uabs}(p)$, both of which rely on the former:

– *Generalized variables*: We proceed by induction on the derivation of $\mathsf{genVar}_{\#}(p)$:

∗ *Rule* $\mathsf{GV_{AX}}$: If $p = (x, \epsilon)$ then the statement clearly holds.

∗ *Rule* $\mathsf{GV}_{\mathsf{HER}}$: Let $\mathsf{genVar}_{\#}(p)$ be derived as

$$\frac{\mathsf{genVar}_{y}(q)}{\mathsf{genVar}_{x}(q@[y{\leftarrow}x])}\ \mathsf{GV}_{\mathsf{HER}}$$

where $p = q@[y{\leftarrow}x]$. By *i.h.*, q is in $\rightarrow_{\mathsf{und}}$-normal form. Then,

· Suppose $p = P\langle(\lambda z.s)m\rangle \rightarrow_{\mathsf{um}} P\langle s, [x{\leftarrow}m]\rangle$, for some $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$. But then, since x cannot be rewritten as a multiplicative redex, it must be that $q = Q\langle(\lambda z.s)m\rangle$, with $P = Q@[y{\leftarrow}x]$; absurd, because q is in $\rightarrow_{\mathsf{um}}$-normal form. Hence, p is in $\rightarrow_{\mathsf{um}}$-normal form.

· Since neither $x \notin \mathsf{dom}(P)$ nor $x \in \mathsf{Val}$, then $p = P\langle z\rangle \rightarrow_{\mathsf{ue}} P\langle s^{\alpha}\rangle$, for some $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$, only if $P = Q@[y{\leftarrow}u]$, for some $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $x \in \mathsf{dom}(Q)$. Absurd, since then it would be that $q = Q\langle x\rangle \rightarrow_{\mathsf{ue}} Q\langle s^{\alpha}\rangle$. Hence, p is in $\rightarrow_{\mathsf{ue}}$-normal form.

We conclude that p is in $\rightarrow_{\mathsf{und}}$-normal form.

∗ *Rule* $\mathsf{GV}_{\mathsf{GC}}$: Let $\mathsf{genVar}_{\#}(p)$ be derived as

$$\frac{\mathsf{genVar}_{x}(q) \quad y \neq x}{\mathsf{genVar}_{x}(q@[y{\leftarrow}t])}\ \mathsf{GV}_{\mathsf{GC}}$$

where $p = q@[y{\leftarrow}t]$. By *i.h.*, q is in $\rightarrow_{\mathsf{und}}$-normal form. Then

· Suppose $p = P\langle(\lambda z.s)m\rangle \rightarrow_{\mathsf{um}} P\langle s, [z{\leftarrow}m]\rangle$, for some $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$. Note that if $P = Q@[y{\leftarrow}t]$, for some $Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}}$, then it would be that $q = Q\langle(\lambda z.s)m\rangle \rightarrow_{\mathsf{um}} Q\langle s, [z{\leftarrow}m]\rangle$; absurd. Hence, it should be that $p = (R\langle y\rangle@[y{\leftarrow}\mathcal{H}])\langle(\lambda z.s)m\rangle$ with $q = R\langle y\rangle$ and $t = \mathcal{H}\langle(\lambda z.s)m\rangle$. In addition, Lemma 13.5.12.1 (Multiplicative evaluation contexts give needed variables) would give that $y \in \mathsf{nv}(R\langle y\rangle) = \mathsf{nv}(q)$. However, Lemma 8.2.4 (Properties of generalized variables) gives that $\mathsf{nv}(q) = \{x\}$. Since $y \neq x$, then this case is impossible, and so p is in $\rightarrow_{\mathsf{um}}$-normal form.

· Suppose $p = P\langle z\rangle \rightarrow_{\mathsf{ue}} P\langle s^{\alpha}\rangle$ for some $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. Note that the variable occurrence to be substituted must appear within q; otherwise, $x \notin \mathsf{dom}(P)$ which is absurd. That is, it should be that $P = Q@[y{\leftarrow}t]$, for some $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $q = Q\langle z\rangle$.

Now, let us suppose that $y = z$ and that $p = q@[y{\leftarrow}t] = (Q@[y{\leftarrow}t])\langle y\rangle \rightarrow_{\mathsf{ue}} (Q@[y{\leftarrow}t])\langle t^{\alpha}\rangle$. Then Lemma 13.5.12.2 (Exponential evaluation contexts give applied variables) would give that $y \in \mathsf{a}(Q\langle y\rangle) = \mathsf{a}(q)$. This is absurd, by Lemma 8.2.4 (Properties of generalized variables).

Hence, it should be that $y \neq z \in \mathsf{dom}(Q)$. However, this would in turn imply that $q = Q\langle z\rangle \rightarrow_{\mathsf{ue}} Q\langle s^{\alpha}\rangle$; absurd. Hence, p is in $\rightarrow_{\mathsf{ue}}$-normal form.

We conclude that p is in $\rightarrow_{\mathsf{und}}$-normal form.

– *Useful abstraction programs*: We proceed by induction on the derivation of $\mathsf{uabs}(p)$:

∗ *Rule* $\mathsf{A}_{\mathsf{Lift}}$: Let $\mathsf{uabs}(p)$ be derived as

$$\frac{}{\mathsf{uabs}(v, \epsilon)}\ \mathsf{A}_{\mathsf{Lift}}$$

where $p = (v, \epsilon)$. Note that $E = \epsilon$ discards the possibility that p were not in $\rightarrow_{\mathsf{ue}}$-normal form. Moreover, v discards the possibility that p were not in $\rightarrow_{\mathsf{um}}$—by Lemma 8.2.2 (Redex in non-normal terms). Hence, p is in $\rightarrow_{\mathsf{und}}$-normal form.

$*$ *Rule* $\mathsf{A_{GV}}$: Let $\mathsf{uabs}(p)$ be derived as

$$\frac{\mathsf{genVar}_x(q)}{\mathsf{uabs}(q@[x{\leftarrow}v])}\ \mathsf{A_{GV}}$$

where $p = q@[x{\leftarrow}v]$. By *i.h.*, q is in $\rightarrow_{\mathsf{und}}$-normal form. Then

- Suppose $p = P\langle(\lambda z.s)m\rangle \rightarrow_{\mathsf{um}} P\langle s, [z{\leftarrow}m]\rangle$, for some $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$.
  Note that if $P = Q@[x{\leftarrow}v]$, for some $Q \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$, then we would have that $q = Q\langle(\lambda z.s)m\rangle \rightarrow_{\mathsf{um}} Q\langle s, [z{\leftarrow}m]\rangle$; absurd.
  Hence, it should be that $P = Q\langle x\rangle@[x{\leftarrow}\mathcal{H}]$, with $q = Q\langle x\rangle$ and $v = \mathcal{H}\langle(\lambda z.s)m\rangle$. The latter is impossible, by Lemma 8.2.2 (Redex in non-normal terms). Hence, p is in $\rightarrow_{\mathsf{um}}$-normal form.
- Suppose $p = P\langle z\rangle \rightarrow_{\mathsf{ue}} P\langle s^\alpha\rangle$, for some $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. Note that the variable occurrence to be substituted must appear within q; otherwise, $x \notin \mathsf{dom}(P)$, which is absurd. That is, it should be that $P = Q@[y{\leftarrow}t]$, for some $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $q = Q\langle z\rangle$.
  Now, let us suppose that $y = z$ and that $p = q@[y{\leftarrow}t] = (Q@[y{\leftarrow}t])\langle y\rangle \rightarrow_{\mathsf{ue}} (Q@[y{\leftarrow}t])\langle t^\alpha\rangle$. Then Lemma 13.5.12.2 (Exponential evaluation contexts give applied variables) would give that $y \in \mathsf{a}(Q\langle y\rangle) = \mathsf{a}(q)$. This is absurd, by Lemma 8.2.4 (Properties of generalized variables).
  Hence, it should be that $y \neq z$ and that $z \in \mathsf{dom}(Q)$. However, this would in turn imply that $q = Q\langle z\rangle \rightarrow_{\mathsf{ue}} Q\langle s^\alpha\rangle$. This is absurd, and so p is in $\rightarrow_{\mathsf{ue}}$-normal form.

We conclude that p is in $\rightarrow_{\mathsf{und}}$-normal form.

$*$ *Rule* $\mathsf{A_{GC}}$: Let $\mathsf{uabs}(p)$ be derived as

$$\frac{\mathsf{uabs}(q)}{\mathsf{uabs}(q@[x{\leftarrow}t])}\ \mathsf{A_{GC}}$$

where $p = q@[x{\leftarrow}t]$. By *i.h.*, q is in $\rightarrow_{\mathsf{und}}$-normal form. Then

- Suppose $p = P\langle(\lambda z.s)m\rangle \rightarrow_{\mathsf{um}} P\langle s, [z{\leftarrow}m]\rangle$, for some $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$.
  Note that if $P = Q@[x{\leftarrow}v]$, for some $Q \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$, then we would have that $q = Q\langle(\lambda z.s)m\rangle \rightarrow_{\mathsf{um}} Q\langle s, [z{\leftarrow}m]\rangle$; absurd.
  Hence, it should be that $P = Q\langle x\rangle@[x{\leftarrow}\mathcal{H}]$, with $q = Q\langle x\rangle$ and $t = \mathcal{H}\langle(\lambda z.s)m\rangle$. However, by Lemma 13.5.12.1 (Multiplicative evaluation contexts give needed variables), we would have that $x \in \mathsf{nv}(Q\langle x\rangle) = \mathsf{nv}(q)$, which is absurd by Lemma 8.2.5 (Properties of useful abstraction programs). Hence, p is in $\rightarrow_{\mathsf{um}}$-normal form.
- Suppose $p = P\langle z\rangle \rightarrow_{\mathsf{ue}} P\langle s^\alpha\rangle$, for some $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. Note that the variable occurrence to be substituted must appear within q; otherwise, $x \notin \mathsf{dom}(P)$ which is absurd. That is, it should be that $P = Q@[x{\leftarrow}t]$, for some $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $q = Q\langle z\rangle$.
  Now, let us suppose that $x = z$ and that $p = q@[x{\leftarrow}t] = (Q@[x{\leftarrow}t])\langle z\rangle \rightarrow_{\mathsf{ue}} (Q@[x{\leftarrow}t])\langle t^\alpha\rangle$. Then Lemma 13.5.12.2 (Exponential evaluation contexts give applied variables) would give that $y \in \mathsf{a}(Q\langle y\rangle) = \mathsf{a}(q)$. This is absurd, by Lemma 8.2.5 (Properties of useful abstraction programs).
  Hence, it should be that $x \neq z \in \mathsf{dom}(Q)$. However, this would in turn imply that $q = Q\langle z\rangle \rightarrow_{\mathsf{ue}} Q\langle s^\alpha\rangle$. This is absurd, and so p is in $\rightarrow_{\mathsf{ue}}$-normal form.

We conclude that p is in $\to_{\mathsf{und}}$-normal form.

– *Useful inert programs*: We proceed by induction on the derivation of $\mathsf{uinert}(p)$:

  * *Rule* $\mathsf{I}_{\mathsf{Lift}}$: Let $\mathsf{uinert}(p)$ be derived as

$$\frac{}{\mathsf{uinert}(i^+, \epsilon)}\ \mathsf{I}_{\mathsf{Lift}}$$

    where $p = (i^+, \epsilon)$. Note that $E = \epsilon$ discards the possibility that p were not in $\to_{\mathsf{ue}}$-normal form. Moreover, i^+ discards the possibility that p were not in $\to_{\mathsf{um}}$-normal form—by Lemma 8.2.2 (Redex in non-normal terms). Hence, p is in $\to_{\mathsf{und}}$-normal form.

  * *Rule* I_{I}: Let $\mathsf{uinert}(p)$ be derived as

$$\frac{\mathsf{uinert}(q) \quad x \in \mathsf{nv}(i)}{\mathsf{uinert}(q@[x{\leftarrow}i])}\ \mathsf{I}_{\mathsf{I}}$$

    where $p = q@[x{\leftarrow}i]$. By *i.h.*, q is in $\to_{\mathsf{und}}$-normal form. Then,

    · Suppose $p = P\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} P\langle s, [z{\leftarrow}m]\rangle$, for some $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$.
    Note that if $P = Q@[x{\leftarrow}i]$, for some $Q \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$, then we would have that $q = Q\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} Q\langle s, [z{\leftarrow}m]\rangle$; absurd.
    Hence, it should be that $P = Q\langle x\rangle@[x{\leftarrow}\mathcal{H}]$, with $q = Q\langle x\rangle$ and $i = \mathcal{H}\langle(\lambda z.s)m\rangle$. However, this is also absurd, by Lemma 8.2.2 (Redex in non-normal terms). Hence, p is in $\to_{\mathsf{und}}$-normal form.

    · Suppose $p = P\langle z\rangle \to_{\mathsf{ue}} P\langle s^\alpha\rangle$, for some $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. Note that the variable occurrence to be substituted must appear within q; otherwise, $x \notin \mathsf{dom}(P)$ which is absurd. That is, it should be that $P = Q@[x{\leftarrow}i]$, for some $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $q = Q\langle z\rangle$. Moreover, note that since $i \notin \mathsf{Val}$, then it must be that $x \neq z$ and that $z \in \mathsf{dom}(Q)$. However, this would imply that $q = Q\langle z\rangle \to_{\mathsf{ue}} Q\langle s^\alpha\rangle$; absurd. Hence, p is in $\to_{\mathsf{ue}}$-normal form.

    We conclude that p is in $\to_{\mathsf{und}}$-normal form.

  * *Rule* I_{GV}: Let $\mathsf{uinert}(p)$ be derived as

$$\frac{\mathsf{genVar}_x(q)}{\mathsf{uinert}(q@[x{\leftarrow}i^+])}\ \mathsf{I}_{\mathsf{GV}}$$

    where $p = q@[x{\leftarrow}i^+]$. By *i.h.*, q is in $\to_{\mathsf{und}}$-normal form. Note that q is in $\to_{\mathsf{und}}$-normal form—as we proved it above for every generalized variable. Then,

    · Suppose $p = P\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} P\langle s, [z{\leftarrow}m]\rangle$, for some $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$.
    Note that if $P = Q@[x{\leftarrow}i]$, for some $Q \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$, then we would have that $q = Q\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} Q\langle s, [z{\leftarrow}m]\rangle$; absurd.
    Hence, it should be that $P = Q\langle x\rangle@[x{\leftarrow}\mathcal{H}]$, with $q = Q\langle x\rangle$ and $i = \mathcal{H}\langle(\lambda z.s)m\rangle$. However, this is also absurd, by Lemma 8.2.2 (Redex in non-normal terms). Hence, p is in $\to_{\mathsf{und}}$-normal form.

    · Suppose $p = P\langle z\rangle \to_{\mathsf{ue}} P\langle s^\alpha\rangle$, for some $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. Note that the variable occurrence to be substituted must appear within q; otherwise, $x \notin \mathsf{dom}(P)$ which is absurd. That is, it should be that $P = Q@[x{\leftarrow}i]$, for some $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $q = Q\langle z\rangle$ and $z \in \mathsf{dom}(Q)$—the latter because $i^+ \notin \mathsf{Val}$. However, Lemma 13.5.12.2 (Exponential evaluation contexts give applied variables) would give that $x \in \mathsf{a}(Q\langle x\rangle) = \mathsf{a}(q)$, which is absurd—by Lemma 8.2.4 (Properties of generalized variables). Hence, p is in $\to_{\mathsf{ue}}$-normal form.

We conclude that p is in $\to_{\mathsf{und}}$-normal form.

∗ *Rule* $\mathsf{I_U}$: Let $\mathsf{uinert}(p)$ be derived as

$$\frac{\mathsf{uinert}(q) \quad x \in \mathsf{u}(q) \quad x \notin \mathsf{a}(q)}{\mathsf{uinert}(q@[x{\leftarrow}v])}\ \mathsf{I_U}$$

where $p = q@[x{\leftarrow}v]$. By *i.h.*, q is in $\to_{\mathsf{und}}$-normal form. Then,

· Suppose $p = P\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} P\langle s, [z{\leftarrow}m]\rangle$, for some $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$.
Note that if $P = Q@[x{\leftarrow}v]$, for some $Q \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$, then we would have that $q = Q\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} Q\langle s, [z{\leftarrow}m]\rangle$; absurd.
Hence, it should be that $P = Q\langle x\rangle@[x{\leftarrow}\mathcal{H}]$, with $q = Q\langle x\rangle$ and $v = \mathcal{H}\langle(\lambda z.s)m\rangle$. However, this is also absurd, by Lemma 8.2.2 (Redex in non-normal terms). Hence, p is in $\to_{\mathsf{und}}$-normal form.

· Suppose $p = P\langle z\rangle \to_{\mathsf{ue}} P\langle s^{\alpha}\rangle$, for some $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. Note that the variable occurrence to be substituted must appear within q; otherwise, $x \notin \mathsf{dom}(P)$ which is absurd. That is, it should be that $P = Q@[x{\leftarrow}v]$, for some $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $q = Q\langle z\rangle$. But then Lemma 13.5.12.2 (Exponential evaluation contexts give applied variables) would give that $x \in \mathsf{a}(Q\langle x\rangle) = \mathsf{a}(q)$; absurd. Hence, p is in $\to_{\mathsf{ue}}$-normal form.

We conclude that p is in $\to_{\mathsf{und}}$-normal form.

∗ *Rule* $\mathsf{I_{GC}}$: Let $\mathsf{uinert}(p)$ be derived as

$$\frac{\mathsf{uinert}(q) \quad x \notin \mathsf{nv}(q)}{\mathsf{uinert}(q@[x{\leftarrow}t])}\ \mathsf{I_{GC}}$$

where $p = q@[x{\leftarrow}t]$. By *i.h.*, q is in $\to_{\mathsf{und}}$-normal form. Then,

· Suppose $p = P\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} P\langle s, [z{\leftarrow}m]\rangle$, for some $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$.
Note that if $P = Q@[x{\leftarrow}t]$, for some $Q \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$, then we would have that $q = Q\langle(\lambda z.s)m\rangle \to_{\mathsf{um}} Q\langle s, [z{\leftarrow}m]\rangle$; absurd.
Hence, it should be that $P = Q\langle x\rangle@[x{\leftarrow}\mathcal{H}]$, with $q = Q\langle x\rangle$ and $t = \mathcal{H}\langle(\lambda z.s)m\rangle$. However, Lemma 13.5.12.1 (Multiplicative evaluation contexts give needed variables) would give that $x \in \mathsf{nv}(Q\langle x\rangle) = \mathsf{nv}(q)$. Hence, p is in $\to_{\mathsf{und}}$-normal form.

· Suppose $p = P\langle z\rangle \to_{\mathsf{ue}} P\langle s^{\alpha}\rangle$, for some $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$. Note that the variable occurrence to be substituted must appear within q; otherwise, $x \notin \mathsf{dom}(P)$ which is absurd. That is, it should be that $P = Q@[x{\leftarrow}v]$, for some $Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}$ such that $q = Q\langle z\rangle$. But then Lemma 13.5.12.2 (Exponential evaluation contexts give applied variables) would give that $x \in \mathsf{a}(Q\langle x\rangle) = \mathsf{a}(q)$, with $\mathsf{a}(q) \subseteq \mathsf{nv}(q)$— by Lemma 8.1.1.2 (Unapplied, applied and needed variables); absurd. This is absurd, and so p is in $\to_{\mathsf{ue}}$-normal form.

We conclude that p is in $\to_{\mathsf{und}}$-normal form.

□

(Click here to go back to main chapter.)

13.5.3 Determinism.

Lemma 13.5.17 (Unique decomposition of Λ-terms)**.**

Let $\mathcal{H}_1\langle t_1\rangle = \mathcal{H}_2\langle t_2\rangle$, with $\mathcal{H}_1, \mathcal{H}_2$ term contexts, let $S \supseteq (\mathsf{a}(\mathcal{H}_1) \cup \mathsf{a}(\mathcal{H}_2))$, and let t_i be an S-reduction place of $\mathcal{H}_i\langle t_i\rangle$, for $i = 1, 2$.

Then $t_1 = t_2$ and $\mathcal{H}_1 = \mathcal{H}_2$.

Proof. (Click here to go back to main chapter.)

By induction on $\mathcal{H}_1$. Cases:

- *Empty*: $\mathcal{H}_1 = \langle\cdot\rangle$. If t_1 is a multiplicative redex then it must be that $\mathcal{H}_2 = \langle\cdot\rangle$ and $t_2 = t_1$. The same is true if t_1 is a variable not in S.
- *Left of an application*: $\mathcal{H}_1 = \mathcal{J}_1 u$. Cases of $\mathcal{H}_2$:
  - *Empty*: If $\mathcal{H}_2 = \langle\cdot\rangle$, then t_2 is a multiplicative redex, implying that $\mathcal{J}_1$ is empty and t_1 is a value, which is absurd. Therefore, this case is impossible.
  - *Left of an application*: Let $\mathcal{H}_2 = \mathcal{J}_2 u$. Then $\mathcal{J}_1\langle t_1\rangle = \mathcal{J}_2\langle t_2\rangle$ and t_i is a S-reduction places of $\mathcal{H}_i\langle t_i\rangle$ for $i = 1, 2$. By *i.h.*, $t_1 = t_2$ and $\mathcal{J}_1 = \mathcal{J}_2$, and then $\mathcal{H}_1 = \mathcal{H}_2$.
  - *Right of an application*: If $\mathcal{H}_2 = i\mathcal{J}_2$, then $i = \mathcal{J}_1\langle t_1\rangle$. Two cases:
    * If $i \in \mathsf{Var}$, then it should be that $\mathcal{J}_1 = \langle\cdot\rangle$ and $t_1 \in \mathsf{Var}$. But since $\mathcal{J}_1$ is not applicative, then t_1 cannot be an S-reduction place of $\mathcal{H}_1\langle t_1\rangle$, which contradicts the hypothesis.
    * Suppose $i = j^+$. Then it should be that $t_1 \in \mathsf{Var}$—by Lemma 8.2.2 (Redex in non-normal terms)—and so that $\mathcal{J}_1$ is applicative. However, this would imply that $t_1 \in \mathsf{a}(i)$—by Lemma 13.5.11.1 (Applicative term contexts give applied variables)—which gives that $t \in \mathsf{a}(\mathcal{H}_2) \subset S$; absurd.

    Therefore, this case is impossible.
- *Right of an application*: $\mathcal{H}_1 = i\mathcal{J}_1$. Cases of $\mathcal{H}_2$:
  - *Empty*: If $\mathcal{H}_2 = \langle\cdot\rangle$, then t_2 is an application that is not a multiplicative redex; absurd. Therefore, this case is impossible.
  - *Left of an application*: $\mathcal{H}_2 = \mathcal{J}_2 u$. This case is identical to the case where the hole of $\mathcal{H}_1$ is on the left of the application while the one of $\mathcal{H}_2$ is on the right, treated above.
  - *Right of an application*: $\mathcal{H}_2 = i\mathcal{J}_2$. Then $\mathcal{J}_1\langle t_1\rangle = \mathcal{J}_2\langle t_2\rangle$ and t_i is a S-reduction places of $\mathcal{H}_i\langle t_i\rangle$ for $i = 1, 2$. By *i.h.*, $t_1 = t_2$ and $\mathcal{J}_1 = \mathcal{J}_2$, and then $\mathcal{H}_1 = \mathcal{H}_2$.

□

(Click here to go back to main chapter.)

Lemma 13.5.18 (Unique decomposition of programs)**.**

Let $P_1\langle t_1\rangle = P_2\langle t_2\rangle$, with $P_1 \in (\mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1} \cup \mathcal{E}^{@}_{\mathcal{U}_1,\mathcal{A}_1})$ and $P_2 \in (\mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2} \cup \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2})$ such that $S \supseteq (\mathcal{U}_1 \cup \mathcal{A}_1 \cup \mathcal{U}_2 \cup \mathcal{A}_2)$, and t_i be an S-reduction place of $P_i\langle t_i\rangle$ for $i = 1, 2$.

Then $t_1 = t_2$ and $P_1 = P_2$.

Proof. (Click here to go back to main chapter.)

We split the analysis in two, namely when $P_1 \in \mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1}$ and $P_1 \in \mathcal{E}^{@}_{\mathcal{U}_1,\mathcal{A}_1}$:

- Let $P_1 \in \mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1}$. We further split the analysis in whether $P_2 \in \mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2}$ or $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$:
  - Let $P_2 \in \mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2}$. We proceed by induction on the derivation of $P_1 \in \mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1}$. Case analysis on the last derivation rule:
    * *Rule* MAX: Let P_1 be derived via an application of rule MAX. Then $P_1 \coloneqq (\mathcal{H}_1, \epsilon)$ for some term context $\mathcal{H}_1$ and so, in addition, it must be that $P_2 = (\mathcal{H}_2, \epsilon)$ for some term context $\mathcal{H}_2$—*i.e.*, $P_2 \in \mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived via an application of MAX. The statement follows by Lemma 8.3.1 (Unique decomposition of Λ-terms).

∗ *Rule* $\mathsf{M_{HER}}$: Let P_1 be derived as follows

$$\frac{Q_1 \in \mathcal{E}_{\mathcal{V}_1,\mathcal{B}_1} \quad x \notin (\mathcal{V}_1 \cup \mathcal{B}_1)}{Q_1\langle x\rangle @[x \leftarrow \mathcal{H}_1] \in \mathcal{E}_{\mathcal{V}_1 \cup \mathsf{u}(\mathcal{H}_1),\mathcal{B}_1 \cup \mathsf{a}(\mathcal{H}_1)}}\ \mathsf{M_{HER}}$$

where $P_1 = Q_1\langle x\rangle @[x \leftarrow \mathcal{H}_1]$, $\mathcal{U}_1 = \mathcal{V}_1 \cup \mathsf{u}(\mathcal{H}_1)$ and $\mathcal{A}_1 = \mathcal{B}_1 \cup \mathsf{a}(\mathcal{H}_1)$. There are two possibilities: either $t_1 \in \mathsf{Var}$ and $t_1 \notin (\mathcal{V}_1 \cup \mathcal{B}_1)$ or $t_1 = (\lambda x_1.u_1)s_1$—by Definition 45 (Reduction places in Useful Open CbNeed). We shall only cover the case of $t_1 = (\lambda x_1.u_1)s_1$, while leaving the other case for the reader.
As a starting point, note that x is a S-reduction place of $Q_1\langle x\rangle$, since $\mathcal{V}_1 \cup \mathcal{B}_1 \subseteq S$.
We now proceed by case analysis on the last derivation rule applied in $P_2 \in \mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2}$.

- The case of rule $\mathsf{M_{AX}}$ is impossible.
- The statement in the case of rule $\mathsf{M_{HER}}$ follows by *i.h.*.
- Suppose $P_2 \coloneqq Q_2@[x \leftarrow y] \in \mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived via an application of rule $\mathsf{M_{VAR}}$. But then we would have that $\mathcal{H}_1\langle t_1\rangle = \mathcal{H}_1\langle(\lambda x_1.u_1)s_1\rangle = y$; absurd by Lemma 8.2.2 (Redex in non-normal terms).
- The cases where P_2 is derived via an application of rule $\mathsf{M_I}$ or an application of rule $\mathsf{M_U}$ are ruled out analogously to the previous case, given that non-variable inerts and values are normal Λ-terms—just like y is in the previous case.
- Suppose $P_2 \in \mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived as follows

$$\frac{Q_2 \in \mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2} \quad x \notin (\mathcal{U}_2 \cup \mathcal{A}_2)}{Q_2@[x \leftarrow \tilde{s}] \in \mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2}}\ \mathsf{M_{GC}}$$

  with $P_2 = Q_2@[x \leftarrow \tilde{s}]$. Then there are two possibilities: either $t_2 = y \neq x$, or $t_2 = (\lambda x_2.u_2)s_2$—by Definition 45 (Reduction places in Useful Open CbNeed). On the one hand, if $t_2 = y$, then the *i.h.* on $Q_1\langle x\rangle = Q_2\langle y\rangle$ gives that $x = u = y$; absurd. On the other hand, if $t_2 = (\lambda x_2.u_2)s_2$, then the *i.h.* on $Q_1\langle x\rangle$ and $Q_2\langle(\lambda x_2.u_2)s_2\rangle$ with respect to $S \coloneqq (\mathcal{V}_1 \cup \mathcal{U}_2 \cup \mathcal{B}_1 \cup \mathcal{A}_2)$ to get that $x = t_1 = t_2 = (\lambda x_2.u_2)s_2$; absurd.
- In the case where P_2 is derived via an application of rule $\mathsf{M_{HER}}$, the statement follows by Lemma 8.3.1 (Unique decomposition of Λ-terms).

∗ *Rule* $\mathsf{M_{GC}}$: Let P_1 be derived as follows

$$\frac{Q_1 \in \mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1} \quad x \notin (\mathcal{U}_1 \cup \mathcal{A}_1)}{Q_1@[x \leftarrow \tilde{s}_1] \in \mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1}}\ \mathsf{M_{GC}}$$

where $P_1 = Q_1@[x \leftarrow \tilde{s}_1]$. Case analysis on the last derivation rule in $P_2 \in \mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2}$:

- The case of rule $\mathsf{M_{AX}}$ is impossible.
- The cases of rules $\mathsf{M_{GC}}$, $\mathsf{M_{VAR}}$, $\mathsf{M_I}$ and $\mathsf{M_U}$ are given by the *i.h.*.
- Suppose $P_2 \in \mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived as follows:

$$\frac{Q_2 \in \mathcal{E}_{\mathcal{V}_2,\mathcal{B}_2} \quad x \notin (\mathcal{V}_2 \cup \mathcal{B}_2)}{Q_2\langle x\rangle @[x \leftarrow \mathcal{H}_2] \in \mathcal{E}_{\mathcal{V}_2 \cup \mathsf{u}(\mathcal{H}_2),\mathcal{B}_2 \cup \mathsf{a}(\mathcal{H}_2)}}\ \mathsf{M_{HER}}$$

  with $P_2 = Q_2\langle x\rangle @[x \leftarrow \mathcal{H}_2]$, $\mathcal{U}_2 = \mathcal{V}_2 \cup \mathsf{u}(\mathcal{H}_2)$ and $\mathcal{A}_2 = \mathcal{B}_2 \cup \mathsf{a}(\mathcal{H}_2)$. But since t_1 is a S-reduction place of $Q_1\langle t_1\rangle$ and x is a S-reduction place of $Q_2\langle x\rangle$, we would then be able to apply the *i.h.* on $Q_1\langle t_1\rangle$ and $Q_2\langle x\rangle$ to get that that $t_1 = x$, which is absurd by Definition 45 (Reduction places in Useful Open CbNeed).

∗ *Rule* $\mathsf{M_{VAR}}$: Let P_1 be derived as follows

$$\frac{Q_1 \in \mathcal{E}_{\mathcal{V}_1,\mathcal{B}_1} \quad x \in (\mathcal{V}_1 \cup \mathcal{B}_1)}{Q_1@[x{\leftarrow}z] \in \mathcal{E}_{\mathsf{upd}(\mathcal{V}_1,x,z),\mathsf{upd}(\mathcal{B}_1,x,z)}}\ \mathsf{M_{VAR}}$$

where $P_1 = Q_1@[x{\leftarrow}z]$, $\mathcal{U}_1 = \mathsf{upd}(\mathcal{V}_1, x, z)$ and $\mathcal{A}_1 = \mathsf{upd}(\mathcal{B}_1, x, z)$. Case analysis on the last derivation rule in $P_2 \in \mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2}$:

· The case of rule $\mathsf{M_{AX}}$ is impossible.

· The case of rule $\mathsf{M_{GC}}$ was covered above, when P_1 was derived by an application of rule $\mathsf{M_{GC}}$ and P_2 was derived by an application of rule $\mathsf{M_{VAR}}$.

· The case of rule $\mathsf{M_{HER}}$ was covered above, when P_1 was derived by an application of rule $\mathsf{M_{HER}}$ and P_2 was derived by an application of rule $\mathsf{M_{VAR}}$.

· The case of rule $\mathsf{M_{VAR}}$ is given by a simple application of the *i.h.*.

· The case of rules $\mathsf{M_I}$ and $\mathsf{M_U}$ are impossible, because the set of variables is pairwise disjoint with the set of values and with the set of non-variale inert Λ-terms.

∗ *Rule* $\mathsf{M_I}$: This case can be analyzed analogously to the previous case.

∗ *Rule* $\mathsf{M_U}$: Same as previous item.

– Let $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$. We now proceed by induction on the derivation of $P_1 \in \mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1}$. Case analysis on the last derivation rule:

∗ *Rule* $\mathsf{M_{AX}}$: Let P_1 be derived via an application of rule $\mathsf{M_{AX}}$. Then $P_1 \coloneqq (\mathcal{H}_1, \epsilon)$ for some term context $\mathcal{H}_1$ and so, in addition, it must be that $P_2 = (\mathcal{H}_2^{@}, \epsilon)$ for some term context $\mathcal{H}_2$—*i.e.*, $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived via an application of rule $\mathsf{E_{AX_1}}$. The statement follows by Lemma 8.3.1 (Unique decomposition of Λ-terms).

∗ *Rule* $\mathsf{M_{HER}}$: Let P_1 be derived as follows

$$\frac{Q_1 \in \mathcal{E}_{\mathcal{V}_1,\mathcal{B}_1} \quad x \notin (\mathcal{V}_1 \cup \mathcal{B}_1)}{Q_1\langle x\rangle@[x{\leftarrow}\mathcal{H}_1] \in \mathcal{E}_{\mathcal{V}_1\cup\mathsf{u}(\mathcal{H}_1),\mathcal{B}_1\cup\mathsf{a}(\mathcal{H}_1)}}\ \mathsf{M_{HER}}$$

where $P_1 = Q_1\langle x\rangle@[x{\leftarrow}\mathcal{H}_1]$, $\mathcal{U}_1 = \mathcal{V}_1 \cup \mathsf{u}(\mathcal{H}_1)$ and $\mathcal{A}_1 = \mathcal{B}_1 \cup \mathsf{a}(\mathcal{H}_1)$. Note that x is a S-reduction place of $Q_1\langle x\rangle$. There are two possibilities: either $t_1 = (\lambda x_1.u_1)s_1$ or $t_1 \in \mathsf{Var}$—by Definition 45 (Reduction places in Useful Open CbNeed). We consider the former case, and leave the latter for the reader.

We now proceed by case analysis on the last derivation rule applied in $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$.

· The case of rule $\mathsf{E_{AX_1}}$ is impossible.

· Let $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ be derived as follows

$$\frac{Q_2 \in \mathcal{E}_{\mathcal{V}_2,\mathcal{B}_2} \quad x \notin (\mathcal{V}_2 \cup \mathcal{B}_2)}{Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}_2^{@}] \in \mathcal{E}^{@}_{(\mathcal{V}_2\setminus\{x\})\cup\mathsf{u}(\mathcal{H}_2^{@}),(\mathcal{B}_2\setminus\{x\})\cup\mathsf{a}(\mathcal{H}_2^{@})}}\ \mathsf{E_{AX_2}}$$

where $P_2 = Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}_2^{@}]$, $\mathcal{U}_2 = (\mathcal{V}_2 \setminus \{x\}) \cup \mathsf{u}(\mathcal{H}_2^{@})$ and $\mathcal{A}_2 = (\mathcal{B}_2 \setminus \{x\}) \cup \mathsf{a}(\mathcal{H}_2^{@})$.

The statement follows trivially by application of the *i.h.* on $Q_1\langle x\rangle$ and $Q_2\langle x\rangle$ and by Lemma 8.3.1 (Unique decomposition of Λ-terms) on $\mathcal{H}_1\langle t_1\rangle$ and $\mathcal{H}_2^{@}\langle t_2\rangle$.

· The case of rules $\mathsf{E_{VAR}}$, $\mathsf{E_I}$ and $\mathsf{E_U}$ are all ruled out by Lemma 8.2.2 (Redex in non-normal terms).

· Suppose $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived as follows

$$\frac{Q_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2} \quad x \notin (\mathcal{U}_2 \cup \mathcal{A}_2)}{Q_2@[x{\leftarrow}\tilde{s}_2] \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}}\ \mathsf{E_{GC}}$$

with $P_2 = Q_2@[x{\leftarrow}\tilde{s}_2]$. Given Definition 45 (Reduction places in Useful Open CbNeed), if $t_2 = y \neq x$ then by *i.h.* we would get that $Q_1 = Q_2$ and that $x = t_1 = t_2 = y$; absurd. Moreover, if $t_2 = x$ then $\tilde{s}_2 \in \mathsf{Val}$ but also $\mathcal{H}_1\langle t_1\rangle = \tilde{s}_2 \in \mathsf{Val}$; absurd by Lemma 8.2.2 (Redex in non-normal terms). Therefore, it could only be that $t_2 = (\lambda x_2.u_2)s_2$—again, by Definition 45 (Reduction places in Useful Ppen CbNeed). However, applying the *i.h.* on $Q_1\langle x\rangle$ and $Q_2\langle t_2\rangle$ gives that $x = t_2$; absurd. Therefore, this case is impossible.

· Let $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ be derived as follows

$$\frac{Q_2 \in \mathcal{E}^{@}_{\mathcal{V}_2,\mathcal{A}_2} \quad x \notin \mathcal{B}_2}{Q_2\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{V}_2\setminus\{x\},\mathcal{A}_2}}\ \mathsf{E_{NL}}$$

where $P_2 = Q_2\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle]$ and $\mathcal{U}_2 = \mathcal{V}_2 \setminus \{x\}$. We can apply the *i.h.* on $Q_1\langle x\rangle$ and $Q_2\langle x\rangle$ taking $S \coloneqq \mathcal{B}_1 \cup \mathcal{B}_2$ to obtain that $Q_1 = Q_2$. Finally, we can apply Lemma 8.3.1 (Unique decomposition of Λ-terms) to obtain that $\mathcal{H} = \langle\cdot\rangle$—and so $P_1 = P_2$—and $t_1 = t_2$.

∗ *Rule* $\mathsf{M_{VAR}}$: Let $P_1 \in \mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1}$ be derived as follows

$$\frac{Q_1 \in \mathcal{E}_{\mathcal{V}_1,\mathcal{B}_1} \quad x \in (\mathcal{V}_1 \cup \mathcal{B}_1)}{Q_1@[x{\leftarrow}z] \in \mathcal{E}_{\mathsf{upd}(\mathcal{V}_1,x,z),\mathsf{upd}(\mathcal{B}_1,x,z)}}\ \mathsf{M_{VAR}}$$

where $P_1 = Q_1@[x{\leftarrow}z]$, $\mathcal{U}_1 = \mathsf{upd}(\mathcal{V}_1, x, z)$ and $\mathcal{A}_1 = \mathsf{upd}(\mathcal{B}_1, x, z)$. Case analysis on the last derivation rule in $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$:

· The case of rule $\mathsf{E_{AX_1}}$ is impossible.

· The case of rule $\mathsf{E_{AX_2}}$ is impossible because z cannot be rewritten as an applicative term context with t_2 plugged into it.

· The case of rule $\mathsf{E_{VAR}}$ is given simply by *i.h.*.

· The cases of rules $\mathsf{E_I}$ and $\mathsf{E_U}$ are ruled out by the fact that non-variable inert Λ-terms and values cannot be rewritten as z.

· Let $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ be derived as follows:

$$\frac{Q_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2} \quad x \notin (\mathcal{U}_2 \cup \mathcal{A}_2)}{Q_2@[x{\leftarrow}z] \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}}\ \mathsf{E_{GC}}$$

with $P_2 = Q_2@[x{\leftarrow}z]$. By application of the *i.h.* on $Q_1\langle t_1\rangle$ and $Q_2\langle t_2\rangle$, we have that $Q_1 = Q_2$—and so $P_1 = P_2$—and $t_1 = t_2$.

· Suppose P_2 is derived as follows

$$\frac{Q_2 \in \mathcal{E}^{@}_{\mathcal{V}_2,\mathcal{A}_2} \quad x \notin \mathcal{B}_2}{Q_2\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{V}_2\setminus\{x\},\mathcal{A}_2}}\ \mathsf{E_{NL}}$$

with $P_2 = Q_2\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle]$ and $\mathcal{U}_2 = \mathcal{V}_2 \setminus \{x\}$. But then application of the *i.h.* gives that $Q_1 = Q_2$ and $t_1 = x$, and so by Definition 45 (Reduction places in Useful Open CbNeed) it must be that $z \in \mathsf{Val}$; absurd.

∗ *Rule* $\mathsf{M_I}$: Let $P_1 \in \mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1}$ be derived as follows:

$$\frac{Q_1 \in \mathcal{E}_{\mathcal{V}_1,\mathcal{B}_1} \quad x \in (\mathcal{V}_1 \cup \mathcal{B}_1)}{Q_1@[x{\leftarrow}i_1^+] \in \mathcal{E}_{(\mathcal{V}_1\setminus\{x\})\cup\mathsf{u}(i^+),(\mathcal{B}_1\setminus\{x\})\cup\mathsf{a}(i^+)}}\ \mathsf{M_I}$$

where $P_1 = Q_1@[x{\leftarrow}i_1^+]$, $\mathcal{U}_1 = (\mathcal{V}_1 \setminus \{x\}) \cup \mathsf{u}(i^+)$, $\mathcal{A}_1 = (\mathcal{A}_1 \setminus \{x\}) \cup \mathsf{a}(i^+)$. Case analysis on the last derivation rule in $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$:

· The case of rule $\mathsf{E}_{\mathsf{AX}_1}$ is impossible.
· Suppose $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived as follows

$$\frac{Q_2 \in \mathcal{E}_{\mathcal{V}_2,\mathcal{B}_2} \quad x \notin (\mathcal{V}_2 \cup \mathcal{B}_2)}{Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}^{@}_2] \in \mathcal{E}^{@}_{(\mathcal{V}_2\setminus\{x\})\cup \mathsf{u}(\mathcal{H}^{@}_2),(\mathcal{B}_2\setminus\{x\})\cup \mathsf{a}(\mathcal{H}^{@}_2)}}\ \mathsf{E}_{\mathsf{AX}_2}$$

with $P_2 = Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}^{@}_2], \mathcal{U}_2 = (\mathcal{V}_2\setminus\{x\})\cup\mathsf{u}(\mathcal{H}^{@}_2)$ and $\mathcal{A}_2 = (\mathcal{B}_2\setminus\{x\})\cup\mathsf{a}(\mathcal{H}^{@}_2)$. Note that if $t_1 = x$ then it should be that $i^+_1 \in \mathsf{Val}$—by Definition 45 (Reduction places in Useful Open CbNeed); absurd. Moreover, if $t_1 = y \neq x$ or if $t_1 = (\lambda x_1.u_1)s_1$, then application of the *i.h.* on $Q_1\langle t_1\rangle$ and $Q_2\langle x\rangle$ gives that $t_1 = x$; absurd in either case. Therefore, this case is impossible.
· The cases of rules $\mathsf{E}_{\mathsf{VAR}}$ and E_{U} are impossible because i^+ cannot be rewritten as a variable or a value.
· The case of E_{I} follows by application of the *i.h.*.
· Let $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ be derived as follows

$$\frac{Q_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2} \quad x \notin (\mathcal{U}_2 \cup \mathcal{A}_2)}{Q_2@[x{\leftarrow}i^+_1] \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}}\ \mathsf{E}_{\mathsf{GC}}$$

where $P_2 = Q_2@[x{\leftarrow}i^+_1]$. Note that $t_1 \neq x$—otherwise $i^+ \in \mathsf{Val}$; absurd. Moreover, $t_2 \neq x$—otherwise application of the *i.h.* on $Q_1\langle t_1\rangle$ and $Q_2\langle t_2\rangle$ would give that $t_1 = t_2 = x$; absurd. If $t_1, t_2 \neq x$, we can apply the *i.h.* on $Q_1\langle t_1\rangle$ and $Q_2\langle t_2\rangle$ to get that $Q_1 = Q_2$—and so $P_1 = P_2$—and $t_1 = t_2$.
· Suppose $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived as follows

$$\frac{Q_2 \in \mathcal{E}^{@}_{\mathcal{V}_2,\mathcal{A}_2} \quad x \notin \mathcal{A}_2}{Q_2\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{V}_2\setminus\{x\},\mathcal{A}_2}}\ \mathsf{E}_{\mathsf{NL}}$$

with $P_2 = Q_2\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle]$ and $\mathcal{U}_2 = \mathcal{V}_2\setminus\{x\}$. However, application of the *i.h.* on $Q_1\langle t_1\rangle$ and $Q_2\langle x\rangle$ would give that $t_1 = x$, which would imply that $i^+_1 \in \mathsf{Val}$—by Definition 45 (Reduction places in Useful Open CbNeed); absurd.

∗ *Rule* M_{U}: Let $P_1 \in \mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1}$ be derived as follows:

$$\frac{Q_1 \in \mathcal{E}_{\mathcal{V}_1,\mathcal{A}_1} \quad x \in (\mathcal{V}_1 \setminus \mathcal{A}_1)}{Q_1@[x{\leftarrow}v_1] \in \mathcal{E}_{\mathcal{V}_1\setminus\{x\},\mathcal{A}_1}}\ \mathsf{M}_{\mathsf{U}}$$

where $P_1 = Q_1@[x{\leftarrow}v_1]$ and $\mathcal{U}_1 = \mathcal{V}_1 \setminus \{x\}$. Case analysis on the last derivation rule in $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$:
· The case of rule $\mathsf{E}_{\mathsf{AX}_1}$ is impossible.
· Suppose $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived as follows

$$\frac{Q_2 \in \mathcal{E}_{\mathcal{V}_2,\mathcal{B}_2} \quad x \notin (\mathcal{V}_2 \cup \mathcal{B}_2)}{Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}^{@}_2] \in \mathcal{E}^{@}_{(\mathcal{V}_2\setminus\{x\})\cup \mathsf{u}(\mathcal{H}^{@}_2),(\mathcal{B}_2\setminus\{x\})\cup \mathsf{a}(\mathcal{H}^{@}_2)}}\ \mathsf{E}_{\mathsf{AX}_2}$$

with $P_2 = Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}^{@}_2], \mathcal{U}_2 = (\mathcal{V}_2\setminus\{x\})\cup\mathsf{u}(\mathcal{H}^{@}_2)$ and $\mathcal{A}_2 = (\mathcal{B}_2\setminus\{x\})\cup\mathsf{a}(\mathcal{H}^{@}_2)$. Note that by Definition 45 (Reduction places in Useful Open CbNeed) it would be that $t_2 = (\lambda x_2.u_2)s_2$. This is absurd, because v_1 cannot be rewritten as $\mathcal{H}^{@}_2\langle(\lambda x_2.u_2)s_2\rangle$—Lemma 8.2.2 (Redex in non-normal terms).

- The cases of rules $\mathsf{E_{VAR}}$ and $\mathsf{E_I}$ are impossible because v_1 cannot be rewritten as a variable or a value.
- The case of $\mathsf{E_I}$ follows by application of the *i.h.*.
- Let $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ be derived as follows

$$\frac{Q_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2} \quad x \notin (\mathcal{U}_2 \cup \mathcal{A}_2)}{Q_2@[x{\leftarrow}i_2^+] \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}}\ \mathsf{E_{GC}}$$

  where $P_2 = Q_2@[x{\leftarrow}i^+]$. Note that $t_1 \neq x$—otherwise $i^+ \in \mathsf{Val}$; absurd. By application of the *i.h.* on $Q_1\langle t_1\rangle$ and $Q_2\langle t_2\rangle$ gives that $Q_1 = Q_2$—and so $P_1 = P_2$—and $t_1 = t_2$.
- In the case where $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived via an application of rule $\mathsf{E_U}$, the statement follows by *i.h.* like in the previous item.
- Suppose $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived as follows

$$\frac{Q_2 \in \mathcal{E}^{@}_{\mathcal{V}_2,\mathcal{A}_2} \quad x \notin \mathcal{A}_2}{Q_2\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{V}_2\setminus\{x\},\mathcal{A}_2}}\ \mathsf{E_{NL}}$$

  with $P_2 = Q_2\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle]$ and $\mathcal{U}_2 = \mathcal{V}_2 \setminus \{x\}$. However, note that it should be that $v_1 = t_2$. By Lemma 8.2.2 (Redex in non-normal terms), this is absurd.

∗ *Rule* $\mathsf{M_{GC}}$: Let $P_1 \in \mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1}$ be derived as follows:

$$\frac{Q_1 \in \mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1} \quad x \notin (\mathcal{V}_1 \cup \mathcal{A}_1)}{Q_1@[x{\leftarrow}\tilde{s}_1] \in \mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1}}\ \mathsf{M_{GC}}$$

where $P_1 = Q_1@[x{\leftarrow}\tilde{s}_1]$. Case analysis on the last derivation rule in $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$:

- The case of rule $\mathsf{E_{AX_1}}$ is impossible.
- Suppose $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived as follows

$$\frac{Q_2 \in \mathcal{E}_{\mathcal{V}_2,\mathcal{B}_2} \quad x \notin (\mathcal{V}_2 \cup \mathcal{B}_2)}{Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}_2^{@}] \in \mathcal{E}^{@}_{(\mathcal{V}_2\setminus\{x\})\cup\mathsf{u}(\mathcal{H}_2^{@}),(\mathcal{B}_2\setminus\{x\})\cup\mathsf{a}(\mathcal{H}_2^{@})}}\ \mathsf{E_{AX_2}}$$

  with $P_2 = Q_2\langle x\rangle@[x{\leftarrow}\mathcal{H}_2^{@}], \mathcal{U}_2 = (\mathcal{V}_2\setminus\{x\})\cup\mathsf{u}(\mathcal{H}_2^{@})$ and $\mathcal{A}_2 = (\mathcal{B}_2\setminus\{x\})\cup\mathsf{a}(\mathcal{H}_2^{@})$. However, by application of the *i.h.* on $Q_1\langle t_1\rangle$ and $Q_2\langle x\rangle$ we would have that $t_1 = x$, forcing $\tilde{s}_1 = \mathcal{H}_2^{@}\langle t_2\rangle$ to be a value—by Definition 45 (Reduction places in Useful Open CbNeed); absurd—by Lemma 8.2.2 (Redex in non-normal terms).
- Let $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ be derived as follows

$$\frac{Q_2 \in \mathcal{E}^{@}_{\mathcal{V}_2,\mathcal{B}_2}}{Q_2@[x{\leftarrow}z_2] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{V}_2,x,z_2),\mathsf{upd}(\mathcal{B}_2,x,z_2)}}\ \mathsf{E_{VAR}}$$

  where $P_2 = Q_2@[x{\leftarrow}z_2]$, $\mathcal{U}_2 = \mathsf{upd}(\mathcal{V}_2, x, z_2)$ and $\mathcal{A}_2 = \mathsf{upd}(\mathcal{B}_2, x, z_2)$. We can then apply the *i.h.* on $Q_1\langle t_1\rangle$ and $Q_2\langle t_2\rangle$ to obtain that $Q_1 = Q_2$—and so $P_1 = P_2$—and $t_1 = t_2$.
- In the cases where $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ s derived via an application of rules $\mathsf{E_I}$, $\mathsf{E_U}$ or $\mathsf{E_{GC}}$, the statement follows by *i.h.*.
- Finally, the case where $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived via an application of rule $\mathsf{E_{NL}}$ is ruled out as the case where it is derived via an application of rule $\mathsf{E_{AX_2}}$.

- Let $P_1 \in \mathcal{E}^{@}_{\mathcal{U}_1,\mathcal{A}_1}$. The case where $P_2 \in \mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2}$ was already covered when we took P_1 to be a multiplicative evaluation context and P_2 to be an exponential one. Let $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}2}$. We proceed by induction on the derivation of $P_1 \in \mathcal{E}^{@}_{\mathcal{U}_1,\mathcal{A}_1}$. Case analysis on the last derivation rule:
  - *Rule* $\mathsf{E}_{\mathsf{AX}_1}$: Let $P_1 \in \mathcal{E}^{@}_{\mathcal{U}_1,\mathcal{A}_1}$ be derived as follows

$$\frac{}{(\mathcal{H}^{@}_1, \epsilon) \in \mathcal{E}^{@}_{\mathsf{u}(\mathcal{H}^{@}_1),\mathsf{a}(\mathcal{H}^{@}_1)}}\ \mathsf{E}_{\mathsf{AX}_1}$$

    where $P_1 = (\mathcal{H}^{@}_1, \epsilon)$, $\mathcal{U}_1 =$ and $\mathcal{A}_1 = \mathsf{a}(\mathcal{H}^{@}_1)$. Then it must be that $P_2 = (\mathcal{H}^{@}_2, \epsilon)$, and so Lemma 8.3.1 (Unique decomposition of Λ-terms) gives us that $\mathcal{H}^{@}_1 = \mathcal{H}^{@}_2$ and so $P_1 = P_2$ and $t_1 = t_2$.
  - *Rule* $\mathsf{E}_{\mathsf{AX}_2}$: Let $P_1 \in \mathcal{E}^{@}_{\mathcal{U}_1,\mathcal{A}_1}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V}_1,\mathcal{B}_1} \quad x \notin (\mathcal{V}_1 \cup \mathcal{B}_1)}{Q_1\langle x\rangle@[x{\leftarrow}\mathcal{H}^{@}_1] \in \mathcal{E}^{@}_{(\mathcal{V}_1\setminus\{x\})\cup\mathsf{u}(\mathcal{H}^{@}_1),(\mathcal{B}_1\setminus\{x\})\cup\mathsf{a}(\mathcal{H}^{@}_1)}}\ \mathsf{E}_{\mathsf{AX}_2}$$

    where $P_1 = Q_1\langle x\rangle@[x{\leftarrow}\mathcal{H}^{@}_1]$, $\mathcal{U}_1 = (\mathcal{V}_1 \setminus \{x\}) \cup \mathsf{u}(\mathcal{H}^{@}_1)$ and $\mathcal{A}_1 = \mathsf{a}(\mathcal{H}^{@}_1)$. By proceeding by case analysis on the last derivation rule in $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$, we find that
    - Rule $\mathsf{E}_{\mathsf{AX}_1}$ is impossible.
    - In the case where $\mathsf{E}_{\mathsf{AX}_2}$ is the last derivation rule in $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$, the statement follows by *i.h.* and by application of Lemma 8.3.1 (Unique decomposition of Λ-terms).
    - Rules $\mathsf{E}_{\mathsf{VAR}}$, E_{I} and E_{U} are impossible, because normal terms cannot be rewritten as $\mathcal{H}^{@}_1\langle t_1\rangle$—by Lemma 8.2.2 (Redex in non-normal terms).
    - Suppose $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ was derived as follows

$$\frac{Q_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2} \quad x \notin (\mathcal{U}_2 \cup \mathcal{A}_2)}{Q_2@[x{\leftarrow}\tilde{s}_2] \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}}\ \mathsf{E}_{\mathsf{GC}}$$

      where $P_2 = Q_2@[x{\leftarrow}\tilde{s}_2]$. But then application of the *i.h.* gives that $x = t_2$, implying that $\mathcal{H}^{@}_1\langle t_1\rangle = \tilde{s}_2 \in \mathsf{Val}$; by application of Lemma 8.2.2 (Redex in non-normal terms), this is absurd.
    - Rule E_{NL} is impossible because $\langle\cdot\rangle$ is not applicative—Lemma 8.3.1 (Unique decomposition of Λ-terms) would otherwise give that $\mathcal{H}^{@}_1 = \langle\cdot\rangle$.
  - *Rule* $\mathsf{E}_{\mathsf{VAR}}$: Let $P_1 \in \mathcal{E}^{@}_{\mathcal{U}_1,\mathcal{A}_1}$ be derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V}_1,\mathcal{B}_1} \quad x \in (\mathcal{V}_1 \cup \mathcal{B}_1)}{Q_1@[x{\leftarrow}z_1] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{V}_1,x,z_1),\mathsf{upd}(\mathcal{B}_1,x,z_1)}}\ \mathsf{E}_{\mathsf{VAR}}$$

    where $P_1 = Q_1@[x{\leftarrow}z_1]$, $\mathcal{U}_1 = \mathsf{upd}(\mathcal{V}_1, x, z_1)$ and $\mathcal{A}_1 = \mathsf{upd}(\mathcal{B}_1, x, z_1)$. By proceeding by case analysis on the last derivation rule in $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$, we find that
    - Rule $\mathsf{E}_{\mathsf{AX}_1}$ is impossible.
    - Rules $\mathsf{E}_{\mathsf{AX}_2}$ and E_{NL} are ruled out by application of the *i.h.* on the underlying exponential evaluation contexts.
    - In the cases where $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived via an application of rule $\mathsf{E}_{\mathsf{VAR}}$ or an application of rule E_{GC}, the statement follows by *i.h.*.
    - Rules E_{I} and E_{U} are impossible because z is not a non-variable inert Λ-term nor is it a value.

– *Rule* $\mathsf{E_I}$, *Rule* $\mathsf{E_U}$, *Rule* $\mathsf{E_{GC}}$: This cases are treated analogously to rule $\mathsf{E_{VAR}}$.
– *Rule* $\mathsf{E_{NL}}$: Let $P_1 \in \mathcal{E}^{@}_{\mathcal{U}_1,\mathcal{A}_1}$ be derived as follows

$$\frac{Q_1 \in \mathcal{E}^{@}_{\mathcal{V}_1,\mathcal{A}_1} \quad x \notin \mathcal{A}_1}{Q_1\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{V}_1\setminus\{x\},\mathcal{A}_1}}\ \mathsf{E_{NL}}$$

where $P_1 = Q_1\langle x\rangle@[x{\leftarrow}\langle\cdot\rangle]$ and $\mathcal{U}_1 = \mathcal{V}_1 \setminus \{x\}$. Note that $t_1 = (\lambda x_1.u_1)s_1$. By proceeding by case analysis on the last derivation rule in $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$, we find that
  * Rule $\mathsf{E_{AX_1}}$ is impossible.
  * Rule $\mathsf{E_{AX_2}}$ is impossible because $\langle\cdot\rangle$ is not applicative—Lemma 8.3.1 (Unique decomposition of Λ-terms) would otherwise give that $\mathcal{H}^{@}_2 = \langle\cdot\rangle$.
  * By application of the *i.h.*, rules $\mathsf{E_{VAR}}$ (resp. $\mathsf{E_I}$; $\mathsf{E_U}$) is excluded from the last possible derivation rule in $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ because variables (resp. useful inert terms ; values) cannot be rewritten as $t_1 = (\lambda x_1.u_1)s_1$.
  * Suppose $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$ is derived as follows

$$\frac{Q_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2} \quad x \notin (\mathcal{U}_2 \cup \mathcal{A}_2)}{Q_2@[x{\leftarrow}\tilde{s}_2] \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}}\ \mathsf{E_{GC}}$$

where $P_2 =$. But then application of the *i.h.* on $Q_1\langle x\rangle$ and $Q_2\langle t_2\rangle$ give that $x = t_2$ and so it should be that $\tilde{s}_2 \in \mathsf{Val}$. But then $\tilde{s}_2 = t_1 = (\lambda x_1.u_1)s_1$; absurd by Lemma 8.2.2 (Redex in non-normal terms).
  * In the case where $\mathsf{E_{NL}}$ is the last derivation rule in $P_2 \in \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2}$, the statement follows by *i.h.*.

□

(Click here to go back to main chapter.)

Corollary 13.5.19 (Determinism of Useful Open CbNeed)**.**
If $p \to_{\mathsf{und}} q$ and $p \to_{\mathsf{und}} r$ then $q = r$.

Proof. *(Click here to go back to main chapter.)*
Let $P_1 \in (\mathcal{E}_{\mathcal{U}_1,\mathcal{A}_1} \cup \mathcal{E}^{@}_{\mathcal{U}_1,\mathcal{A}_1})$ be such that $p = P_1\langle t_1\rangle \to_{\mathsf{und}} P_1\langle m_1\rangle = q$, and let $P_2 \in (\mathcal{E}_{\mathcal{U}_2,\mathcal{A}_2} \cup \mathcal{E}^{@}_{\mathcal{U}_2,\mathcal{A}_2})$ be such that $p = P_2\langle t_2\rangle \to_{\mathsf{und}} P_2\langle m_2\rangle = r$.

First, note that if $p = P_1\langle t_1\rangle \to_{\mathsf{um}} P_1\langle m_1\rangle = q$, then it must be that t_1 is a β-redex. Similarly, if $p = P_1\langle t_1\rangle \to_{\mathsf{ue}} P_1\langle m_1\rangle = q$, then it must be that $t_1 \in \mathsf{Var}$, $t_1 \in \mathsf{dom}(P_1)$, and $P(t_1) \in \mathsf{Val}$. This implies that t_1 is a S-reduction place of $P_1\langle t_1\rangle$ for *any* S. Similarly, we can prove that t_2 is a S-reduction place $P_2\langle t_2\rangle$ for *any* S.

Thus, if we take $S \coloneqq \mathsf{Var}$, we may apply Lemma 8.3.2 (Unique decomposition of programs) to obtain that $P_1 = P_2$ and $t_1 = t_2$. That is, $q = r$.

□

(Click here to go back to main chapter.)

13.6 Proofs of Chapter 9 (Multi types for Useful Open CbNeed

13.6.1 Useful Open CbNeed correctness.

Lemma 13.6.1 (Relevance of the Useful Open CbNeed type system)**.**
Let e *be an expression and* $\Phi \triangleright_U \Gamma \vdash^{(m,e,r)} \mathsf{e} : M$ *be a type derivation. If* $x \notin \mathsf{fv}(\mathsf{e})$ *then* $x \notin \mathsf{dom}(\Gamma)$*.*

Proof. (Click here to go back to main chapter.)
The Open CbNeed multi type system being mostly based on the CbNeed one, this is trivially provable by induction on the number of typing rules applied in Φ.

□

(Click here to go back to main chapter.)

Lemma 13.6.2 (Typing properties of values)**.**
Let $\Phi \triangleright_U \Gamma \vdash^{(m,e,r)} v : M$ *with* $M \in \mathsf{Tight}$*.*
Then $(m, e, r) = (0, 0, |v|_{\mathsf{nd}})$*,* $\mathsf{dom}(\Gamma) = \mathsf{nv}(v)$*, and* $M \in \mathsf{Abs}$*.*

Proof. (Click here to go back to main chapter.)
Since values cannot be inert-typed, then Φ must be of the form

$$\dfrac{\left(\dfrac{}{\emptyset \vdash^{(0,0,0)} \lambda x.u : \mathsf{abs}}\ \mathsf{abs}\right)_{i\in I}}{\emptyset \vdash^{(0,0,0)} \lambda x.u : [\mathsf{abs}]_{i\in I}}\ \mathsf{many}$$

□

(Click here to go back to main chapter.)

Lemma 13.6.3 (Typing properties of useful inert terms)**.**
Let $\Phi \triangleright_U \Gamma \vdash^{(m,e,r)} i^+ : M$ *such that* $\mathsf{tight}_\Gamma(\mathsf{a}(i^+))$*.*
Then $(m, e, r) = (0, 0, |i^+|_{\mathsf{nd}})$*,* $\mathsf{dom}(\Gamma) = \mathsf{nv}(i^+)$*,* $\mathsf{tight}(\Gamma)$*, and* $M \in \mathsf{Inert}$*.*

Proof. (Click here to go back to main chapter.)
Note that i^+ may only be an application Λ_{L}-term. Say $i^+ = us$. We proceed by structural induction on i^+, proceeding by case analysis on the shape of u:

- Let $u = x \in \mathsf{Var}$. Then $|i^+|_{\mathsf{nd}} = |s|_{\mathsf{nd}} + 1$. Moreover, $\mathsf{a}(i^+) = \{x\} \cup \mathsf{a}(s)$, implying that $\mathsf{tight}_\Gamma(\mathsf{a}(i^+))$ and so $\Gamma(x) \in \mathsf{Tight}$. Hence, Φ must be of the form

$$\dfrac{\dfrac{J \neq \emptyset}{x : [\mathsf{inert}]_{j\in J} \vdash^{(0,0,0)} x : [\mathsf{inert}]_{j\in J}}\ \mathsf{ax}_T \qquad \Theta \triangleright_U \Delta \vdash^{(m'',e'',r'')} s : [\mathsf{tight}]}{\{x : [\mathsf{inert}]_{j\in J}\} \biguplus \Delta \vdash^{(m'',e'',r''+1)} us : [\mathsf{inert}]_{j\in J}}\ \mathsf{app}_i$$

  We proceed by case analysis on the shape of s:
  - The statement follows easily if $s \in \mathsf{Var}$.
  - Let $s = j^+$ for some non-variable inert Λ-term j^+. The statement follows by *i.h.* on Θ, applicable because $\mathsf{tight}_\Gamma(\mathsf{a}(i^+))$ implies that $\mathsf{tight}_\Delta(\mathsf{a}(j^+))$.

– Let $s \in \mathsf{Val}$. The statement follows by application of Lemma 9.1.2 (Typing properties of values) on Θ.

- Let $u = j^+$. It is easily shown by *i.h.* on the sub-type derivation corresponding to u that $M \in \mathsf{Inert}$. Thus, Φ can only be of the following form

$$\frac{\Psi \rhd_U \Pi \vdash^{(m',e',r')} u : [\mathsf{inert}]_{j\in J} \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} s : [\mathsf{tight}] \quad J \neq \emptyset}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r''+1)} us : [\mathsf{inert}]_{j\in J}}\ \mathsf{app}_i$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'' + 1)$. By application of the *i.h.* on Ψ, we have that $\Psi \rhd_U \Pi \vdash^{(0,0,|u|_{\mathsf{nd}})} u : [\mathsf{inert}]_{j\in J}$, with $\mathsf{dom}(\Pi) = \mathsf{nv}(u)$ and $\mathsf{tight}(\Pi)$. We proceed by case analysis on the shape of s:

– The statement follows easily if $s \in \mathsf{Var}$.

– Let $s = k^+$. The statement follows by *i.h.* on Θ, applicable because $\mathsf{tight}_\Gamma(\mathsf{a}(i^+))$ implies that $\mathsf{tight}_\Delta(\mathsf{a}(s))$.

– Let $s \in \mathsf{Val}$. The statement follows by Lemma 9.1.2 (Typing properties of values) on Θ. □

(Click here to go back to main chapter.)

Lemma 13.6.4 (Typing properties of generalized variables)**.**
Let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} p : M$ *such that* $\mathsf{genVar}_x(p)$.
Then $\mathsf{dom}(\Gamma) = \mathsf{nv}(p) = \{x\}$ *and* $\Gamma(x) = M$. *Moreover, if* $M \in \mathsf{Tight}$ *then* $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$.

Proof. (Click here to go back to main chapter.)
Let $\mathsf{genVar}_x(p)$. By Lemma 8.2.4 (Properties of generalized variables), we have that $\mathsf{nv}(p) = x$. We proceed by induction on the derivation of $\mathsf{genVar}_x(p)$:

- The case where $p = (x, \epsilon)$ is trivial.
- Let $\mathsf{genVar}_x(p)$ be derived as

$$\frac{\mathsf{genVar}_y(q)}{\mathsf{genVar}_x(q@[y{\leftarrow}x])}\ \mathsf{GV}_{\mathsf{HER}}$$

with $p = q@[y{\leftarrow}x]$. It is easy to prove by *i.h.* that y is on the type context of the sub-type derivation corresponding to q, and so $\mathrm{ES}_{\mathrm{gc}}$ cannot be the last typing rule in Φ. Hence, the latter must be of the form

$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m',e',r')} q : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} x : N}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} q@[y{\leftarrow}x] : M}\ \mathrm{ES}$$

By *i.h.* on Ψ, note that Φ must in fact be of the form

$$\frac{\Psi \rhd_U y : M \vdash^{(m',e',r')} q : M \quad \dfrac{}{\Theta \rhd_U x : M \vdash^{(m'',e'',r'')} x : M}\ \{\mathsf{ax}, \mathsf{ax}_T\}}{\{x : M\} \vdash^{(m'+m'',e'+e'',r'+r'')} q@[y{\leftarrow}x] : M}\ \mathrm{ES}$$

The 'Moreover' part follows easily, in particular by the fact that $y \in \mathsf{nv}(q)$—by Lemma 8.2.4 (Properties of generalized variables)—and so $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |x|_{\mathsf{nd}} = |q|_{\mathsf{nd}}$.

- Let $\mathsf{genVar}_x(p)$ be derived as
$$\frac{\mathsf{genVar}_x(q) \quad y \neq x}{\mathsf{genVar}_x(q@[y{\leftarrow}t])}\ \mathsf{GV_{GC}}$$
with $p = q@[y{\leftarrow}t]$. It is easy to prove by *i.h.* that y is not in the domain of the type context of the sub-type derivation corresponding to q, and so ES cannot be the last typing rule in Φ. Hence, the latter must be of the form
$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} q : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} q@[x{\leftarrow}y] : M}\ \mathrm{ES_{gc}}$$
By *i.h.* on Ψ, we have that $\Psi \rhd_U \{x : M\} \vdash^{(m,e,r)} q : M$. The 'Moreover' part follows by the fact that $y \notin \mathsf{nv}(q)$—by Lemma 8.2.4 (Properties of generalized variables)—and so $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}}$. □

(Click here to go back to main chapter.)

Lemma 13.6.5 (Typing properties of useful abstraction programs)**.**
Let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} p : M$ *such that*
- $\mathsf{uabs}(p)$*, and*
- $M \in \mathsf{Tight}$*.*

Then $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$*,* $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$*, and* $M \in \mathsf{Abs}$*.*

Proof. (Click here to go back to main chapter.)
By induction on the derivation of $\mathsf{uabs}(p)$:
- The statement holds by Lemma 9.1.2 (Typing properties of values) if $p = (v, \epsilon)$.
- Let $\mathsf{uabs}(p)$ be derived as
$$\frac{\mathsf{genVar}_x(q)}{\mathsf{uabs}(q@[x{\leftarrow}v])}\ \mathsf{A_{GV}}$$
with $p = q@[x{\leftarrow}v]$. Note that $\mathsf{nv}(q@[x{\leftarrow}v]) = \mathsf{nv}(v)$—by Lemma 8.2.4 (Properties of generalized variables)—and so $|q@[x{\leftarrow}v]|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |v|_{\mathsf{nd}}$. We do case analysis on the last typing rule in Φ:
  - Let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} p : M$ be of the form
$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m',e',r')} q : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} v : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} q@[x{\leftarrow}v] : M}\ \mathrm{ES}$$
where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. By application of the *i.h.* on Ψ, it must be that Φ is of the form
$$\frac{\Psi \rhd_U x : M \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} v : M}{\Delta \vdash^{(m'',e'',|q|_{\mathsf{nd}}+r'')} q@[x{\leftarrow}v] : M}\ \mathrm{ES}$$
Since $M \in \mathsf{Tight}$ by hypothesis, we can apply the *i.h.* on Θ to obtain that $\Theta \rhd_U \Delta \vdash^{(0,0,|t|_{\mathsf{nd}})} v : M$, with $\mathsf{dom}(\Delta) = \mathsf{nv}(v)$ and $\mathsf{tight}(\Delta)$. Thus, Φ satisfies the statement.

– Suppose Φ is of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} q : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} q@[x{\leftarrow}v] : M}\ \mathrm{ES_{gc}}$$

However, an application of the *i.h.* on Ψ followed by an application of Lemma 8.2.4 (Properties of generalized variables) would yield that $x \in \mathsf{dom}(\Gamma)$, which makes this case absurd.

- Let $\mathsf{uabs}(p)$ be derived as

$$\frac{\mathsf{uabs}(q)}{\mathsf{uabs}(q@[x{\leftarrow}t])}\ \mathsf{GV_{GC}}$$

with $p = q@[x{\leftarrow}t]$. Note that $\mathsf{nv}(q@[x{\leftarrow}t]) = \mathsf{nv}(q)$ and so $|q@[x{\leftarrow}t]|_{\mathsf{nd}} = |q|_{\mathsf{nd}}$. We proceed by induction on the last typing rule in Φ:

– Suppose Φ is of the form

$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m',e',r')} q : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} q@[x{\leftarrow}t] : M}\ \mathrm{ES}$$

However, an application of the *i.h.* on Ψ together with an application of Lemma 8.2.5 (Properties of useful abstraction programs) would yield that $y \notin \mathsf{dom}(\Pi; x : N) = \emptyset$. This would mean that $N = \mathbf{0}$, making this case absurd.

– Let Φ be derived as

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} q : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} q@[x{\leftarrow}t] : M}\ \mathrm{ES_{gc}}$$

The statement then holds by *i.h.* on Ψ.

□

(Click here to go back to main chapter.)

Lemma 13.6.6 (Typing properties of useful inert programs)**.**
Let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} p : M$ *such that*

- $\mathsf{uinert}(p)$,
- $\mathsf{tight}_\Gamma(\mathsf{a}(p))$.

Then $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$, $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$, $\mathsf{tight}(\Gamma)$, *and* $M \in \mathsf{Inert}$.

Proof. (Click here to go back to main chapter.)
By induction on the derivation of $\mathsf{uinert}(p)$:

- If $\mathsf{I}_{\mathsf{Lift}}$ is the last rule in $\mathsf{uinert}(p)$, then the statement holds by Lemma 9.1.3 (Typing properties of useful inert terms).
- Let $\mathsf{uinert}(p)$ be derived as

$$\frac{\mathsf{uinert}(q) \quad x \in \mathsf{nv}(q)}{\mathsf{uinert}(q@[x{\leftarrow}i])}\ \mathsf{I_I}$$

with $q = q@[x{\leftarrow}i]$. Note that $\mathsf{nv}(p) = (\mathsf{nv}(q) \setminus \{x\}) \cup \mathsf{nv}(i)$ and $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |i|_{\mathsf{nd}}$. We proceed by case analysis on whether $x \in \mathsf{a}(q)$:

– Let $x \notin \mathsf{a}(q)$. We proceed by case analysis on the last typing rule in Φ:

∗ Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m',e',r')} q : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} q@[x \leftarrow t] : M}\ \text{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. Note then $\mathsf{tight}_\Gamma(\mathsf{a}(q))$ implies that $\mathsf{tight}_{\Pi;x:N}(\mathsf{a}(q))$. We can thus apply the *i.h.* on Ψ to obtain that $\Psi \rhd_U \Pi; x : N \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M$, with $\mathsf{dom}(\Pi; x : N) = \mathsf{nv}(q)$, $\mathsf{tight}(\Pi; x : N)$ and $M \in \mathsf{Inert}$. In particular, this means that $N \in \mathsf{Tight}$, so if $i \in \mathsf{Var}$ then the statement follows easily.
Moreover, if $i \notin \mathsf{Var}$ then we can apply the *i.h.* on Θ—since $\mathsf{tight}_\Gamma(\mathsf{a}(p))$ implies that $\mathsf{tight}_\Delta(\mathsf{a}(i))$—to obtain that $\Theta \rhd_U \Delta \vdash^{(0,0,|i|_{\mathsf{nd}})} i : N$ with $\mathsf{dom}(\Delta) = \mathsf{nv}(i)$ and $\mathsf{tight}(\Delta)$. Thus, we conclude that Φ satisfies the statement, in particular noting that $\mathsf{dom}(\Gamma) = \mathsf{dom}(\Pi) \cup \mathsf{dom}(\Delta) = (\mathsf{dom}(\Pi; x : N) \setminus \{x\}) \cup \mathsf{dom}(\Delta) = (\mathsf{nv}(q) \setminus \{x\}) \cup \mathsf{nv}(i) = \mathsf{nv}(q@[x \leftarrow i])$.

∗ Suppose Φ is of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} q : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} q@[x \leftarrow i] : M}\ \text{ES}_{\text{gc}}$$

But since $x \notin \mathsf{a}(q)$ then $\mathsf{tight}_\Gamma(\mathsf{a}(p))$ implies that $\mathsf{tight}_\Gamma(\mathsf{a}(q))$ and so we would be able to apply the *i.h.* on Ψ to obtain that $x \in \mathsf{nv}(q) = \mathsf{dom}(\Gamma)$. Thus, this case is absurd.

– Let $x \in \mathsf{a}(q)$. We proceed by case analysis on the last typing rule in Φ:

∗ Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m',e',r')} q : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} i : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} q@[x \leftarrow i] : M}\ \text{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. We proceed by case analysis on the shape of i:

· Let $i = y \in \mathsf{Var}$. Note that then $|i|_{\mathsf{nd}} = 0$ and $\mathsf{a}(p) = (\mathsf{a}(q) \setminus \{x\}) \cup \{y\}$, and so $\mathsf{tight}_\Gamma(\mathsf{a}(p))$ implies that $\Delta(y) \in \mathsf{Tight}$. This means that $\Theta \rhd_U y : N \vdash^{(0,0,|t|_{\mathsf{nd}})} y : N$, with $N \in \mathsf{Tight}$. Thus, we have that $\mathsf{tight}_\Gamma(\mathsf{a}(p))$ implies that $\mathsf{tight}_{\Pi;x:N}(\mathsf{a}(q))$, and we can apply the *i.h.* on Ψ to obtain that $\Psi \rhd_U \Pi; x : N \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M$, $\mathsf{dom}(\Pi; x : N) = \mathsf{nv}(q)$, $\mathsf{tight}(\Pi; x : N)$ and $M \in \mathsf{Inert}$. Thus, we conclude that Φ satisfies the statement, in particular noting that $\mathsf{dom}(\Gamma) = \mathsf{dom}(\Pi) \cup \mathsf{dom}(\Delta) = (\mathsf{dom}(\Pi; x : N) \setminus \{x\}) \cup \mathsf{dom}(\Delta) = (\mathsf{nv}(q) \setminus \{x\}) \cup \mathsf{nv}(i) = \mathsf{nv}(q@[x \leftarrow i])$.

· Let $i \notin \mathsf{Var}$. Note that then $\mathsf{a}(p) = (\mathsf{a}(q) \setminus \{x\}) \cup \mathsf{a}(i)$, meaning that $\mathsf{tight}_\Gamma(\mathsf{a}(p))$ implies that $\mathsf{tight}_\Delta(\mathsf{a}(i))$. Thus, we can apply the *i.h.* on Θ to obtain that $\Theta \rhd_U \Delta \vdash^{(0,0,|i|_{\mathsf{nd}})} i : N$, $\mathsf{dom}(\Delta) = \mathsf{nv}(i)$, $\mathsf{tight}(\Delta)$, and $N \in \mathsf{Inert}$. The last conclusion allows us to infer from $\mathsf{tight}_\Gamma(\mathsf{a}(p))$ that $\mathsf{tight}_{\Pi;x:N}(\mathsf{a}(q))$. Thus, we can apply the *i.h.* on Ψ to obtain that $\Psi \rhd_U \Pi; x : N \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M$, $\mathsf{dom}(\Pi; x : N) = \mathsf{nv}(q)$, $\mathsf{tight}(\Pi; x : N)$ and $M \in \mathsf{Inert}$.

∗ Suppose Φ is of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} q : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} q@[x \leftarrow i] : M}\ \text{ES}_{\text{gc}}$$

But since $x \notin \mathsf{dom}(\Gamma)$ then $\mathsf{tight}_\Gamma(\mathsf{a}(p))$ implies that $\mathsf{tight}_\Gamma(\mathsf{a}(q))$ and so we would be able to apply the *i.h.* on Ψ to obtain that $x \in \mathsf{nv}(q) = \mathsf{dom}(\Gamma)$. Thus, this case is absurd.

- Let $\mathsf{uinert}(p)$ be derived as

$$\frac{\mathsf{genVar}_x(q)}{\mathsf{uinert}(q@[x{\leftarrow}i^+])}\ \mathsf{I_{GV}}$$

with $q = q@[x{\leftarrow}i^+]$. Note that $\mathsf{nv}(q) = \{x\}$—by Lemma 8.2.4 (Properties of generalized variables)—and so $\mathsf{nv}(p) = \mathsf{nv}(i)$ and $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |i|_{\mathsf{nd}}$. Moreover, note that $\mathsf{a}(p) = (\mathsf{a}(q) \setminus \{x\}) \cup \mathsf{a}(t)$. We proceed by case analysis on the last typing rule in Φ:

  - Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m',e',r')} q : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} i^+ : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} q@[x{\leftarrow}i^+] : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. Since $\mathsf{tight}_\Gamma(\mathsf{a}(p))$ implies that $\mathsf{tight}_\Delta(\mathsf{a}(i^+))$, we can apply Lemma 9.1.3 (Typing properties of useful inert terms) to obtain that $\Theta \rhd_U \Delta \vdash^{(0,0,|i^+|_{\mathsf{nd}})} i^+ : N$, $\mathsf{dom}(\Delta) = \mathsf{nv}(i^+)$, $\mathsf{tight}(\Delta)$, and $N \in \mathsf{Inert}$. Application of Lemma 9.1.4 (Typing properties of generalized variables) on Ψ thus gives us that $\Psi \rhd_U x : M \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M$, with $M \in \mathsf{Inert}$. Therefore, the statement holds by noting that Φ is of the form

$$\frac{\Psi \rhd_U x : M \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M \quad \Theta \rhd_U \Delta \vdash^{(0,0,|i^+|_{\mathsf{nd}})} i^+ : M}{\Delta \vdash^{(0,0,|q|_{\mathsf{nd}}+|i^+|_{\mathsf{nd}})} q@[x{\leftarrow}i^+] : M}\ \mathrm{ES}$$

  - Suppose Φ is of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} q : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} q@[x{\leftarrow}i^+] : M}\ \mathrm{ES_{gc}}$$

But then Lemma 9.1.4 (Typing properties of generalized variables) on Ψ gives us that $\mathsf{dom}(\Gamma) = \{x\}$. Hence, this case is absurd.

- Let $\mathsf{uinert}(p)$ be derived as

$$\frac{\mathsf{uinert}(q) \quad x \in \mathsf{ul}\,(q)}{\mathsf{uinert}(q@[x{\leftarrow}v])}\ \mathsf{I_U}$$

with $q = q@[x{\leftarrow}v]$. Note that $\mathsf{nv}(p) = (\mathsf{nv}(q) \setminus \{x\}) \cup \mathsf{nv}(v)$ and $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |t|_{\mathsf{nd}}$. We proceed by case analysis on the last typing rule in Φ:

  - Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m',e',r')} q : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} v : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} q@[x{\leftarrow}v] : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. Note that since $x \notin \mathsf{a}(q)$ then $\mathsf{tight}_\Gamma(\mathsf{a}(p))$ implies that $\mathsf{tight}_{\Pi;x:N}(\mathsf{a}(q))$. Hence, we can apply the *i.h.* on Ψ to obtain that $\Psi \rhd_U \Pi; x : N \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M$, $\mathsf{dom}(\Pi; x : N) = \mathsf{nv}(q)$, $\mathsf{tight}(\Pi; x : N)$ and $M \in \mathsf{Inert}$. Moreover, since we have that $N \in \mathsf{Tight}$, we can apply Lemma 9.1.2 (Typing properties of values) to obtain that $\Theta \rhd_U \Delta \vdash^{(0,0,|v|_{\mathsf{nd}})} v : N$ and $\mathsf{dom}(\Delta) = \mathsf{nv}(v)$. The statement clearly holds.

– Suppose Φ is of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} q : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} q@[x{\leftarrow}v] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

But then $\mathsf{tight}_\Gamma(\mathsf{a}(p))$ together with the fact that $x \notin \mathsf{dom}(\Gamma)$ (or that $x \notin \mathsf{a}(q)$) imply that $\mathsf{tight}_\Gamma(\mathsf{a}(q))$. Thus, we would be able to apply the *i.h.* on Ψ to obtain that $x \in \mathsf{nv}(q) = \mathsf{dom}(\Gamma)$. This makes this case absurd.

- Let $\mathsf{uinert}(p)$ be derived as

$$\frac{\mathsf{uinert}(q) \quad x \notin \mathsf{nv}(q)}{\mathsf{uinert}(q@[x{\leftarrow}t])}\ \mathsf{I}_{\mathsf{GC}}$$

with $q = q@[x{\leftarrow}t]$. Note that $\mathsf{nv}(p) = \mathsf{nv}(q)$, $\mathsf{a}(p) = \mathsf{a}(q)$, and $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}}$. We proceed by case analysis on the last typing rule in Φ:

– Suppose Φ is of the form

$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m',e',r')} q : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} q@[x{\leftarrow}t] : M}\ \mathrm{ES}$$

But then we would be able to infer from $\mathsf{tight}_\Gamma(\mathsf{a}(p))$ that $\mathsf{tight}_{\Pi;x:N}(\mathsf{a}(q))$, since $x \notin \mathsf{a}(q)$. Thus, we would be able to apply the *i.h.* on Ψ to obtain that $\mathsf{nv}(q) = \mathsf{dom}(\Pi; x : N) = \mathsf{dom}(\Pi) \cup \{x\}$. But $x \notin \mathsf{nv}(q)$, and so this case is absurd.

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} q : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} q@[x{\leftarrow}t] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

Since $\mathsf{a}(p) = \mathsf{a}(q)$, we can apply the *i.h.* on Ψ to get that $(m, e, r) = (0, 0, |q|_{\mathsf{nd}})$, $\mathsf{dom}(\Gamma) = \mathsf{nv}(q) = \mathsf{nv}(p)$, $\mathsf{tight}(\Gamma)$ and $M \in \mathsf{Inert}$.

□

□

(Click here to go back to main chapter.)

Proposition 13.6.7 (Typing properties of Useful Open CbNeed-normal forms)**.**
Let $p \in \mathcal{PR}$ be such that $\mathsf{unorm}(p)$, and let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} p : M$ be a tight type derivation for it. Then $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$ and $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.

Proof. (Click here to go back to main chapter.)

Let p be in $\to_{\mathsf{und}}$-normal form. We proceed by case analysis according to what is given in Proposition 8.2.7 (Syntactic characterization of Useful Open CbNeed-normal forms):

- If $\mathsf{genVar}_{\#}(p)$, then the statement follows from Lemma 9.1.4 (Typing properties of generalized variables).
- If $\mathsf{uabs}(p)$, then the statement follows from Lemma 9.1.5 (Typing properties of useful abstraction programs) on Φ.
- Let $\mathsf{uinert}(p)$. Since $\mathsf{tight}(\Gamma)$ implies that $\mathsf{tight}_\Gamma(\mathsf{a}(p))$, then the statement follows from Lemma 9.1.6 (Typing properties of useful inert programs).

□

(Click here to go back to main chapter.)

Linear Substitution for Useful Open CbNeed. Proving Linear Substitution for the Useful Open CbNeed case requires adapting the analysis of plugged variables in the Open CbNeed case—see Lemma 13.4.4 (Plugged variables and domain of type contexts) in Chapter 7 (Multi types for Open CbNeed)—to the Useful Open CbNeed case. This is achieved simply by switching from checking the tightness of *needed* variables of the subject expression, to only checking the tightness of its *applied* variables, as follows:

Lemma 13.6.8 (Plugged variables and domain of type contexts)**.**

1. *Let $\mathcal{H}$ be a term context such that $x \notin \mathsf{a}(\mathcal{H})$, and let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} \mathcal{H}\langle x\rangle : M$ be such that $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}))$.*
   Then $x \in \mathsf{dom}(\Gamma)$. Moreover, if $\mathcal{H}$ is applicative then $\Gamma(x) \notin \mathsf{Abs}$.
2. *Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be such that $x \notin \mathcal{A}$ and $x \notin \mathsf{dom}(P)$, and let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} P\langle x\rangle : M$ be such that $\mathsf{tight}_\Gamma(\mathcal{A})$.*
   Then $x \in \mathsf{dom}(\Gamma)$. Moreover, if $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ then $\Gamma(x) \notin \mathsf{Abs}$.

Proof.

1. By structural induction on $\mathcal{H}$.
   - Let $\mathcal{H} \coloneqq \langle\cdot\rangle$ be a term context. Then $x \in \mathsf{dom}(\Gamma)$, because Φ is either an ax or an ax_T rule. Since this $\langle\cdot\rangle$ is not an applicative term context, the moreover part holds trivially.
   - Let $\mathcal{H} \coloneqq \mathcal{J}t$. Note that $\mathsf{a}(\mathcal{H}) = \mathsf{a}(\mathcal{J})$. Since all three typing rules app, app_i and $\mathsf{app}_{\mathrm{gc}}$ are proven rather similarly, we only proceed to prove the statement for rule app:
     Let Φ be of the form
     $$\frac{\Psi \rhd_U \Pi \vdash^{(m',e',r')} \mathcal{J}\langle x\rangle : [N \multimap M] \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m''+1,e'+e'',r'+r'')} \mathcal{J}\langle x\rangle t : M}\ \mathsf{app}$$
     with $\Gamma = \Pi \biguplus \Delta$. Since $\Pi \subseteq \Gamma$, then $\mathsf{tight}_\Pi(\mathsf{a}(\mathcal{J}))$ and we can apply the *i.h.* on Ψ to obtain that $x \in \mathsf{dom}(\Pi) \subseteq \mathsf{dom}(\Gamma)$.
     Let $\mathcal{H}$ be an applicative term context. The moreover part is proven by considering two sub-cases:
     - Let $\mathcal{J} \coloneqq \langle\cdot\rangle$. Then $\Psi \rhd_U x : [N \multimap M] \vdash^{(0,1,0)} x : [N \multimap M]$, and so $\Gamma(x) = [N \multimap M] \uplus \Delta(x) \notin \mathsf{Abs}$.
     - Let $\mathcal{J}$ be itself applicative. Then the *i.h.* on Ψ yields that $\Pi(x) \notin \mathsf{Abs}$, in turn implying that $\Gamma(x) \notin \mathsf{Abs}$.
   - Let $\mathcal{H} \coloneqq i\mathcal{J}$. Note that $\mathsf{a}(\mathcal{H}) = \mathsf{a}(i) \cup \mathsf{a}(\mathcal{J})$. We proceed by case analysis on the last typing rule in Φ.
     - Suppose Φ is of the form
       $$\frac{\Psi \rhd_U \Pi \vdash^{(m',e',r')} i : [N \multimap M] \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} \mathcal{J}\langle x\rangle : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m''+1,e'+e'',r'+r'')} i\mathcal{J}\langle x\rangle : M}\ \mathsf{app}$$
       with $\Gamma = \Pi \biguplus \Delta$. However, note that $i \neq y \in \mathsf{Var}$ or else we could infer that $\Pi = \{y : [N \multimap M]\}$, contradicting the assumption that $\mathsf{tight}(\Pi \biguplus \Delta)$. But if $i \notin \mathsf{Var}$, and since $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}))$ implies that $\mathsf{tight}_\Gamma(\mathsf{a}(i))$, then we would be able to apply Lemma 9.1.3 (Typing properties of useful inert terms) on Ψ to obtain that $[N \multimap M] \in \mathsf{Inert}$—which is also absurd. Therefore, this case is impossible.
     - The case where $\mathsf{app}_{\mathrm{gc}}$ is the last typing rule in Φ is ruled out as we did for the previous case.

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi \vdash^{(m',e',r')} i : [\mathsf{inert}]_{j\in J} \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} \mathcal{J}\langle x\rangle : [\mathsf{tight}] \quad J \neq \emptyset}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r''+1)} i\mathcal{J}\langle x\rangle : [\mathsf{inert}]_{j\in J}}\ \mathsf{app}_i$$

with $\Gamma = \Pi \biguplus \Delta$. Since $\Delta \subseteq (\Pi \biguplus \Delta)$, then we can apply the *i.h.* on Θ to obtain that $x \in \mathsf{dom}(\Delta) \subseteq \mathsf{dom}(\Gamma)$.
Moreover, if $\mathcal{H}$ is applicative then so is $\mathcal{J}$ and then $\Delta(x) \notin \mathsf{Abs}$ by *i.h.* on Θ, implying that $\Gamma(x) \notin \mathsf{Abs}$.

2. By induction on the derivation of $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$.
   - Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{}{(\mathcal{H},\epsilon) \in \mathcal{E}_{\mathsf{a}(\mathcal{H}),\mathsf{u}(\mathcal{H})}}\ \mathsf{M_{AX}}$$

   with $P = (\mathcal{H},\epsilon)$. Note that $\mathcal{A} = \mathsf{a}(\mathcal{H})$. Then Φ is of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} \mathcal{H}\langle x\rangle : M}{\Gamma \vdash^{(m,e,r)} (\mathcal{H}\langle x\rangle,\epsilon) : M}\ \mathsf{Lift}$$

   and the statement is proven by Lemma 13.6.8.1 (Plugged variables and domain of type contexts) on Ψ.
   - Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad z \in (\mathcal{V}\cup\mathcal{B})}{Q@[z{\leftarrow}y] \in \mathcal{E}_{\mathsf{upd}(\mathcal{V},z,y),\mathsf{upd}(\mathcal{B},z,y)}}\ \mathsf{M_{VAR}}$$

   with $P = Q@[z{\leftarrow}y]$, $\mathcal{U} = \mathsf{upd}(\mathcal{V},z,y)$ and $\mathcal{A} = \mathsf{upd}(\mathcal{B},z,y)$. Note that $x \neq z$, because $x \notin \mathsf{dom}(P)$. We do case analysis on the last typing rule in Φ.
     – Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; z : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \dfrac{}{\Theta \rhd_U y : N \vdash^{(m'',e'',r'')} y : N}\ \{\mathsf{ax},\mathsf{ax}_T\} \quad N \neq \mathbf{0}}{\Pi \biguplus \{y : N\} \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle@[z{\leftarrow}y] : M}\ \mathrm{ES}$$

     There are two cases concerning z and $\mathcal{B}$. On the one hand, if $z \notin \mathcal{B}$, then we can apply the *i.h.* on Ψ to conclude that $x \in \mathsf{dom}(\Pi) \subseteq \mathsf{dom}(\Pi \biguplus \{y : N\})$. On the other hand, if $z \in \mathcal{B}$, then $\mathcal{A} = \mathsf{upd}(\mathcal{B},z,y) = (\mathcal{B}\setminus\{z\})\cup\{y\}$, and so $N \in \mathsf{Tight}$ by hypothesis. Hence, we can apply the *i.h.* on Ψ to conclude that $x \in \mathsf{dom}(\Pi) \subseteq \mathsf{dom}(\Gamma)$.
     In either case, the moreover part holds by the fact that $x \neq z$: if $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$, then $Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$, so $\Pi(x) \notin \mathsf{Abs}$ by *i.h.*, and then $\Gamma(x) \notin \mathsf{Abs}$.
     – Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle x\rangle : M \quad \Gamma(z) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle x\rangle@[z{\leftarrow}y] : M}\ \mathrm{ES_{gc}}$$

     Note that $\mathsf{tight}_\Gamma(\mathcal{B})$, and so we can conclude that $x \in \mathsf{dom}(\Gamma)$ by *i.h.* on Ψ. The moreover part holds by *i.h.* as well.
   - Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad z \in (\mathcal{V}\cup\mathcal{B})}{Q@[z{\leftarrow}i^+] \in \mathcal{E}_{(\mathcal{V}\setminus\{z\})\cup\mathsf{u}(i^+),(\mathcal{B}\setminus\{z\})\cup\mathsf{a}(i^+)}}\ \mathsf{M_I}$$

   with $P = Q@[z{\leftarrow}i^+]$, $\mathcal{U} = (\mathcal{V}\setminus\{z\})\cup\mathsf{u}(i^+)$ and $\mathcal{A} = (\mathcal{B}\setminus\{z\})\cup\mathsf{a}(i^+)$. Note that $x \neq z$ because $x \notin \mathsf{dom}(P)$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; z : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} i^+ : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle@[z\leftarrow i^+] : M}\ \mathrm{ES}$$

There are two cases concerning z and $\mathcal{B}$. On the one hand, if $z \notin \mathcal{B}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi;z:N}(\mathcal{B})$, thus allowing us to apply the *i.h.* on Ψ to conclude that $x \in \mathsf{dom}(\Pi) \subseteq \mathsf{dom}(\Gamma)$. On the other hand, if $z \in \mathcal{B}$, and since $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Delta(\mathsf{a}(t))$, then we can obtain that $N \in \mathsf{Inert}$—by Lemma 9.1.3 (Typing properties of useful inert terms). Therefore, $\mathsf{tight}_{\Pi;z:N}(\mathcal{B})$, and so we are able to apply the *i.h.* on Ψ to obtain that $x \in \mathsf{dom}(\Pi) \subseteq \mathsf{dom}(\Gamma)$. The moreover part holds by *i.h.* on Ψ.

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle x\rangle : M \quad \Gamma(z) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle x\rangle@[z\leftarrow i^+] : M}\ \mathrm{ES_{gc}}$$

Since $z \notin \mathsf{dom}(\Gamma)$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathcal{B})$. By *i.h.* on Ψ, we conclude that $x \in \mathsf{dom}(\Gamma)$. The moreover part also holds by *i.h.* on Ψ.

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{U},\mathcal{A}} \quad x \notin (\mathcal{U} \cup \mathcal{A})}{Q@[z\leftarrow t] \in \mathcal{E}_{\mathcal{U},\mathcal{A}}}\ \mathsf{M_{GC}}$$

with $P = Q@[z\leftarrow t]$. Note that $x \neq z$ because $x \notin \mathsf{dom}(P)$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; z : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle@[z\leftarrow t] : M}\ \mathrm{ES}$$

Note that since $z \notin \mathcal{B}$, then $\mathsf{tight}_{\Pi \uplus \Delta}(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi;z:N}(\mathcal{B})$. We are thus allowed to apply the *i.h.* on Ψ to prove the statement, concluding that $x \in \mathsf{dom}(\Pi) \subseteq \mathsf{dom}(\Gamma)$. The moreover part follows from applying the *i.h.* on Ψ as well.

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle x\rangle : M \quad \Gamma(z) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle x\rangle@[z\leftarrow t] : M}\ \mathrm{ES_{gc}}$$

The statement holds by *i.h.* on Ψ.

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad x \in (\mathcal{V} \setminus \mathcal{B})}{Q@[z\leftarrow v] \in \mathcal{E}_{\mathcal{V}\setminus\{x\},\mathcal{B}}}\ \mathsf{M_U}$$

with $P = Q@[z\leftarrow v]$, $\mathcal{U} = \mathcal{V} \setminus \{z\}$, $\mathcal{A} = \mathcal{B}$, and $x \neq z$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be derived as

$$\frac{\Psi \rhd_U \Pi; z : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} v : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle@[z\leftarrow v] : M}\ \mathrm{ES}$$

Note that since $z \notin \mathcal{B}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi;z:N}(\mathcal{B})$. Thus, we can apply the *i.h.* on Ψ to obtain that $x \in \mathsf{dom}(\Pi) \subseteq \mathsf{dom}(\Gamma)$. The moreover part follows from applying the *i.h.* on Ψ as well.

– Let Φ be derived as

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle x\rangle : M \quad \Gamma(z) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle x\rangle@[z{\leftarrow}v] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

The statement holds by *i.h.* on Ψ.

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad z \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle z\rangle@[z{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{V}\cup \mathsf{u}(\mathcal{H}),\mathcal{B}\cup \mathsf{a}(\mathcal{H})}}\ \mathsf{M_{HER}}$$

with $P = Q\langle z\rangle@[z{\leftarrow}\mathcal{H}]$, $\mathcal{U} = \mathcal{V} \cup \mathsf{u}(\mathcal{H})$, $\mathcal{A} = \mathcal{B} \cup \mathsf{a}(\mathcal{H})$, and $x \neq z$ —by the variable convention. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be derived as

$$\frac{\Psi \rhd_U \Pi; z : N \vdash^{(m',e',r')} Q\langle z\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} \mathcal{H}\langle x\rangle : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle z\rangle@[z{\leftarrow}\mathcal{H}\langle x\rangle] : M}\ \mathrm{ES}$$

Then by Lemma 13.6.8.1 (Plugged variables and domain of type contexts), we have that $x \in \mathsf{dom}(\Delta) \subseteq \mathsf{dom}(\Gamma)$. To prove the moreover part, we further split the analysis on the shape of $\mathcal{H}$:

∗ Let $\mathcal{H} \neq \langle\cdot\rangle$. Then $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ has been derived as follows

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad z \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle z\rangle@[z{\leftarrow}\mathcal{H}^{@}]}\ \mathsf{E_{AX_2}}$$

and the moreover part holds by the moreover part in Lemma 13.6.8.1 (Plugged variables and domain of type contexts).

∗ Let $\mathcal{H} = \langle\cdot\rangle$. Then $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ has been derived as follows

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad z \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle z\rangle@[z{\leftarrow}\langle\cdot\rangle]}\ \mathsf{E_{NL}}$$

and we can further refine the shape of Φ as follows

$$\frac{\Psi \rhd_U \Pi; z : N \vdash^{(m',e',r')} Q\langle z\rangle : M \quad \Theta \rhd_U x : N \vdash^{(m'',e'',r'')} x : N \quad N \neq \mathbf{0}}{\Pi \uplus \{x : N\} \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle z\rangle@[z{\leftarrow}x] : M}\ \mathrm{ES}$$

Thus, the moreover part holds by *i.h.*, noting that $(\Pi; z : N)(z) = N \notin \mathsf{Abs}$ implies that $(\Pi \uplus \{x : N\})(x) \notin \mathsf{Abs}$.

– Suppose Φ is of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle z\rangle : M \quad \Gamma(z) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle z\rangle@[z{\leftarrow}\mathcal{H}\langle x\rangle] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

However, $\mathsf{tight}_\Gamma(\mathcal{A})$ would imply that $\mathsf{tight}_\Gamma(\mathcal{B})$. Thus, we would be able to apply the *i.h.* on Ψ to conclude that $z \in \mathsf{dom}(\Gamma)$; this makes this case absurd.

□

Proof. (Click here to go back to main chapter.)

1. By induction on the derivation of $\mathcal{H}^{@}$:
   - Let $\mathcal{H} = (\langle\cdot\rangle\, t)$. We proceed by case analysis on the last typing rule in Φ:
     - Let Φ be derived as

$$\frac{\Xi \triangleright_U x : [O \multimap N] \vdash^{(0,1,0)} x : [O \multimap N] \quad \Omega \triangleright_U \Delta \vdash^{(m'',e'',r'')} t : O \quad O \neq \mathbf{0}}{(\Delta \setminus\!\setminus \{x\}); x : ([O \multimap N] \uplus \Delta(x)) \vdash^{(m''+1,e''+1,r'')} x\, t : N}\ \mathsf{app}$$

       with $\Gamma = (\Delta \setminus\!\setminus \{x\})$ and $M = [O \multimap N] \uplus \Delta(x)$ and $(m, e, r) = (m'' + 1, e'' + 1, r'')$. We then propose splitting M into $M_1 \coloneqq [O \multimap N]$ and $M_2 \coloneqq \Delta(x)$. Now, for every $\Psi \triangleright_U \Pi \vdash^{(m',e',r')} v : M_1$, we can derive Θ as follows

$$\frac{\Psi \triangleright_U \Pi \vdash^{(m',e',r')} v : M_1 \quad \Xi \triangleright_U \Delta \vdash^{(m'',e'',r'')} t : O}{\Pi \biguplus \Delta \vdash^{(m'+m''+1,e'+e'',r'+r'')} v\, t : N}\ \mathsf{app}$$

       verifying that
       - $\Pi \biguplus \Delta = (\Pi \biguplus (\Delta \setminus\!\setminus \{x\})); x : \Delta(x)$, and
       - $(m' + m'' + 1, e' + e'', r' + r'') = (m + m', e + e' - 1, r + r')$.
     - It is easy to verify that if Φ has app_i as its last typing rule, then the statement holds trivially, as values are not inert-typable.
     - Let Φ be derived as

$$\frac{\dfrac{}{x : [\mathbf{0} \multimap N] \vdash^{(0,1,0)} x : [\mathbf{0} \multimap N]}\ \mathsf{ax}}{x : [\mathbf{0} \multimap N] \vdash^{(1,1,0)} x\, t : N}\ \mathsf{app}_{\mathrm{gc}}$$

       with $\Gamma = \emptyset$, $M = [\mathbf{0} \multimap N]$ and $(m, e, r) = (1, 1, 0)$. Let M be split into $M_1 \coloneqq [\mathbf{0} \multimap N]$ and $M_2 \coloneqq \mathbf{0}$. Now, for every $\Psi \triangleright_U \Pi \vdash^{(m',e',r')} v : M_1$, we can derive Θ as

$$\frac{\Psi \triangleright_U \Pi \vdash^{(m',e',r')} v : [\mathbf{0} \multimap N]}{\Pi \vdash^{(m'+1,e',r')} v\, t : N}\ \mathsf{app}_{\mathrm{gc}}$$

       verifying that
       - $\Pi = (\emptyset \biguplus \Pi); x : \mathbf{0} = (\Gamma \biguplus \Pi); x : M_2$, and
       - $(m' + 1, e' + 1, r') = (m + m', e + e' - 1, r + r')$.
   - Let $\mathcal{H}^{@} = \mathcal{J}^{@} t$. Case analysis on the last typing rule in Φ:
     - Let Φ be of the form

$$\frac{\Delta; x : M_\Delta \vdash^{(m_1'',e_1'',r_1'')} \mathcal{J}^{@}\langle x\rangle : [O \multimap N] \quad \Sigma; x : M_\Sigma \vdash^{(m_2'',e_2'',r_2'')} t : O \quad O \neq \mathbf{0}}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1''+m_2''+1,e_1''+e_2'',r_1''+r_2'')} \mathcal{H}^{@}\langle x\rangle t : N}\ \mathsf{app}$$

       where $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$ and

$$(m, e, r) = (m_1'' + m_2'' + 1, e_1'' + e_2'', r_1'' + r_2'')$$

       By applying the *i.h.* on the left-hand side premise, there exists a splitting $M_\Delta = M_{\Delta,1} \uplus M_{\Delta,2}$, with $M_{\Delta,1} \neq \mathbf{0}$, such that for every $\Psi \triangleright_U \Pi \vdash^{(m',e',r')} v : M_{\Delta,1}$ there exists

$$\Theta' \triangleright_U (\Delta \biguplus \Pi); x : M_{\Delta,2} \vdash^{(m_1''+m',e_1''+e'-1,r_1''+r')} \mathcal{J}^{@}\langle v\rangle : [O \multimap N]$$

We then propose splitting M in $M_1 \coloneqq M_{\Delta,1}$ and $M_2 \coloneqq M_{\Delta,2} \uplus M_\Sigma$, verifying that, for every such $\Psi \rhd_U \Pi \vdash^{(m',e',r')} v : M_{\Delta,1}$, we can derive Θ as follows

$$\frac{\Theta' \rhd_U (\Delta \biguplus \Pi); x : M_{\Delta,2} \vdash^{(m_1''+m',e_1''+e'-1,r_1''+r')} \mathcal{J}^{@}\langle v\rangle : [O \multimap N] \quad \Sigma; x : M_\Sigma \vdash^{(m_2'',e_2'',r_2'')} t : O}{(\Delta \biguplus \Pi \biguplus \Sigma); x : (M_{\Delta,2} \uplus M_\Sigma) \vdash^{(m_1''+m'+m_2''+1,e_1''+e'-1+e_2'',r_1''+r'+r_2'')} \mathcal{J}^{@}\langle v\rangle t : N}\ \mathsf{app}$$

In particular, note that

$$\begin{array}{rl} & (m_1'' + m' + m_2'' + 1, e_1'' + e' - 1 + e_2'', r_1'' + r' + r_2'') \\ = & ((m_1'' + m_2'' + 1) + m', (e_1'' + e_2'') + e' - 1, (r_1'' + r_2'') + r') \\ = & (m + m', e + e' - 1, r + r') \end{array}$$

– Let Φ be of the form

$$\frac{\Delta; x : M_\Delta \vdash^{(m_1'',e_1'',r_1'')} \mathcal{J}^{@}\langle x\rangle : [\mathsf{inert}]_{j\in J} \quad \Sigma; x : M_\Sigma \vdash^{(m_2'',e_2'',r_2'')} t : [\mathsf{tight}]}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1''+m_2''+1,e_1''+e_2'',r_1''+r_2'')} \mathcal{J}^{@}\langle x\rangle t : [\mathsf{inert}]_{j\in J}}\ \mathsf{app}_i$$

where $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$ and

$$(m, e, r) = (m_1'' + m_2'' + 1, e_1'' + e_2'', r_1'' + r_2'')$$

The statement holds by *i.h.* on the left-hand premise and then deriving Θ via the app_i rule.

– Let Φ be of the form

$$\frac{\Gamma; x : M \vdash^{(m,e,r)} \mathcal{J}^{@}\langle x\rangle : [\mathbf{0} \multimap N]}{\Gamma; x : M \vdash^{(m,e,r)} \mathcal{J}^{@}\langle x\rangle t : N}\ \mathsf{app}_{\mathrm{gc}}$$

The statement holds by *i.h.* on the premise and then deriving Θ via the $\mathsf{app}_{\mathrm{gc}}$ rule.

- Let $\mathcal{H}^{@} = i\mathcal{J}^{@}$. We consider the different cases corresponding to the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Phi_i \rhd_U \Delta; x : M_\Delta \vdash^{(m_1'',e_1'',r_1'')} i : [O \multimap N] \quad \Phi_{\mathcal{J}^{@}\langle x\rangle} \rhd_U \Sigma; x : M_\Sigma \vdash^{(m_2'',e_2'',r_2'')} \mathcal{J}^{@}\langle x\rangle : O}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1''+m_2''+1,e_1''+e_2'',r_1''+r_2'')} i\mathcal{J}^{@}\langle x\rangle : N}\ \mathsf{app}$$

where $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$ and

$$(m, e, r) = (m_1'' + m_2'' + 1, e_1'' + e_2'', r_1'' + r_2'')$$

By applying the *i.h.* on $\Phi_{\mathcal{J}^{@}\langle x\rangle}$, there exists a splitting $M_\Sigma = M_{\Sigma,1} \uplus M_{\Sigma,2}$, with $|M_{\Sigma,1}| \neq \mathbf{0}$, such that for every $\Psi \rhd_U \Pi \vdash^{(m',e',r')} v : M_{\Sigma,1}$ there exists

$$\Phi_{\mathcal{J}^{@}\langle v\rangle} \rhd_U \Sigma; x : M_{\Sigma,2} \vdash^{(m_2''+m',e_2''+e'-1,r_2'')} \mathcal{J}\langle v\rangle : O$$

Let M be split in $M_1 \coloneqq M_{\Sigma,1}$ and $M_2 \coloneqq M_\Delta \uplus M_{\Sigma,2}$. We then derive Θ as

$$\frac{\Delta; x : M_\Delta \vdash^{(m_1'',e_1'',r_1'')} i : [O \multimap N] \quad \Phi_{\mathcal{J}^{@}\langle v\rangle} \rhd_U \Sigma; x : M_{\Sigma,2} \vdash^{(m_2''+m',e_2''+e'-1,r_2'')} \mathcal{J}^{@}\langle v\rangle : O}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_{\Sigma,2}) \vdash^{(m_1''+m_2''+m'+1,e_1''+e_2''+e'-1,r_1''+r_2''+r')} i\mathcal{J}^{@}\langle v\rangle : N}\ \mathsf{app}$$

noting that

$$\begin{array}{rl} & (m_1'' + m_2'' + m' + 1, e_1'' + e_2'' + e', r_1'' + r_2'' + r') \\ = & ((m_1'' + m_2'' + 1) + m', (e_1'' + e_2'') + e', (r_1'' + r_2'') + r') \\ = & (m + m', e + e' - 1, r + r') \end{array}$$

– Let Φ be of the form

$$\frac{\Phi_i \rhd_U \Delta; x : M_\Delta \vdash^{(m_1'', e_1'', r_1'')} i : [\mathsf{inert}]_{j \in J} \quad \Phi_{\mathcal{J}^{@}\langle x \rangle} \rhd_U \Sigma; x : M_\Sigma \vdash^{(m_2'', e_2'', r_2'')} \mathcal{J}^{@}\langle x \rangle : [\mathsf{tight}]}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1'' + m_2'' + 1, e_1'' + e_2'', r_1'' + r_2'' + 1)} i\mathcal{J}^{@}\langle x \rangle : [\mathsf{inert}]_{j \in J}}\ \mathsf{app}_i$$

where $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$ and

$$(m, e, r) = (m_1'' + m_2'', e_1'' + e_2'', r_1'' + r_2'' + 1)$$

The statement holds by *i.h.* on $\Phi_{\mathcal{J}^{@}\langle x \rangle}$ and then deriving Θ via the app_i rule.

– Suppose Φ is of the form

$$\frac{\Phi_i \rhd_U \Gamma; x : M \vdash^{(m'', e'', r'')} i : [\mathbf{0} \multimap N]}{\Gamma; x : M \vdash^{(m''+1, e'', r'')} i\mathcal{J}^{@}\langle x \rangle : N}\ \mathsf{app}_{\mathrm{gc}}$$

where $(m, e, r) = (m''+1, e'', r'')$. Note that $x \notin \mathsf{nv}(\mathcal{H}^{@}) \supseteq \mathsf{a}(i)$, and then $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}^{@}))$ implies that $\mathsf{tight}_\Gamma(\mathsf{a}(i))$. However, this implies —by Lemma 9.1.3 (Typing properties of useful inert terms) on Φ_i— that $[\mathbf{0} \multimap N] = \mathsf{Inert}$, making this case absurd.

2. By induction on the derivation of $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$:
   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{}{(\mathcal{H}^{@}, \epsilon) \in \mathcal{E}^{@}_{\mathsf{u}(\mathcal{H}^{@}), \mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_1}$$

with $\mathcal{U} = \mathsf{u}(\mathcal{H}^{@})$, $\mathcal{A} = \mathsf{a}(\mathcal{H}^{@})$. Then Φ is of the form

$$\frac{\Phi' \rhd_U \Gamma; x : M \vdash^{(m,e,r)} \mathcal{H}^{@}\langle x \rangle : N}{\Gamma; x : M \vdash^{(m,e,r)} (\mathcal{H}^{@}\langle x \rangle, \epsilon) : N}\ \mathsf{Lift}$$

and the statement holds by application of Lemma 9.1.8.1 (Linear Substitution for Useful Open CbNeed in applicative term contexts) on Φ' and then deriving Φ via the Lift rule.

   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle y \rangle @ [y \leftarrow \mathcal{H}^{@}] \in \mathcal{E}^{@}_{\mathcal{V} \cup \mathsf{u}(\mathcal{H}^{@}), \mathcal{B} \cup \mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_2}$$

with $P = Q\langle y \rangle @ [y \leftarrow \mathcal{H}^{@}]$, $\mathcal{U} = \mathcal{V} \cup \mathsf{u}(\mathcal{H}^{@})$, and $\mathcal{A} = \mathcal{B} \cup \mathsf{a}(\mathcal{H}^{@})$. Note that $x \neq y$ —by α-conversion. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be derived as

$$\frac{\Phi' \rhd_U \Delta; x : M_\Delta; y : O \vdash^{(m_1, e_1, r_1)} Q\langle y \rangle : N \quad \Phi'' \rhd_U \Sigma; x : M_\Sigma \vdash^{(m_2, e_2, r_2)} \mathcal{H}^{@}\langle x \rangle : O \quad O \neq \mathbf{0}}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1+m_2, e_1+e_2, r_1+r_2)} Q\langle y \rangle @ [y \leftarrow \mathcal{H}^{@}\langle x \rangle] : N}\ \mathrm{ES}$$

with $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$, and

$$(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$$

Note that $x \notin \mathcal{A}$ implies that $x \notin \mathsf{a}(\mathcal{H}^{@})$, and that $\mathsf{tight}_{\Gamma}(\mathcal{A})$ implies that $\mathsf{tight}_{\Sigma}(\mathsf{a}(\mathcal{H}^{@}))$. All this means in turn that $x \in \mathsf{dom}(\Sigma; x : M_{\Sigma})$; *i.e.*, $M_{\Sigma} \neq \mathbf{0}$—by Lemma 13.6.8.1 (Plugged variables and domain of type contexts). Therefore, we can apply Lemma 9.1.8.1 (Linear Substitution for Useful Open CbNeed in applicative term contexts) on Φ'' to obtain a splitting $M_{\Sigma} \coloneqq M_{\Sigma,1} \uplus M_{\Sigma,2}$, with $M_{\Sigma,1} \neq \mathbf{0}$ such that for every $\Phi \rhd_U \Pi \vdash^{(m',e',r')} v : M_{\Sigma,1}$ there exists

$$\Theta' \rhd_U (\Sigma \biguplus \Pi); x : M_{\Sigma,2} \vdash^{(m_2+m',e_2+e'-1,r_2+r')} \mathcal{H}^{@}\langle v \rangle : O$$

Consequently, we propose splitting M into $M_1 = M_{\Sigma,1}$ and $M_2 = M_{\Delta} \uplus M_{\Sigma,2}$, and can then derive Θ as follows

$$\frac{\Phi' \rhd_U \Delta; x : M_{\Delta}; y : O \vdash^{(m_1,e_1,r_1)} Q\langle y \rangle : N \quad \Theta' \rhd_U (\Sigma \biguplus \Pi); x : M_{\Sigma,2} \vdash^{(m_2+m',e_2+e'-1,r_2+r')} \mathcal{H}^{@}\langle v \rangle : O}{(\Delta \biguplus \Sigma \biguplus \Pi); x : (M_{\Delta} \uplus M_{\Sigma,2}) \vdash^{(m_1+m_2+m',e_1+e_2+e'-1,r_1+r_2+r')} Q\langle y \rangle @ [y \leftarrow \mathcal{H}^{@}\langle v \rangle] : N}\ \mathrm{ES}$$

verifying in particular that $(m_1 + m_2 + m', e_1 + e_2 + e' - 1, r_1 + r_2 + r') = (m + m', e + e' - 1, r + r')$.

– Suppose Φ is derived as

$$\frac{\Phi' \rhd_U \Gamma; x : M \vdash^{(m,e,r)} Q\langle y \rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle y \rangle @ [y \leftarrow \mathcal{H}^{@}\langle x \rangle] : N}\ \mathrm{ES}_{\mathrm{gc}}$$

However, since $y \notin \mathsf{dom}(Q)$ —by the variable convention and the way in which $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ has been derived— and $\mathsf{tight}_{\Gamma;x:M}(\mathcal{A})$, then we could apply Lemma 13.6.8.2 (Plugged variables and domain of type contexts) on Φ' to obtain that $y \in \mathsf{dom}(\Gamma; x : M)$. Therefore, this case is not possible.

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y \leftarrow z] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{V},y,z),\mathsf{upd}(\mathcal{B},y,z)}}\ \mathsf{E}_{\mathsf{VAR}}$$

with $P = Q@[y \leftarrow z]$, $\mathcal{U} = \mathsf{upd}(\mathcal{V}, y, z)$, and $\mathcal{A} = \mathsf{upd}(\mathcal{B}, y, z)$. Note that $x \neq y$, since $x \notin \mathsf{dom}(\Gamma)$ by hypothesis. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be derived as

$$\frac{\Phi' \rhd_U \Delta; x : M_{\Delta}; y : O \vdash^{(m_1,e_1,r_1)} P\langle x \rangle : N \quad \overline{z : O \vdash^{(m_2,e_2,r_2)} z : O} \quad O \neq \mathbf{0}}{(\Delta; x : M_{\Delta}) \biguplus \{z : O\} \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} P\langle x \rangle @ [y \leftarrow z] : N}\ \mathrm{ES}$$

with $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. To determine the shape of Γ, let us consider the shape of z:

∗ Let $z \neq x$. Then $\Gamma = (\Delta \biguplus \{z : O\})$ and $M = M_{\Delta}$. Note that if $y \in \mathcal{B}$ then $z \in \mathcal{A}$ and so $O \in \mathsf{Tight}$ by hypothesis. Thus, we can safely infer from $\mathsf{tight}_{\Gamma}(\mathcal{A})$ that $\mathsf{tight}_{(\Delta \biguplus \{y:O\})}(\mathcal{B})$. Hence, we can apply the *i.h.* on Φ' to obtain a splitting of M_{Δ} into $M_{\Delta,1} \neq \mathbf{0}$ and $M_{\Delta,2}$ such that for every $\Psi \rhd_U \Pi \vdash^{(m',e',r')} v : M_{\Delta,1}$ there exists

$$\Theta' \rhd_U (\Delta \biguplus \Pi); x : M_{\Delta,2}; y : O \vdash^{(m_1+m',e_1+e',r_1+r')} Q\langle v \rangle : N$$

We then propose splitting M into $M_1 \coloneqq M_{\Delta,1}$ and $M_2 \coloneqq M_{\Delta,2}$ and derive Θ for such Ψ as

$$\frac{\Theta' \rhd_U (\Delta \uplus \Pi); x : M_{\Delta,2}; y : O \vdash^{(m_1+m',e_1+e'-1,r_1+r')} Q\langle v\rangle : N \quad \overline{z : O \vdash^{(m_2,e_2,r_2)} z : O}}{(\Delta \uplus \Pi \uplus \{z : O\}); x : M_{\Delta,2} \vdash^{(m_1+m'+m_2,e_1+e'-1+e_2,r_1+r'+r_2)} P\langle v\rangle@[y{\leftarrow}z] : N}\ \mathrm{ES}$$

∗ Let $z = x$. Then $\Gamma = \Delta$ and $M = M_\Delta \uplus O$. Note that then $y \notin \mathcal{B}$—or otherwise it would be that $x \in \mathcal{A}$, contradicting the hypothesis. We can apply the *i.h.* on Φ' to obtain a splitting of M_Δ into $M_{\Delta,1} \neq \mathbf{0}$ and $M_{\Delta,2}$ such that for every $\Psi \rhd_U \Pi \vdash^{(m',e',r')} v : M_{\Delta,1}$ there exists

$$\Theta' \rhd_U (\Delta \biguplus \Pi); x : M_{\Delta,2}; y : O \vdash^{(m_1+m',e_1+e'-1,r_1+r')} Q\langle v\rangle : N$$

noting that $y \notin \mathsf{dom}(\Pi)$—by the variable convention and Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system). We then propose splitting M into $M_1 \coloneqq M_{\Delta,1}$ and $M_2 \coloneqq M_{\Delta,2} \uplus O$ and derive Θ for such Ψ as

$$\frac{\Theta' \rhd_U (\Delta \biguplus \Pi); x : M_{\Delta,2}; y : O \vdash^{(m_1+m',e_1+e'-1,r_1+r')} Q\langle v\rangle : N \quad \overline{z : O \vdash^{(m_2,e_2,r_2)} z : O}}{(\Delta \biguplus \Pi); x : (M_{\Delta,2} \uplus O) \vdash^{(m_1+m'+m_2,e_1+e'-1+e_2,r_1+r'+r_2)} P\langle v\rangle@[y{\leftarrow}z] : N}\ \mathrm{ES}$$

– Let Φ be derived as

$$\frac{\Phi' \rhd_U \Gamma; x : M \vdash^{(m,e,r)} Q\langle x\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle x\rangle@[y{\leftarrow}z] : N}\ \mathrm{ES}_{\mathrm{gc}}$$

Since $\mathsf{tight}_\Gamma(\mathcal{A})$ and $y \notin \mathsf{dom}(\Gamma)$ imply that $\mathsf{tight}_\Gamma(\mathcal{B})$, we can apply the *i.h.* on Φ' and prove the statement easily.

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}i^+] \in \mathcal{E}^{@}_{(\mathcal{V}\setminus\{y\})\cup\mathsf{u}(i^+),(\mathcal{B}\setminus\{y\})\cup\mathsf{a}(i^+)}}\ \mathsf{E_I}$$

with $P = Q@[y{\leftarrow}i^+]$, $\mathcal{U} = (\mathcal{V} \setminus \{y\}) \cup \mathsf{u}(i^+)$, and $\mathcal{A} = (\mathcal{B} \setminus \{y\}) \cup \mathsf{a}(i^+)$. Note that $x \neq y$, since $x \notin \mathsf{dom}(\Gamma)$. We proceed by case analysis on the last typing rule of Φ:

– Let Φ be derived as

$$\frac{\Phi' \rhd_U \Delta; x : M_\Delta; y : O \vdash^{(m_1,e_1,r_1)} Q\langle x\rangle : N \quad \Phi'' \rhd_U \Sigma; x : M_\Sigma \vdash^{(m_2,e_2,r_2)} i^+ : O \quad O \neq \mathbf{0}}{(\Delta \uplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle x\rangle@[y{\leftarrow}i^+] : N}\ \mathrm{ES}$$

with $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$, and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that $x \notin \mathcal{A}$, and so $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Sigma;x:M_\Sigma}(\mathsf{a}(i^+))$. Since $i^+ \notin \mathsf{Var}$ then we can apply Lemma 9.1.3 (Typing properties of useful inert terms) on Φ'' to obtain that $O \in \mathsf{Inert}$. Thus, we can infer that $\mathsf{tight}_{\Delta;x:M_\Delta;y:O}(\mathcal{B})$ due to $\mathsf{tight}_\Gamma(\mathcal{A})$. Next, we apply the *i.h.* on Φ' to obtain a splitting of M_Δ into $M_{\Delta,1} \neq \mathbf{0}$ and $M_{\Delta,2}$ such that for every $\Psi \rhd_U \Pi \vdash^{(m',e',r')} v : M_{\Delta,1}$ there exists

$$\Theta' \rhd_U (\Delta \biguplus \Pi); x : M_{\Delta,2}; y : O \vdash^{(m_1+m',e_1+e',r_1+r')} Q\langle v\rangle : N$$

We then propose splitting M into $M_1 \coloneqq M_{\Delta,1}$ and $M_2 \coloneqq M_{\Delta,2} \uplus M_\Sigma$, and are able to derive Θ as

$$\frac{\Theta' \qquad \Phi'' \rhd_U \Sigma; x : M_\Sigma \vdash^{(m_2,e_2,r_2)} i^+ : O}{(\Delta \biguplus \Pi \biguplus \Sigma); x : (M_{\Delta,2} \uplus M_\Sigma) \vdash^{(m_1+m'+m_2,e_1+e'-1+e_2,r_1+r'+r_2)} P\langle v\rangle@[y{\leftarrow}i^+] : N}\ \mathrm{ES}$$

– Let Φ be derived as

$$\frac{\Phi' \rhd_U \Gamma; x : M \vdash^{(m,e,r)} Q\langle x\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle x\rangle@[y{\leftarrow}i^+] : N}\ \mathrm{ES}_{\mathrm{gc}}$$

Since $\mathsf{tight}_\Gamma(\mathcal{A})$ and $y \notin \mathsf{dom}(\Gamma)$ imply that $\mathsf{tight}_\Gamma(\mathcal{B})$, we can apply the *i.h.* on Φ' and prove the statement easily.

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{Q@[y{\leftarrow}t] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E_{GC}}$$

with $P = Q@[y{\leftarrow}t]$. Note that $x \neq y$, since $x \notin \mathsf{dom}(\Gamma)$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be derived as

$$\frac{\Phi' \rhd_U \Delta; x : M_\Delta; y : O \vdash^{(m_1,e_1,r_1)} Q\langle x\rangle : N \quad \Phi'' \rhd_U \Sigma; x : M_\Sigma \vdash^{(m_2,e_2,r_2)} t : O \quad O \neq \mathbf{0}}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle x\rangle@[y{\leftarrow}t] : N}\ \mathrm{ES}$$

with $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$, and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$.
Since $y \notin \mathcal{A}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Delta;y:O}(\mathcal{A})$. Hence, we can apply the *i.h.* on Φ' to obtain a splitting of M_Δ into $M_{\Delta,1} \neq \mathbf{0}$ and $M_{\Delta,2}$ such that for every $\Psi \rhd_U \Pi \vdash^{(m',e',r')} v : M_{\Delta,1}$ there exists

$$\Theta' \rhd_U (\Delta \biguplus \Pi); x : M_{\Delta,2}; y : O \vdash^{(m_1+m',e_1+e',r_1+r')} Q\langle v\rangle : N$$

We then propose splitting M into $M_1 \coloneqq M_{\Delta,1}$ and $M_2 \coloneqq M_{\Delta,2} \uplus M_\Sigma$, and are able to derive Θ as

$$\frac{\Theta' \qquad \Phi'' \rhd_U \Sigma; x : M_\Sigma \vdash^{(m_2,e_2,r_2)} t : O}{(\Delta \biguplus \Pi \biguplus \Sigma); x : (M_{\Delta,2} \uplus M_\Sigma) \vdash^{(m_1+m'+m_2,e_1+e'-1+e_2,r_1+r'+r_2)} P\langle v\rangle@[y{\leftarrow}t] : N}\ \mathrm{ES}$$

– Let Φ be derived as

$$\frac{\Phi' \rhd_U \Gamma; x : M \vdash^{(m,e,r)} Q\langle x\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle x\rangle@[y{\leftarrow}t] : N}\ \mathrm{ES}_{\mathrm{gc}}$$

Since $\mathsf{tight}_\Gamma(\mathcal{A})$ and $y \notin \mathsf{dom}(\Gamma)$ imply that $\mathsf{tight}_\Gamma(\mathcal{B})$, we can apply the *i.h.* on Φ' and prove the statement easily.

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \setminus \mathcal{B})}{Q@[y{\leftarrow}w] \in \mathcal{E}^{@}_{\mathcal{V}\setminus\{y\},\mathcal{B}}}\ \mathsf{E_U}$$

with $P = Q@[y{\leftarrow}w]$, $\mathcal{U} = \mathcal{V} \setminus \{y\}$, and $\mathcal{A} = \mathcal{B}$. Note that $x \neq y$, since $x \notin \mathsf{dom}(\Gamma)$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be derived as

$$\frac{\Phi' \triangleright_U \Delta; x : M_\Delta; y : O \vdash^{(m_1,e_1,r_1)} Q\langle x\rangle : N \quad \Phi'' \triangleright_U \Sigma; x : M_\Sigma \vdash^{(m_2,e_2,r_2)} w : O \quad O \neq \mathbf{0}}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle x\rangle@[y{\leftarrow}w] : N}\ \text{ES}$$

with $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$, and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$.
Since $y \notin \mathcal{B}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Delta;y:O}(\mathcal{B})$. We can then apply the *i.h.* on Φ' to obtain a splitting of M_Δ into $M_{\Delta,1} \neq \mathbf{0}$ and $M_{\Delta,2}$ such that for every $\Psi \triangleright_U \Pi \vdash^{(m',e',r')} v : M_{\Delta,1}$ there exists

$$\Theta' \triangleright_U (\Delta \biguplus \Pi); x : M_{\Delta,2}; y : O \vdash^{(m_1+m',e_1+e'-1,r_1+r')} Q\langle v\rangle : N$$

We propose splitting M into $M_1 \coloneqq M_{\Delta,1}$ and $M_2 \coloneqq M_{\Delta,2} \uplus M_\Sigma$, and are then able to derive Θ as

$$\frac{\Theta' \triangleright_U (\Delta \biguplus \Pi); x : M_{\Delta,2}; y : O \vdash^{(m_1+m',e_1+e'-1,r_1+r')} Q\langle v\rangle : N \qquad \Phi''}{(\Delta \biguplus \Pi \biguplus \Sigma); x : (M_{\Delta,2} \uplus M_\Sigma) \vdash^{(m_1+m'+m_2,e_1+e'-1+e_2,r_1+r'+r_2)} Q\langle v\rangle@[y{\leftarrow}w] : N}\ \text{ES}$$

– Let Φ be derived as

$$\frac{\Phi' \triangleright_U \Gamma; x : M \vdash^{(m,e,r)} Q\langle x\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle x\rangle@[y{\leftarrow}w] : N}\ \text{ES}_{\text{gc}}$$

Since $\mathsf{tight}_\Gamma(\mathcal{B})$, we can apply the *i.h.* on Φ' and prove the statement easily.

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{Q\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E}_{\mathsf{NL}}$$

with $P = Q\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle]$. Note that we may assume that $x \neq y$—by α-conversion. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be derived as

$$\frac{\Phi' \triangleright_U \Delta; x : M_\Delta; y : M_\Sigma \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : N \quad \Phi'' \triangleright_U x : M_\Sigma \vdash^{(m_2,e_2,r_2)} x : M_\Sigma}{(\Delta \biguplus \Sigma); x : (M_\Delta \uplus M_\Sigma) \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle y\rangle@[y{\leftarrow}x] : N}\ \text{ES}$$

with $\Gamma = \Delta \biguplus \Sigma$, $M = M_\Delta \uplus M_\Sigma$, and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$.
Note that $\mathsf{tight}_\Gamma(\mathcal{A})$ and the fact that $x, y \notin \mathcal{A}$ imply that $\mathsf{tight}_{\Delta;x:M_\Delta;y:M_\Sigma}(\mathcal{A})$. Consequently, we can apply Lemma 13.6.8.2 (Plugged variables and domain of type contexts) on Φ' to obtain that $M_\Sigma \neq \mathbf{0}$ and, moreover, $M_\Sigma \notin \mathsf{Abs}$. Two sub-cases:

∗ Let $M_\Sigma \notin \mathsf{Inert}$. Then we can conclude that $M_\Sigma \notin \mathsf{Tight}$ and so Φ'' is of the form

$$\frac{M_\Sigma \notin \mathsf{Tight}}{x : M_\Sigma \vdash^{(0,1,0)} x : M_\Sigma}\ \mathsf{ax}$$

with $(m_2, e_2, r_2) = (0, 1, 0)$.
We propose splitting M into $M_1 \coloneqq M_\Sigma$ and $M_2 \coloneqq M_\Delta$, noting that for every $\Psi \triangleright_U \Pi \vdash^{(m',e',r')} v : M_{\Delta,1}$ we can derive Θ as

$$\frac{\Phi' \triangleright_U \Delta; x : M_\Delta; y : M_\Sigma \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : N \quad \Psi \triangleright_U \Pi \vdash^{(m',e',r')} v : M_{\Delta,1}}{(\Delta \biguplus \Pi); x : M_\Sigma \vdash^{(m_1+m',e_1+e',r_1+r')} Q\langle y\rangle@[y{\leftarrow}x] : N}\ \text{ES}$$

In particular, we verify that

$$\begin{array}{rcl}(m_1+m',e_1+e',r_1+r') & = & (0+m_1+m',1+e_1+e'-1,0+r_1+r')\\ & = & (m_2+m_1+m',e_2+e_1+e'-1,r_2+r_1+r')\\ & = & (m+m',e+e'-1,r+r')\end{array}$$

* Let $M_\Sigma \in \mathsf{Inert}$. We propose splitting M into $M_1 \coloneqq M_\Sigma$ and $M_2 \coloneqq M_\Delta$, noting that the statement holds trivially because values are not typable with types in Inert.

– Suppose Φ is of the form

$$\frac{\Phi' \rhd_U \Gamma; x:M \vdash^{(m,e,r)} Q\langle y\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x:M \vdash^{(m,e,r)} Q\langle y\rangle @ [y \leftarrow x] : N}\ \mathrm{ES_{gc}}$$

Note that $y \notin \mathsf{dom}(Q)$ —by the variable convention and the way in which $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ has been derived— and that $\mathsf{tight}_{\Gamma;x:M}(\mathcal{A})$ by hypothesis. However, we would then be able to apply Lemma 13.6.8 (Plugged variables and domain of type contexts) to obtain that $y \in \mathsf{dom}(\Gamma; x:M)$, making this case absurd.

□

(Click here to go back to main chapter.)

The following is required to apply Lemma 7.1.4 (Linear Substitution for Open CbNeed) in the proof of Proposition 7.1.6.2 (Quantitative Subject Reduction for Open CbNeed - exponential case) to obtain the right indices.

Lemma 13.6.9 (Splitting multi types of Open CbNeed type derivations)**.**

Let $v \in \mathsf{Val}$, $M \coloneqq N \uplus O$, and let $\Phi \rhd_U \Gamma \vdash^{(m,e)} v : M$ be a type derivation. Then there exist type derivations

$$\begin{array}{c}\Psi \rhd_U \Pi \vdash^{(m',e')} v : N\\ \Theta \rhd_U \Delta \vdash^{(m'',e'')} v : O\end{array}$$

such that $\Gamma = \Pi \uplus \Delta$ and $(m,e) = (m'+m'', e'+e'')$.

Proof.

Trivial, given that the many typing rule is the only one that can derive a multi type on the right—*i.e.*, the derived type of the subject, in this case M—for values.

□

Lemma 13.6.10 (Quantitative Subject Reduction for $\to_{\mathsf{um}}$ in term contexts)**.**

Let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} \mathcal{H}\langle(\lambda x.u)s\rangle : M$ such that $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}))$. Then $m \geq 1$ and there exists $\Phi' \rhd_U \Gamma \vdash^{(m-1,e,r)} (\mathcal{H}\langle u\rangle, [x \leftarrow s]) : M$.

Proof. (Click here to go back to main chapter.)

By structural induction on $\mathcal{H}$:

- *Empty context*; *i.e.*, $\mathcal{H} = \langle\cdot\rangle$. We do case analysis on the last typing rule in Φ:
  – Let Φ be of the form

$$\dfrac{\dfrac{\dfrac{\Theta \rhd_U \Pi; x:N \vdash^{(m',e',r')} u : M}{\Pi \vdash^{(m',e',r')} \lambda x.u : N \multimap M}\ \mathsf{fun}}{\Pi \vdash^{(m',e',r')} \lambda x.u : [N \multimap M]}\ \mathsf{many} \qquad \Xi \rhd_U \Delta \vdash^{(m'',e'',r'')} s : N}{\Pi \uplus \Delta \vdash^{(m'+m''+1,e'+e'',r'+r'')} (\lambda x.u)s : M}\ \mathsf{app}$$

with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'' + 1, e' + e'', r' + r'')$. Note that $m \geq 1$. We can then derive Φ' as

$$\dfrac{\dfrac{\Theta \rhd_U \Pi; x : N \vdash^{(m',e',r')} u : M}{\Pi; x : N \vdash^{(m',e',r')} (u, \epsilon) : M}\ \mathsf{Lift} \qquad \Xi \rhd_U \Delta \vdash^{(m'',e'',r'')} s : N}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} (u, [x \leftarrow s]) : M}\ \mathrm{ES}$$

– app_i is ruled out as the last typing rule in Φ by the fact that values are not typed with inert.

– Let Φ be of the form

$$\dfrac{\dfrac{\dfrac{\Theta \rhd_U \Gamma \vdash^{(m',e',r')} u : M \quad x \notin \mathsf{dom}(\Gamma)}{\Gamma \vdash^{(m',e',r')} \lambda x.u : \mathbf{0} \multimap M}\ \mathsf{fun}}{\Gamma \vdash^{(m',e',r')} \lambda x.u : [\mathbf{0} \multimap M]}\ \mathsf{many}}{\Gamma \vdash^{(m'+1,e',r')} (\lambda x.u)s : M}\ \mathsf{app}_{\mathrm{gc}}$$

with $(m, e, r) = (m' + 1, e', r')$. Note that $m \geq 1$. We can then derive Φ' as

$$\dfrac{\dfrac{\Theta \rhd_U \Gamma \vdash^{(m',e',r')} u : M}{\Gamma \vdash^{(m',e',r')} (u, \epsilon) : M}\ \mathsf{Lift} \qquad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m',e',r')} (u, [x \leftarrow s]) : M}\ \mathrm{ES}_{\mathrm{gc}}$$

- *Application left*; *i.e.*, $\mathcal{H} = \mathcal{J}t$. We do case analysis on the last typing rule in Ψ:

  – Let Ψ be of the form

$$\dfrac{\Psi \rhd_U \Pi \vdash^{(m',e',r')} \mathcal{J}\langle(\lambda x.u)s\rangle : [N \multimap M] \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m''+1,e'+e'',r'+r'')} \mathcal{J}\langle(\lambda x.u)s\rangle t : M}\ \mathsf{app}$$

  with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'' + 1, e' + e'', r' + r'')$. Note that $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}))$ implies $\mathsf{tight}_\Pi(\mathsf{a}(\mathcal{J}))$. By *i.h.* on Ψ, we have that $m' \geq 1$ and there exists $\Psi' \rhd_U \Pi \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, [x \leftarrow s]) : [N \multimap M]$. We do case analysis on the last typing rule in Ψ':

  * Let Ψ' be of the form

$$\dfrac{\dfrac{\Omega \rhd_U \Pi_1; x : O \vdash^{(m'_1,e'_1,r'_1)} \mathcal{J}\langle u\rangle : [N \multimap M]}{\Pi_1; x : O \vdash^{(m'_1,e'_1,r'_1)} (\mathcal{J}\langle u\rangle, \epsilon) : [N \multimap M]}\ \mathsf{Lift} \qquad Z \rhd_U \Pi_2 \vdash^{(m'_2,e'_2,r'_2)} s : O \quad O \neq \mathbf{0}}{\Pi_1 \biguplus \Pi_2 \vdash^{(m'_1+m'_2,e'_1+e'_2,r'_1+r'_2)} (\mathcal{J}\langle u\rangle, [x \leftarrow s]) : [N \multimap M]}\ \mathrm{ES}$$

    with $\Pi = \Pi_1 \biguplus \Pi_2$ and $(m' - 1, e', r') = (m'_1 + m'_2, e'_1 + e'_2, r'_1 + r'_2)$. We can then derive Φ' as follows

$$\dfrac{\dfrac{\Omega \qquad \Theta}{(\Pi_1 \biguplus \Delta); x : O \vdash^{(m'_1+m''+1,e'_1+e'',r'_1+r'')} \mathcal{J}\langle u\rangle t : M}\ \mathsf{app} \qquad Z}{\Pi_1 \biguplus \Delta \biguplus \Pi_2 \vdash^{(m'_1+m''+1+m'_2,e'_1+e''+e'_2,r'_1+r''+r'_2)} (\mathcal{J}\langle u\rangle t, [x \leftarrow s]) : M}\ \mathrm{ES}$$

    noting that $\Pi_1 \biguplus \Delta \biguplus \Pi_2 = \Pi \biguplus \Delta = \Gamma$ and that

$$\begin{array}{rl} & (m'_1 + m'' + 1 + m'_2, e'_1 + e'' + e'_2, r'_1 + r'' + r'_2) \\ = & (m' - 1 + m'' + 1, e' + e'', r' + r'') \\ = & (m - 1, e, r) \end{array}$$

∗ Let Ψ' be of the form

$$\dfrac{\dfrac{\Omega \rhd_U \Pi \vdash^{(m'-1,e',r')} \mathcal{J}\langle u\rangle : [N \multimap M]}{\Pi \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, \epsilon) : [N \multimap M]}\ \mathsf{Lift} \qquad \Pi(x) = \mathbf{0}}{\Pi \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [N \multimap M]}\ \mathrm{ES}_{\mathrm{gc}}$$

We can then derive Φ' as follows

$$\dfrac{\dfrac{\Omega \qquad \Theta}{\Pi \uplus \Delta \vdash^{(m'-1+m''+1,e'+e'',r'+r'')} \mathcal{J}\langle u\rangle t : M}\ \mathsf{app} \qquad (\Pi \uplus \Delta)(x) = \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'-1+m''+1,e'+e'',r'+r'')} (\mathcal{J}\langle u\rangle t, [x{\leftarrow}s]) : M}\ \mathrm{ES}_{\mathrm{gc}}$$

noting that $(m' - 1 + m'' + 1, e' + e'', r' + r'') = (m - 1, e, r)$.

– Let Φ be of the form

$$\dfrac{\Psi \rhd_U \Pi \vdash^{(m',e',r')} \mathcal{J}\langle(\lambda x.u)s\rangle : [\mathsf{inert}]_{j\in J} \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : [\mathsf{tight}] \quad J \neq \emptyset}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r''+1)} \mathcal{J}\langle(\lambda x.u)s\rangle t : [\mathsf{inert}]_{j\in J}}\ \mathsf{app}_i$$

with $\Gamma = \Pi \uplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'' + 1)$. Note that $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}))$ implies $\mathsf{tight}_\Pi(\mathsf{a}(\mathcal{J}))$. Hence, by *i.h.* on Ψ, $m' \geq 1$ and there exists

$$\Psi' \rhd_U \Pi \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [\mathsf{inert}]_{j\in J}$$

We do case analysis on the last typing rule in Ψ':

∗ Let Ψ' be of the form

$$\dfrac{\dfrac{\Xi \rhd_U \Pi_1; x : O \vdash^{(m'_1,e'_2,r'_1)} \mathcal{J}\langle u\rangle : [\mathsf{inert}]_{j\in J}}{\Pi_1; x : O \vdash^{(m'_1,e'_2,r'_1)} (\mathcal{J}\langle u\rangle, \epsilon) : [\mathsf{inert}]_{j\in J}}\ \mathsf{Lift} \qquad \Omega \rhd_U \Pi_2 \vdash^{(m'_2,e'_2,r'_2)} s : O \quad O \neq \mathbf{0}}{\Pi_1 \uplus \Pi_2 \vdash^{(m'_1+m'_2,e'_1+e'_2,r'_1+r'_2)} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [\mathsf{inert}]_{j\in J}}\ \mathrm{ES}$$

with $\Pi = \Pi_1 \uplus \Pi_2$ and

$$(m' - 1, e', r') = (m'_1 + m'_2, e'_1 + e'_2, r'_1 + r'_2)$$

We can then derive Φ' as follows

$$\dfrac{\dfrac{\Xi \qquad \Theta}{(\Pi_1 \uplus \Delta) ; x : O \vdash^{(m'_1+m'',e'_1+e'',r'_1+r''+1)} \mathcal{J}\langle u\rangle t : \uplus_{j\in J} [\mathsf{inert}]}\ \mathsf{app}_i \qquad \Omega}{\Pi_1 \uplus \Delta \uplus \Pi_2 \vdash^{(m'_1+m''+m'_2,e'_1+e''+e'_2,r'_1+r''+1+r'_2)} (\mathcal{J}\langle u\rangle t, [x{\leftarrow}s]) : \uplus_{j\in J} [\mathsf{inert}]}\ \mathrm{ES}$$

Note that $\Pi_1 \uplus \Delta \uplus \Pi_2 = \Pi \uplus \Delta = \Gamma$ and that

$$\begin{array}{ll} & (m'_1 + m'' + m'_2, e'_1 + e'' + e'_2, r'_1 + r'' + 1 + r'_2) \\ = & (m' - 1 + m'', e' + e'', r' + r'' + 1) \\ = & (m - 1, e, r) \end{array}$$

∗ Let Ψ' be of the form

$$\dfrac{\dfrac{\Xi \rhd_U \Pi \vdash^{(m'-1,e',r')} \mathcal{J}\langle u\rangle : [\mathsf{inert}]_{j\in J}}{\Pi \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, \epsilon) : [\mathsf{inert}]_{j\in J}}\ \mathsf{Lift} \qquad \Pi(x) = \mathbf{0}}{\Pi \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [\mathsf{inert}]_{j\in J}}\ \mathrm{ES}_{\mathrm{gc}}$$

We can then derive Φ' as follows

$$\dfrac{\dfrac{\dfrac{\Xi \qquad \Theta}{\Pi \biguplus \Delta \vdash^{(m'-1+m'',e'+e'',r'+r''+1)} \mathcal{J}\langle u\rangle t : [\mathsf{inert}]_{j\in J}}\ \mathsf{app}_i}{\Pi \biguplus \Delta \vdash^{(m'-1+m'',e'+e'',r'+r''+1)} (\mathcal{J}\langle u\rangle t, \epsilon) : [\mathsf{inert}]_{j\in J}}\ \mathsf{Lift} \qquad (\Pi \biguplus \Delta)(x) = \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'-1+m'',e'+e'',r'+r''+1)} (\mathcal{J}\langle u\rangle t, [x{\leftarrow}s]) : [\mathsf{inert}]_{j\in J}}\ \mathrm{ES_{gc}}$$

noting that

$$(m' - 1 + m'', e' + e'', r' + r'' + 1) = (m - 1, e, r)$$

– Let Ψ be of the form

$$\dfrac{\Psi \triangleright_U \Gamma \vdash^{(m',e',r')} \mathcal{J}\langle(\lambda x.u)s\rangle : [\mathbf{0} \multimap M]}{\Gamma \vdash^{(m'+1,e',r')} \mathcal{J}\langle(\lambda x.u)s\rangle t : M}\ \mathsf{app}_{\mathrm{gc}}$$

with $(m, e, r) = (m' + 1, e', r')$. Since $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}))$ implies $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{J}))$, then, by *i.h.* on Ψ, $m' \geq 1$ and there exists

$$\Psi' \triangleright_U \Gamma \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [\mathbf{0} \multimap M]$$

We do case analysis on the last typing rule in Ψ':

∗ Let Ψ' be of the form

$$\dfrac{\dfrac{\Theta \triangleright_U \Pi; x : N \vdash^{(m'_1,e'_1,r'_1)} \mathcal{J}\langle u\rangle : [\mathbf{0} \multimap M]}{\Pi; x : N \vdash^{(m'_1,e'_1,r'_1)} (\mathcal{J}\langle u\rangle, \epsilon) : [\mathbf{0} \multimap M]}\ \mathsf{Lift} \qquad \Xi \triangleright_U \Delta \vdash^{(m'_2,e'_2,r'_2)} s : N}{\Pi \biguplus \Delta \vdash^{(m'_1+m'_2,e'_1+e'_2,r'_1+r'_2)} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [\mathbf{0} \multimap M]}\ \mathrm{ES}$$

with $\Pi \biguplus \Delta = \Gamma$ and $(m'_1 + m'_2, e'_1 + e'_2, r'_1 + r'_2) = (m' - 1, e', r')$. We can then build Φ' as

$$\dfrac{\dfrac{\Theta}{\Pi; x : N \vdash^{(m'_1+1,e'_1,r'_1)} \mathcal{J}\langle u\rangle t : M}\ \mathsf{app}_{\mathrm{gc}} \qquad \Xi}{\Pi \biguplus \Delta \vdash^{(m'_1+1+m'_2,e'_1+e'_2,r'_1+r'_2)} (\mathcal{J}\langle u\rangle t, [x{\leftarrow}s]) : M}\ \mathrm{ES}$$

noting that

$$(m'_1 + 1 + m'_2, e'_1 + e'_2, r'_1 + r'_2) = (m', e', r') = (m - 1, e, r)$$

∗ Let Ψ' be of the form

$$\dfrac{\dfrac{\Theta \triangleright_U \Gamma \vdash^{(m'-1,e',r')} \mathcal{J}\langle u\rangle : [\mathbf{0} \multimap M]}{\Gamma \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, \epsilon) : [\mathbf{0} \multimap M]}\ \mathsf{Lift} \qquad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m'-1,e',r')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [\mathbf{0} \multimap M]}\ \mathrm{ES_{gc}}$$

We can then build Φ' as

$$\dfrac{\dfrac{\Theta}{\Gamma \vdash^{(m',e',r')} \mathcal{J}\langle u\rangle t : M}\ \mathsf{app}_{\mathrm{gc}} \qquad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m',e',r')} (\mathcal{J}\langle u\rangle t, [x{\leftarrow}s]) : M}\ \mathrm{ES_{gc}}$$

noting that $(m', e', r') = (m - 1, e, r)$.

- *Application right*; *i.e.*, $\mathcal{H} = i\mathcal{J}$: We do case analysis on the last typing rule in Φ.
  - Suppose Φ is of the form

$$\frac{\Psi \rhd_U \Pi \vdash^{(m',e',r')} i : [N \multimap M] \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} \mathcal{J}\langle(\lambda x.u)s\rangle : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m''+1,e'+e'',r'+r'')} i\mathcal{J}\langle(\lambda x.u)s\rangle : M}\ \mathsf{app}$$

    with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'' + 1, e' + e'', r' + r'')$. Since $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}))$ implies that $\mathsf{tight}_\Gamma(\mathsf{a}(i))$, we would then be able to apply Lemma 9.1.3 (Typing properties of useful inert terms) on Ψ to conclude that $[\mathbf{0} \multimap M] \in \mathsf{Inert}$. This is absurd, and then so is this case.
  - Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi \vdash^{(m',e',r')} i : [\mathsf{inert}]_{j\in J} \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} \mathcal{J}\langle(\lambda x.u)s\rangle : [\mathsf{tight}]}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r''+1)} i\mathcal{J}\langle(\lambda x.u)s\rangle : [\mathsf{inert}]_{j\in J}}\ \mathsf{app}_i$$

    with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'' + 1)$. Note that $\mathsf{tight}_{\Pi\biguplus\Delta}(\mathsf{a}(\mathcal{H}))$ implies $\mathsf{tight}_\Delta(\mathsf{a}(\mathcal{J}))$. Hence, by *i.h.* on Θ, $m'' \geq 1$ (hence, $m \geq 1$) and there exists $\Theta' \rhd_U \Delta \vdash^{(m''-1,e'',r'')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [\mathsf{tight}]$. We do case analysis on the last typing rule in Θ':
    - Let Θ' be of the form

$$\frac{\dfrac{\Xi \rhd_U \Delta_1; x : O \vdash^{(m_1'',e_1'',r_1'')} \mathcal{J}\langle u\rangle : N}{\Delta_1; x : O \vdash^{(m_1'',e_1'',r_1'')} (\mathcal{J}\langle u\rangle, \epsilon) : N}\ \mathsf{Lift} \quad \Omega \rhd_U \Delta_2 \vdash^{(m_2'',e_2'',r_2'')} s : O}{\Delta_1 \biguplus \Delta_2 \vdash^{(m_1''+m_2'',e_1''+e_2'',r_1''+r_2'')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : N}\ \mathrm{ES}$$

      with $\Delta = \Delta_1 \biguplus \Delta_2$ and $(m'' - 1, e'', r'') = (m_1'' + m_2'', e_1'' + e_2'', r_1'' + r_2'')$. We can then derive Ψ' as follows

$$\frac{\dfrac{\dfrac{\Psi \qquad \Xi}{(\Pi \biguplus \Delta_1); x : O \vdash^{(m'+m_1'',e'+r_1'',r'+r_1''+1)} i\mathcal{J}\langle u\rangle : M}\ \mathsf{app}_i}{(\Pi \biguplus \Delta_1); x : O \vdash^{(m'+m_1'',e'+r_1'',r'+r_1''+1)} (i\mathcal{J}\langle u\rangle, \epsilon) : M}\ \mathsf{Lift} \quad \Omega}{\Pi \biguplus \Delta_1 \biguplus \Delta_2 \vdash^{(m'+m_1''+m_2'',e'+r_1''+e_2'',r'+r_1''+1+r_2'')} (i\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : M}\ \mathrm{ES}$$

      noting that $\Pi \biguplus \Delta_1 \biguplus \Delta_2 = \Pi \biguplus \Delta = \Gamma$ and that

$$\begin{array}{rl} & (m' + m_1'' + m_2'', e' + r_1'' + e_2'', r' + r_1'' + 1 + r_2'') \\ = & (m' + (m'' - 1), e' + e'', r' + r'' + 1) \\ = & (m - 1, e, r) \end{array}$$

    - Let Θ' be of the form

$$\frac{\dfrac{\Xi \rhd_U \Delta \vdash^{(m''-1,e'',r'')} \mathcal{J}\langle u\rangle : [\mathsf{tight}]}{\Delta \vdash^{(m''-1,e'',r'')} (\mathcal{J}\langle u\rangle, \epsilon) : [\mathsf{tight}]}\ \mathsf{Lift} \quad \Delta(x) = \mathbf{0}}{\Delta \vdash^{(m''-1,e'',r'')} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [\mathsf{tight}]}\ \mathrm{ES}_{\mathrm{gc}}$$

      We can then derive Φ' as follows

$$\frac{\dfrac{\dfrac{\Psi \qquad \Xi}{\Pi \biguplus \Delta \vdash^{(m'+m''-1,e'+e'',r'+r''+1)} i\mathcal{J}\langle u\rangle : M}\ \mathsf{app}_i}{\Pi \biguplus \Delta \vdash^{(m'+m''-1,e'+e'',r'+r''+1)} (i\mathcal{J}\langle u\rangle, \epsilon) : M}\ \mathsf{Lift} \quad (\Pi \biguplus \Delta)(x) = \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m''-1,e'+e'',r'+r''+1)} (i\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : M}\ \mathrm{ES}_{\mathrm{gc}}$$

noting that

$$(m' + m'' - 1, e' + e'', r' + r'' + 1) = (m - 1, e, r)$$

– Suppose Φ is of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m',e',r')} i : [\mathbf{0} \multimap M]}{\Gamma \vdash^{(m'+1,e',r')} i\mathcal{H}\langle(\lambda x.u)s\rangle : M}\ \mathsf{app}_{\mathrm{gc}}$$

with $(m, e, r) = (m' + 1, e', r')$. Since $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}))$ implies that $\mathsf{tight}_\Gamma(\mathsf{a}(i))$, we can apply Lemma 9.1.3 (Typing properties of useful inert terms) on Ψ to conclude that $[\mathbf{0} \multimap M] \in \mathsf{Inert}$. This is absurd, and then so is this case.

□

(Click here to go back to main chapter.)

Proposition 13.6.11 (Quantitative Subject Reduction for Useful Open CbNeed)**.**
Let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} p : M$ be a tight type derivation.

1. Multiplicative*: if $p \to_{\mathsf{um}} p'$, then $m \geq 1$ and there exists a type derivation $\Phi' \rhd_U \Gamma \vdash^{(m-1,e,r)} p' : M$.*
2. Exponential*: if $p \to_{\mathsf{ue}} p'$, then $e \geq 1$ and there exists a type derivation $\Phi' \rhd_U \Gamma \vdash^{(m,e-1,r)} p' : M$.*

Proof. (Click here to go back to main chapter.)

1. We prove this by means of a weaker statement:

$$\textit{Let } p = P\langle(\lambda x.u)s\rangle \to_{\mathsf{um}} P\langle u, [x \leftarrow s]\rangle = p' \textit{ with } P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}\textit{, and let}$$

$$\Phi \rhd_U \Gamma \vdash^{(m,e,r)} P\langle(\lambda x.u)s\rangle : M$$

$$\textit{be a type derivation such that } \mathsf{tight}_\Gamma(\mathcal{A})\textit{. Then } m \geq 1 \textit{ and there exists}$$

$$\Phi' \rhd_U \Gamma \vdash^{(m-1,e,r)} P\langle u, [x \leftarrow s]\rangle : M$$

We proceed by induction on the derivation of $P \in \mathcal{E}_{\mathcal{V}}$.

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be of the form

$$\frac{}{(\mathcal{H}, \epsilon) \in \mathcal{E}_{\mathsf{u}(\mathcal{H}),\mathsf{a}(\mathcal{H})}}\ \mathsf{MAx}$$

where $P = (\mathcal{H}, \epsilon)$, $\mathcal{U} = \mathsf{u}(\mathcal{H})$ and $\mathcal{A} = \mathsf{a}(\mathcal{H})$. Then Φ is of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} \mathcal{H}\langle(\lambda x.u)s\rangle : M}{\Gamma \vdash^{(m,e,r)} (\mathcal{H}\langle(\lambda x.u)s\rangle, \epsilon) : M}\ \mathsf{Lift}$$

The statement holds by application of Lemma 9.1.9 (Quantitative Subject Reduction for $\to_{\mathsf{um}}$ in term contexts) on Ψ, giving us that $m \geq 1$ and the existence of $\Psi' \rhd_U \Gamma \vdash^{(m-1,e,r)} (\mathcal{H}\langle u\rangle, [x \leftarrow s]) : M$, and allowing us to the latter as premise for an application of the Lift rule.

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}z] \in \mathcal{E}_{\mathsf{upd}(\mathcal{V},y,z),\mathsf{upd}(\mathcal{B},y,z)}}\ \mathsf{M}_{\mathsf{VAR}}$$

where $P = Q@[y{\leftarrow}z]$, $\mathcal{U} = \mathsf{upd}(\mathcal{V}, y, z)$ and $\mathcal{A} = \mathsf{upd}(\mathcal{B}, y, z)$. We proceed by case analysis on the last typing rule in Φ.

  - Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m',e',r')} Q\langle(\lambda x.u)s\rangle : M \quad \Theta \rhd_U z : N \vdash^{(m'',e'',r'')} z : N \quad N \neq \mathbf{0}}{\Pi \biguplus \{z : N\} \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle(\lambda x.u)s\rangle@[y{\leftarrow}z] : M}\ \mathrm{ES}$$

    with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. Note that if $y \in \mathcal{B}$ then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\{z:N\}}(\{z\})$; *i.e.*, $N \in \mathsf{Tight}$. Hence, $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi;y:N}(\mathcal{B})$ and we can apply the *i.h.* on Ψ to get that $m' \geq 1$ (hence $m \geq 1$) to obtain $\Psi' \rhd_U \Pi; y : N \vdash^{(m'-1,e',r')} Q\langle u, [x{\leftarrow}s]\rangle : M$. Therefore, we can derive Φ' as

$$\frac{\Psi' \rhd_U \Pi; y : N \vdash^{(m'-1,e',r')} Q\langle u, [x{\leftarrow}s]\rangle : M \quad \Theta \rhd_U z : N \vdash^{(m'',e'',r'')} z : N \quad N \neq \mathbf{0}}{\Pi \biguplus \{z : N\} \vdash^{(m'-1+m'',e'+e'',r'+r'')} Q\langle u, [x{\leftarrow}s]\rangle@[y{\leftarrow}z] : M}\ \mathrm{ES}$$

  - Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle(\lambda x.u)s\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle(\lambda x.u)s\rangle@[y{\leftarrow}z] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

    Note that since $y \notin \mathsf{dom}(\Gamma)$, we can apply the *i.h.* on Ψ to get that $m \geq 1$ to obtain $\Psi \rhd_U \Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle : M$. Therefore, we can derive Φ' as

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle u, [x{\leftarrow}s]\rangle@[y{\leftarrow}z] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}t] \in \mathcal{E}_{(\mathcal{V}\setminus\{y\})\cup\mathsf{u}(t),(\mathcal{B}\setminus\{y\})\cup\mathsf{a}(t)}}\ \mathsf{M}_{\mathsf{I}}$$

where $P = Q@[y{\leftarrow}i^+]$, $\mathcal{U} = (\mathcal{V} \setminus \{y\}) \cup \mathsf{u}(i^+)$ and $\mathcal{A} = (\mathcal{B} \setminus \{y\}) \cup \mathsf{a}(i^+)$. We proceed by case analysis on the last typing rule in Φ.

  - Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m',e',r')} Q\langle(\lambda x.u)s\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} i^+ : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle(\lambda x.u)s\rangle@[y{\leftarrow}i^+] : M}\ \mathrm{ES}$$

    where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$.
    Note that $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Delta(\mathsf{a}(i^+))$. Consequently, we have that $N \in \mathsf{Inert}$—by Lemma 9.1.3 (Typing properties of useful inert terms) on Θ. Thus, we also have that $\mathsf{tight}_{\Pi;y:N}(\mathcal{B})$, and so we can apply the *i.h.* on Ψ to get that $m' \geq 1$ (hence, $m \geq 1$) and the existence of $\Psi' \rhd_U \Pi; y : N \vdash^{(m'-1,e',r')} Q\langle u, [x{\leftarrow}s]\rangle : M$. We can simply derive Φ' as follows

$$\frac{\Psi' \rhd_U \Pi; y : N \vdash^{(m'-1,e',r')} Q\langle u, [x{\leftarrow}s]\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} i^+ : N}{\Pi \biguplus \Delta \vdash^{(m'-1+m'',e'+e'',r'+r'')} Q\langle u, [x{\leftarrow}s]\rangle@[y{\leftarrow}i^+] : M}\ \mathrm{ES}$$

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle(\lambda x.u)s\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle(\lambda x.u)s\rangle@[y{\leftarrow}i^+] : M}\ \mathrm{ES_{gc}}$$

Since $y \notin \mathsf{dom}(\Gamma)$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathcal{B})$. We can apply the *i.h.* on Ψ to get that $m \geq 1$ to obtain $\Psi' \rhd_U \Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle : M$. We can simply derive Φ' as follows

$$\frac{\Psi' \rhd_U \Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle@[y{\leftarrow}i^+] : M}\ \mathrm{ES_{gc}}$$

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{Q@[y{\leftarrow}t] \in \mathcal{E}_{\mathcal{U},\mathcal{A}}}\ \mathsf{M_{GC}}$$

where $P = Q@[y{\leftarrow}t]$. We proceed by case analysis on the last typing rule in Φ.

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m',e',r')} Q\langle(\lambda x.u)s\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle(\lambda x.u)s\rangle@[y{\leftarrow}t] : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. Note that since $y \notin \mathcal{B}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies $\mathsf{tight}_{\Gamma;y:N}(\mathcal{B})$. Hence, by *i.h.* on Ψ, $m' \geq 1$ (hence, $m \geq 1$) and there exists $\Psi' \rhd_U \Pi; y : N \vdash^{(m'-1,e',r')} Q\langle u, [x{\leftarrow}s]\rangle : M$. We can simply derive Φ' as follows

$$\frac{\Psi' \rhd_U \Pi; y : N \vdash^{(m'-1,e',r')} Q\langle u, [x{\leftarrow}s]\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N}{\Pi \biguplus \Delta \vdash^{(m'-1+m'',e'+e'',r'+r'')} Q\langle u, [x{\leftarrow}s]\rangle@[y{\leftarrow}t] : M}\ \mathrm{ES}$$

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle(\lambda x.u)s\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle(\lambda x.u)s\rangle@[y{\leftarrow}t] : M}\ \mathrm{ES_{gc}}$$

By *i.h.* on Ψ, $m \geq 1$ and there exists $\Psi' \rhd_U \Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle : M$. We can simply derive Φ' as follows

$$\frac{\Psi' \rhd_U \Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle@[y{\leftarrow}t] : M}\ \mathrm{ES_{gc}}$$

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \setminus \mathcal{B})}{Q@[y{\leftarrow}v] \in \mathcal{E}_{\mathcal{V}\setminus\{y\},\mathcal{B}}}\ \mathsf{M_U}$$

where $P = Q@[y{\leftarrow}v]$, $\mathcal{U} = \mathcal{V} \setminus \{y\}$ and $\mathcal{A} = \mathcal{B}$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m',e',r')} Q\langle(\lambda x.u)s\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} v : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle(\lambda x.u)s\rangle@[y{\leftarrow}v] : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$.
Since $y \notin \mathcal{B}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi;y:N}(\mathcal{B})$. We can then apply the *i.h.* on Θ to get that $m' \geq 1$ (hence $m \geq 1$) to obtain $\Psi' \rhd_U \Pi; y : N \vdash^{(m'-1,e',r')} Q\langle u, [x{\leftarrow}s]\rangle : M$. Therefore, we can derive Θ as

$$\frac{\Psi' \rhd_U \Pi; y : N \vdash^{(m'-1,e',r')} Q\langle u, [x{\leftarrow}s]\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} v : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'-1+m'',e'+e'',r'+r'')} Q\langle u, [x{\leftarrow}s]\rangle@[y{\leftarrow}v] : M}\ \mathrm{ES}$$

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle(\lambda x.u)s\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle(\lambda x.u)s\rangle@[y{\leftarrow}v] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

By *i.h.* on Ψ, $m \geq 1$ and there exists $\Psi' \rhd_U \Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle : M$. We can simply derive Φ' as follows

$$\frac{\Psi' \rhd_U \Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m-1,e,r)} Q\langle u, [x{\leftarrow}s]\rangle@[y{\leftarrow}v] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

- Let $P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle y\rangle@[y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{V}\cup\mathsf{u}(\mathcal{H}),\mathcal{B}\cup\mathsf{a}(\mathcal{H})}}\ \mathsf{M_{HER}}$$

where $P = Q\langle y\rangle@[y{\leftarrow}\mathcal{H}]$, $\mathcal{U} = \mathcal{V} \cup \mathsf{u}(\mathcal{H})$ and $\mathcal{A} = \mathcal{B} \cup \mathsf{a}(\mathcal{H})$. We proceed by case analysis on the last typing rule in Φ.
– Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m',e',r')} Q\langle y\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} \mathcal{H}\langle(\lambda x.u)s\rangle : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle(\lambda x.u)s\rangle] : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. Note that $x \notin \mathsf{dom}(\Pi; y : N)$, since $x \in \mathsf{dom}(\Pi; y : N)$ would imply that $x \in \mathsf{fv}(Q\langle y\rangle)$—by Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system)—and so this case may be discarded by α-conversion.
Moreover, note that $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}))$. We can then apply Lemma 9.1.9 (Quantitative Subject Reduction for $\to_{\mathsf{um}}$ in term contexts) on Θ to get that $m'' \geq 1$ (hence, $m \geq 1$) to obtain $\Theta' \rhd_U \Delta \vdash^{(m''-1,e'',r'')} \mathcal{H}\langle u\rangle[x{\leftarrow}s] : N$. We now proceed by case analysis on the last typing rule in Θ':
∗ Let Θ' be of the form

$$\frac{\dfrac{\Xi \rhd_U \Delta_1; x : O \vdash^{(m_1'',e_1'',r_1'')} \mathcal{H}\langle u\rangle : N}{\Delta_1; x : O \vdash^{(m_1'',e_1'',r_1'')} (\mathcal{H}\langle u\rangle, \epsilon) : N}\ \mathsf{Lift} \quad \Omega \rhd_U \Delta_2 \vdash^{(m_2'',e_2'',r_2'')} s : O}{\Delta_1 \biguplus \Delta_2 \vdash^{(m_1''+m_2'',e_1''+e_2'',r_1''+r_2'')} (\mathcal{H}\langle u\rangle, [x{\leftarrow}s]) : N}\ \mathrm{ES}$$

with $\Delta_1 \biguplus \Delta_2 = \Delta$ and $(m_1'' + m_2'', e_1'' + e_2'', r_1'' + r_2'') = (m'' - 1, e'', r'')$. We can then derive Φ' as follows

$$\dfrac{\dfrac{\Psi \qquad \Xi}{(\Pi \biguplus \Delta_1) ; x : O \vdash^{(m'+m_1'', e'+e_1'', r'+r_1'')} Q\langle y\rangle @[y \leftarrow \mathcal{H}\langle u\rangle] : M}\ \text{ES} \qquad \Omega}{\Pi \biguplus \Delta_1 \biguplus \Delta_2 \vdash^{(m'+m_1''+m_2'', e'+e_1''+e_2'', r'+r_1''+r_2'')} (Q\langle y\rangle @[y \leftarrow \mathcal{H}\langle u\rangle]) @[x \leftarrow s] : M}\ \text{ES}$$

noting that $\Pi \biguplus \Delta_1 \biguplus \Delta_2 = \Pi \biguplus \Delta = \Gamma$, that

$$\begin{array}{rcl} (Q\langle y\rangle @[y \leftarrow \mathcal{H}\langle u\rangle]) @[x \leftarrow s] & = & (Q\langle y\rangle @[y \leftarrow \mathcal{H}])\, \langle u, [x \leftarrow s]\rangle \\ & = & P\langle u, [x \leftarrow s]\rangle \end{array}$$

and that

$$\begin{array}{rcl} (m' + m_1'' + m_2'', e' + e_1'' + e_2'', r' + r_1'' + r_2'') & = & (m' + m'' - 1, e' + e'', r' + r'') \\ & = & (m - 1, e, r) \end{array}$$

∗ Let Θ' be of the form

$$\dfrac{\dfrac{\Xi \rhd_U \Delta \vdash^{(m''-1,e'',r'')} \mathcal{H}\langle u\rangle : N}{\Delta \vdash^{(m''-1,e'',r'')} (\mathcal{H}\langle u\rangle, \epsilon) : N}\ \mathsf{Lift} \qquad \Delta(x) = \mathbf{0}}{\Delta \vdash^{(m''-1,e'',r'')} (\mathcal{H}\langle u\rangle, [x \leftarrow s]) : N}\ \text{ES}_{\text{gc}}$$

Since $y \notin \mathsf{dom}(\Pi)$, we can then derive Φ' as follows

$$\dfrac{\dfrac{\Psi \qquad \Xi}{\Pi \biguplus \Delta \vdash^{(m'+m''-1,e'+e'',r'+r'')} Q\langle y\rangle @[y \leftarrow \mathcal{H}\langle u\rangle] : M}\ \text{ES} \qquad (\Pi \biguplus \Delta)(x) = \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m''-1,e'+e'',r'+r'')} (Q\langle y\rangle @[y \leftarrow \mathcal{H}\langle u\rangle]) @[x \leftarrow s] : M}\ \text{ES}_{\text{gc}}$$

noting that

$$\begin{array}{rcl} (Q\langle y\rangle @[y \leftarrow \mathcal{H}\langle u\rangle]) @[x \leftarrow s] & = & (Q\langle y\rangle @[y \leftarrow \mathcal{H}])\, \langle u, [x \leftarrow s]\rangle \\ & = & P\langle u, [x \leftarrow s]\rangle \end{array}$$

and that $(m' + m'' - 1, e' + e'', r' + r'') = (m - 1, e, r)$.

– Suppose Φ is of the form

$$\dfrac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle y\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle y\rangle @[y \leftarrow \mathcal{H}\langle (\lambda x.u)s\rangle] : M}\ \text{ES}_{\text{gc}}$$

But since $y \notin \mathsf{dom}(\Pi)$ and $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathcal{B})$, then we can apply Lemma 13.6.8 (Plugged variables and domain of type contexts) on Ψ to obtain that $y \in \mathsf{dom}(\Gamma)$, which makes this case absurd.

2. We prove this by means of a weaker statement:

Let $p = P\langle x\rangle \to_{\mathsf{ue}} P\langle v^\alpha\rangle = q$, with $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$, $x \in \mathsf{dom}(P)$ and $P(x) = v$, and let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} P\langle x\rangle : M$ such that $\mathsf{tight}_\Gamma(\mathcal{A})$. Then $e \geq 1$ and there exists $\Phi' \rhd_U \Gamma \vdash^{(m,e-1,r)} P\langle v\rangle : M$.

We proceed by induction on the derivation of $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$.

- Note that the case where $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ is derived as

$$\frac{}{(\mathcal{H}^{@}, \epsilon) \in \mathcal{E}^{@}_{\mathsf{u}(\mathcal{H}^{@}),\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_1}$$

  where $P = (\mathcal{H}^{@}, \epsilon)$ is impossible, because it contradicts the hypothesis that $x \in \mathsf{dom}(P)$.
- Note that $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ cannot be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}] \in \mathcal{E}^{@}_{\mathcal{V}\cup\mathsf{u}(\mathcal{H}^{@}),\mathcal{B}\cup\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_2}$$

  because then $x \notin \mathsf{dom}(\Gamma)$, contradicting the hypothesis.
- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}z] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{V},y,z),\mathsf{upd}(\mathcal{B},y,z)}}\ \mathsf{E}_{\mathsf{VAR}}$$

  where $P = Q@[y{\leftarrow}z]$, $\mathcal{U} = \mathsf{upd}(\mathcal{V}, y, z)$ and $\mathcal{A} = \mathsf{upd}(\mathcal{B}, y, z)$. Note that since z is not a value, then $x \neq y$ and $x \in \mathsf{dom}(Q)$. We proceed by case analysis on the last typing rule in Φ:
  - Let Φ be of the form

$$\frac{\Psi \triangleright_U \Pi; y : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \triangleright_U z : N \vdash^{(m'',e'',r'')} z : N \quad N \neq \mathbf{0}}{\Pi \uplus \{z : N\} \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle@[y{\leftarrow}z] : M}\ \mathrm{ES}$$

    where $\Gamma = \Pi \uplus \{z : N\}$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. Note that if $y \in \mathcal{B}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $N \in \mathsf{Tight}$. Thus, we can apply the *i.h.* on Ψ and get that $m' \geq 1$ (hence $m \geq 1$) to obtain $\Psi' \triangleright_U \Pi; y : N \vdash^{(m',e'-1,r')} Q\langle v\rangle : M$. We can then derive Φ' as

$$\frac{\Psi' \triangleright_U \Pi; y : N \vdash^{(m',e'-1,r')} Q\langle v\rangle : M \quad \Theta \triangleright_U z : N \vdash^{(m'',e'',r'')} z : N \quad N \neq \mathbf{0}}{\Pi \uplus \{z : N\} \vdash^{(m'-1+m'',e'+e'',r'+r'')} Q\langle v\rangle@[y{\leftarrow}z] : M}\ \mathrm{ES}$$

  - Let Φ be of the form

$$\frac{\Psi \triangleright_U \Gamma \vdash^{(m,e,r)} Q\langle x\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle x\rangle@[y{\leftarrow}z] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

    Since $y \notin \mathsf{dom}(\Gamma)$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathcal{B})$ and so we can apply the *i.h.* on Ψ to get that $m \geq 1$ to obtain $\Psi \triangleright_U \Gamma \vdash^{(m,e-1,r)} Q\langle v, \epsilon\rangle : M$. We can then derive Φ' as

$$\frac{\Psi \triangleright_U \Gamma \vdash^{(m,e-1,r)} Q\langle v\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle v\rangle@[y{\leftarrow}z] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}t] \in \mathcal{E}^{@}_{(\mathcal{V}\setminus\{y\})\cup\mathsf{u}(t),(\mathcal{B}\setminus\{y\})\cup\mathsf{a}(t)}}\ \mathsf{E}_{\mathsf{I}}$$

  where $P = Q@[y{\leftarrow}i^+]$, $\mathcal{U} = (\mathcal{V} \setminus \{y\}) \cup \mathsf{u}(i^+)$ and $\mathcal{A} = (\mathcal{B} \setminus \{y\}) \cup \mathsf{a}(i^+)$. Note that $x \neq y$, because values are not inert terms, and so $x \in \mathsf{dom}(Q)$. We proceed by case analysis on the last typing rule in Φ.

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} i^+ : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle@[y{\leftarrow}i^+] : M}\ \text{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$.
Note that $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Delta(\mathsf{a}(i^+))$, and so we can apply Lemma 9.1.3 (Typing properties of useful inert terms) on Θ to infer that $N \in \mathsf{Inert}$. Hence, $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi;y:N}(\mathcal{B})$ and we can apply the *i.h.* on Ψ to get that $e' \geq 1$ (hence $e \geq 1$) to obtain $\Psi' \rhd_U \Pi; y : N \vdash^{(m',e'-1,r')} Q\langle v\rangle : M$. We can derive Φ' as

$$\frac{\Psi' \rhd_U \Pi; y : N \vdash^{(m',e'-1,r')} Q\langle v\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'-1+e'',r'+r'')} Q\langle v\rangle@[y{\leftarrow}t] : M}\ \text{ES}$$

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle x\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle x\rangle@[y{\leftarrow}t] : M}\ \text{ES}_{\text{gc}}$$

Since $y \notin \mathsf{dom}(\Gamma)$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathcal{B})$. Thus, we can apply the *i.h.* on Ψ, getting that $e \geq 1$ to obtaining a $\Psi' \rhd_U \Gamma \vdash^{(m,e-1,r)} Q\langle v\rangle : M$. We can then derive Φ' as follows

$$\frac{\Psi' \rhd_U \Gamma \vdash^{(m,e-1,r)} Q\langle v\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e-1,r)} Q\langle v\rangle@[y{\leftarrow}t] : M}\ \text{ES}_{\text{gc}}$$

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{Q@[y{\leftarrow}t] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E_{GC}}$$

We proceed by case analysis on the last typing rule in Φ and on whether $x = y$ or $x \neq y$.
– Let $x \neq y$ and Φ be of the form

$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle@[y{\leftarrow}t] : M}\ \text{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$.
Note that $\mathsf{tight}_\Gamma(\mathcal{A})$ together with the fact that $y \notin \mathcal{W}$ imply $\mathsf{tight}_{\Pi;y:N}(\mathcal{A})$. Hence, we can apply the *i.h.* on Ψ to get that $e' \geq 1$ (hence $e \geq 1$) to obtain $\Psi' \rhd_U \Pi; y : N \vdash^{(m',e'-1,r')} Q\langle v^\alpha\rangle : M$. We can then derive Φ' as follows

$$\frac{\Psi' \rhd_U \Pi; y : N \vdash^{(m',e'-1,r')} Q\langle v\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e''-1,r'+r'')} Q\langle v\rangle@[y{\leftarrow}t] : M}\ \text{ES}$$

– Let $x \neq y$ and Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle x\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle x\rangle@[y{\leftarrow}t] : M}\ \text{ES}_{\text{gc}}$$

We can apply the *i.h.* on Ψ to get that $e \geq 1$ to obtain $\Psi \rhd_U \Gamma \vdash^{(m,e-1,r)} Q\langle v\rangle : M$. We can then derive Φ' as

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e-1,r)} Q\langle v\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e-1,r)} Q\langle v\rangle @ [y \leftarrow t] : M}\ \mathrm{ES_{gc}}$$

– Let $x = y$, $t = v$ (*i.e.*, t is a value) and let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} v : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle @ [x \leftarrow v] : M}\ \mathrm{ES}$$

with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. Note that $x = y \notin \mathcal{A}$, that we may assume that $x \notin \mathsf{dom}(Q)$ by α-conversion, and that $M, N \neq \mathbf{0}$. Moreover, note that $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Gamma;x:N}(\mathcal{A})$. Hence, we can apply Lemma 9.1.8.2 (Linear Substitution for Useful Open CbNeed) on Ψ to obtain a splitting of the form $N = N_1 \uplus N_2$, with $N_1 \neq \mathbf{0}$ such that for every $\Xi \rhd_U \Sigma \vdash^{(m''',e''',r''')} v : N_1$ there exists

$$\Omega \rhd_U (\Pi \biguplus \Delta_1); x : N_2 \vdash^{(m'+m''',e'+e'''-1,r'+r''')} Q\langle v\rangle : M$$

We now apply Lemma 13.6.9 (Splitting multi types of Useful Open CbNeed type derivations) on Θ with said splitting, obtaining type derivations $\Theta_1 \rhd_U \Delta_1 \vdash^{(m_1'',e_1'',r_1'')} v : N_1$ and $\Theta_2 \rhd_U \Delta_2 \vdash^{(m_2'',e_2'',r_2'')} v : N_2$ such that $\Delta = \Delta_1 \biguplus \Delta_2$ and $(m'', e'', r'') = (m_1'' + m_2'', e_1'' + e_2'', r_1'' + r_2'')$.
Therefore, by what is given by Lemma 9.1.8.2 (Linear Substitution for Useful Open CbNeed), there exists

$$\Omega \rhd_U (\Pi \biguplus \Delta_1); x : N_2 \vdash^{(m'+m_1'',e'+e_1''-1,r'+r_1'')} Q\langle v\rangle : M$$

We finally do case analysis on whether $N_2 = \mathbf{0}$:

* Let $N_2 = \mathbf{0}$. By Lemma 13.6.9 (Splitting multi types of Useful Open CbNeed type derivations), this means that $\Delta_1 = \Delta$, $\Delta_2 = \mathbf{0}$, $(m_1'', e_1'', r_1'') = (m'', e'', r'')$ and $(m_2'', e_2'', r_2'') = (0, 0, 0)$. We can then derive Φ' as follows

$$\frac{\Omega \rhd_U \Pi \biguplus \Delta \vdash^{(m'+m'',e'+e''-1,r'+r'')} Q\langle v\rangle : M \quad (\Pi \biguplus \Delta)(x) = \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e''-1,r'+r'')} Q\langle v\rangle @ [x \leftarrow v] : M}\ \mathrm{ES_{gc}}$$

* Let $N_2 \neq \mathbf{0}$. We can then derive Φ' as follows

$$\frac{\Omega \rhd_U (\Pi \biguplus \Delta_1); x : N_2 \vdash^{(m'+m_1'',e'+e_1''-1,r'+r_1'')} Q\langle v\rangle : M \quad \Theta_2 \rhd_U \Delta_2 \vdash^{(m_2'',e_2'',r_2'')} v : N_2}{\Pi \biguplus \Delta_1 \biguplus \Delta_2 \vdash^{(m'+m_1''+m_2'',e'+e_1''-1+e_2'',r'+r_1''+r_2'')} Q\langle v\rangle @ [x \leftarrow v] : M}\ \mathrm{ES}$$

– Suppose $x = y$ and Φ is of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle y\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle y\rangle @ [y \leftarrow t] : M}\ \mathrm{ES_{gc}}$$

But then Lemma 13.6.8 (Plugged variables and domain of type contexts) on Ψ gives that $y \in \mathsf{dom}(\Gamma)$—absurd.

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{A}} \quad y \in (\mathcal{V} \setminus \mathcal{A})}{Q@[y{\leftarrow}v] \in \mathcal{E}^{@}_{\mathcal{V}\setminus\{y\},\mathcal{A}}}\ \mathsf{E_U}$$

where $P = Q@[y{\leftarrow}v]$ and $\mathcal{U} = \mathcal{V} \setminus \{y\}$. We proceed by case analysis on the last typing rule in Φ and on whether $x = y$ or $x \neq y$:

  - Let $x \neq y$ and Φ be of the form

$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} w : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle@[y{\leftarrow}w] : M}\ \mathrm{ES}$$

    where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$.
    Since $y \notin \mathcal{A}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi;y:N}(\mathcal{A})$. Hence, we can apply the *i.h.* on Ψ to get that $e' \geq 1$ (hence $e \geq 1$) to obtain $\Psi' \rhd_U \Pi; y : N \vdash^{(m',e'-1,r')} Q\langle v\rangle : M$. We can derive Φ' as

$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m',e'-1,r')} Q\langle v\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} t : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'-1+e'',r'+r'')} Q\langle v\rangle@[y{\leftarrow}t] : M}\ \mathrm{ES}$$

  - Let $x \neq y$ and Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle x\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle x\rangle@[y{\leftarrow}w] : M}\ \mathrm{ES_{gc}}$$

    We can apply the *i.h.* on Ψ to get that $e \geq 1$ to obtain $\Psi' \rhd_U \Gamma \vdash^{(m,e-1,r)} Q\langle v\rangle : M$. We can derive Φ' as

$$\frac{\Psi' \rhd_U \Gamma \vdash^{(m,e-1,r)} Q\langle v\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e-1,r)} Q\langle v\rangle@[y{\leftarrow}t] : M}\ \mathrm{ES_{gc}}$$

  - Let $x = y$, $t = v$ and Φ be of the form

$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m'',e'',r'')} v : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle@[x{\leftarrow}v] : M}\ \mathrm{ES}$$

    with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m' + m'', e' + e'', r' + r'')$. Note that $x = y \notin \mathcal{A}$, that we may assume that $x \notin \mathsf{dom}(Q)$ by α-conversion, and that $M, N \neq \mathbf{0}$. Moreover, note that $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Gamma;x:N}(\mathcal{A})$. Hence, we can apply Lemma 9.1.8.2 (Linear Substitution for Useful Open CbNeed) on Ψ to obtain a splitting of the form $N = N_1 \uplus N_2$, with $N_1 \neq \mathbf{0}$ such that for every $\Xi \rhd_U \Sigma \vdash^{(m''',e''',r''')} v : N_1$ there exists

$$\Omega \rhd_U (\Pi \biguplus \Delta_1); x : N_2 \vdash^{(m'+m''',e'+e'''-1,r'+r''')} Q\langle v\rangle : M$$

    We now apply Lemma 13.6.9 (Splitting multi types of Useful Open CbNeed type derivations) on Θ with said splitting, obtaining type derivations $\Theta_1 \rhd_U \Delta_1 \vdash^{(m''_1,e''_1,r''_1)} v : N_1$ and $\Theta_2 \rhd_U \Delta_2 \vdash^{(m''_2,e''_2,r''_2)} v : N_2$ such that $\Delta = \Delta_1 \biguplus \Delta_2$ and $(m'', e'', r'') = (m''_1 + m''_2, e''_1 + e''_2, r''_1 + r''_2)$.

Therefore, by what is given by Lemma 9.1.8.2 (Linear Substitution for Useful Open CbNeed), there exists

$$\Omega \rhd_U (\Pi \biguplus \Delta_1); x : N_2 \vdash^{(m'+m_1'', e'+e_1''-1, r'+r_1'')} Q\langle v\rangle : M$$

We finally do case analysis on whether $N_2 = \mathbf{0}$:

* Let $N_2 = \mathbf{0}$. By Lemma 13.6.9 (Splitting multi types of Useful Open CbNeed type derivations), this means that $\Delta_1 = \Delta$, $\Delta_2 = \mathbf{0}$, $(m_1'', e_1'', r_1'') = (m'', e'', r'')$ and $(m_2'', e_2'', r_2'') = (0, 0, 0)$. We can then derive Φ' as follows

$$\frac{\Omega \rhd_U \Pi \biguplus \Delta \vdash^{(m'+m'', e'+e''-1, r'+r'')} Q\langle v\rangle : M \quad (\Pi \biguplus \Delta)(x) = \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'', e'+e''-1, r'+r'')} Q\langle v\rangle @ [x \leftarrow v] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

* Let $N_2 \neq \mathbf{0}$. We can then derive Φ' as follows

$$\frac{\Omega \rhd_U (\Pi \biguplus \Delta_1); x : N_2 \vdash^{(m'+m_1'', e'+e_1''-1, r'+r_1'')} Q\langle v\rangle : M \quad \Theta_2 \rhd_U \Delta_2 \vdash^{(m_2'', e_2'', r_2'')} v : N_2}{\Pi \biguplus \Delta_1 \biguplus \Delta_2 \vdash^{(m'+m_1''+m_2'', e'+e_1''-1+e_2'', r'+r_1''+r_2'')} Q\langle v\rangle @ [x \leftarrow v] : M}\ \mathrm{ES}$$

– Suppose $x = y$ and Φ is of the form

$$\frac{\Psi' \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle y\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle y\rangle @ [y \leftarrow w] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

But then Lemma 13.6.8 (Plugged variables and domain of type contexts) on Ψ gives that $y \in \mathsf{dom}(\Gamma)$—absurd.

- Finally, note that $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ cannot be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{W},} \quad y \notin \mathcal{W}}{Q\langle y\rangle @ [y \leftarrow \mathcal{H}] \in \mathcal{E}_{\mathcal{W} \cup \mathsf{nv}(\mathcal{H})}}\ \mathsf{O}_{\mathsf{HER}}$$

because then $x \notin \mathsf{dom}(P)$, contradicting the hypothesis.

□

(Click here to go back to main chapter.)

Theorem 13.6.12 (Tight Correctness for Useful Open CbNeed)**.**
Given $p \in \mathcal{PR}$ and a tight type derivation $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} p : M$, there exists $q \in \mathcal{PR}$ such that

1. *q is in $\rightarrow_{\mathsf{und}}$-normal form,*
2. *there exists a reduction sequence $d : p \rightarrow^*_{\mathsf{und}} q$, and*
3. *$(m, e, r) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})$.*
4. *$\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$.*

Proof. (Click here to go back to main chapter.)

By induction on $m + e$, and proceeding by case analysis on whether $p \rightarrow_{\mathsf{und}}$-reduces or not.

First, note that if p is in $\rightarrow_{\mathsf{und}}$-normal form, then we can apply Proposition 9.1.7 (Typing properties of Useful Open CbNeed-normal forms) to get that $(m, e, r) = (0, 0, |p|_{\mathsf{nd}})$ and $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.

Otherwise, if $p \rightarrow_{\mathsf{und}} q$ for some $q \in \mathcal{PR}$, then there are two sub-cases to consider:

1. *Multiplicative steps*: Let $p \to_{\mathsf{um}} r$. By Proposition 9.1.10.1 (Quantitative Subject Reduction for Useful Open CbNeed - multiplicative steps), there exists $\Psi \rhd_U \Gamma \vdash^{(m-1,e,r)} r : M$. By *i.h.* on Ψ, there exist $q \in \mathcal{PR}$ such that
   (a) q is in $\to_{\mathsf{und}}$-normal form,
   (b) there exists a reduction sequence $d' : r \to^*_{\mathsf{und}} q$,
   (c) $(m-1, e, r) = (|d'|_{\mathsf{m}}, |d'|_{\mathsf{e}}, |q|_{\mathsf{nd}})$, and
   (d) $\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$.

   Since $p \to_{\mathsf{um}} r$, we see that there exists $\to_{\mathsf{und}}$-normalizing reduction sequence d given as follows

$$d \,:\, p \;\to^*_{\mathsf{und}} q = p \to_{\mathsf{um}} \underbrace{r \;\to^*_{\mathsf{und}} q}_{d'}$$

   and so the statement is satisfied. In particular, note that

$$(m, e, r) = (|d'|_{\mathsf{m}} + 1, |d'|_{\mathsf{e}}, |q|_{\mathsf{nd}}) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})$$

2. *Exponential steps*: Let $p \to_{\mathsf{ue}} r$. By Proposition 9.1.10.2 (Quantitative Subject Reduction for Useful Open CbNeed - exponential steps), there exists $\Psi \rhd_U \Gamma \vdash^{(m,e-1,r)} r : M$. By *i.h.* on Ψ, there exist $q \in \mathcal{PR}$ such that
   (a) q is in $\to_{\mathsf{und}}$-normal form,
   (b) there exists a reduction sequence $d' : r \to^*_{\mathsf{und}} q$,
   (c) $(m, e-1, r) = (|d'|_{\mathsf{m}}, |d'|_{\mathsf{e}}, |q|_{\mathsf{nd}})$, and
   (d) $\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$.

   Since $p \to_{\mathsf{ue}} r$, we see that there exists $\to_{\mathsf{und}}$-normalizing reduction sequence d given as follows

$$d \,:\, p \;\to^*_{\mathsf{und}} q = p \to_{\mathsf{ue}} \underbrace{r \;\to^*_{\mathsf{und}} q}_{d'}$$

   and so the statement is satisfied. In particular, note that

$$(m, e, r) = (|d'|_{\mathsf{m}}, |d'|_{\mathsf{e}} + 1, |q|_{\mathsf{nd}}) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})$$

□

(Click here to go back to main chapter.)

13.6.2 Useful Open CbNeed completeness.

Lemma 13.6.13 (Tight typability of values)**.**
Let $v \in \mathsf{Val}$ and $M \in \mathsf{Abs}$. Then there exists tight type derivation $\Phi \rhd_U \Gamma \vdash^{(0,0,|v|_{\mathsf{nd}})} v : M$ such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(v)$.

Proof. *(Click here to go back to main chapter.)*
Let $n := |M|$, noting that $n \geq 1$. The statement holds by taking Φ to be

$$\dfrac{\left(\dfrac{}{\emptyset \vdash^{(0,0,0)} \lambda x.u : \mathsf{abs}}\ \mathsf{abs}\right)_{i=1}^{n}}{\emptyset \vdash^{(0,0,0)} \lambda x.u : [\mathsf{abs}]_{i=1}^{n}}\ \mathsf{many}$$

□

(Click here to go back to main chapter.)

Lemma 13.6.14 (Tight typability of useful inert terms)**.**
Let i^+ be a useful inert term, $M \in \mathsf{Inert}$, and $f : \mathsf{ul}(i^+) \to \{\mathsf{Inert}, \mathsf{Abs}\}$ be a choice function. Then there exists

- *type context Γ such that $\mathsf{dom}(\Gamma) = \mathsf{a}(i^+)$ and $\mathsf{inert}(\Gamma)$;*
- *type context Π such that $\mathsf{dom}(\Pi) = \mathsf{ul}(i^+)$ and $\Pi(z) \in f(z)$, for every $z \in \mathsf{dom}(\Pi)$;*
- *tight type derivation $\Phi \rhd_U \Gamma; \Pi \vdash^{(0,0,|i^+|_{\mathsf{nd}})} i^+ : M$.*

Proof. (Click here to go back to main chapter.)
Let $i^+ = us$, $M \in \mathsf{Inert}$ and $f : \mathsf{ul}(i^+) \to \{\mathsf{Inert}, \mathsf{Abs}\}$. We proceed by case analysis on the shape of u:

- Let $u = x \in \mathsf{Var}$. Then $\mathsf{a}(t) = \{x\} \cup \mathsf{a}(s)$, $\mathsf{u}(t) = \mathsf{u}(s)$, and $|t|_{\mathsf{nd}} = |s|_{\mathsf{nd}} + 1$. We proceed by case analysis on the shape of s:
  - Let $s = y \in \mathsf{Var}$. Then $\mathsf{a}(t) = \{x\}$, and $\mathsf{u}(t) = \{y\}$ and $|t|_{\mathsf{nd}} = 1$.
    On one hand, if $y = x$, then $\mathsf{ul}(i^+) = \emptyset$ and the statement holds by taking any $N \in \mathsf{Inert}$, and defining $\Gamma \coloneqq \{x : (M \uplus N)\}$, $\Pi \coloneqq \emptyset$, and Φ as follows

$$\dfrac{\dfrac{}{x : M \vdash^{(0,0,0)} x : M}\ \mathsf{ax}_T \quad \dfrac{}{x : N \vdash^{(0,0,0)} x : N}\ \mathsf{ax}_T}{x : (M \uplus N) \vdash^{(0,0,1)} xx : M}\ \mathsf{app}_i$$

    On the other hand, if $y \neq x$, then $\mathsf{u}(t) \setminus \mathsf{a}(t) = \{y\}$ and the statement holds by taking $N \in f(y)$ and defining $\Gamma \coloneqq \{x : M\}$, $\Pi = \{y : N\}$, and Φ as follows:

$$\dfrac{\dfrac{}{x : M \vdash^{(0,0,0)} x : M}\ \mathsf{ax}_T \quad \dfrac{N \in f(y)}{y : N \vdash^{(0,0,0)} y : N}\ \mathsf{ax}_T}{\{x : M\}; \{y : N\} \vdash^{(0,0,1)} xy : M}\ \mathsf{app}_i$$

  - Let $s = j^+$. Then $\mathsf{a}(t) = \{x\} \cup \mathsf{a}(j^+)$, and $\mathsf{u}(t) = \mathsf{u}(j^+)$ and $|t|_{\mathsf{nd}} = |j^+|_{\mathsf{nd}} + 1$. Let $N \coloneqq [\mathsf{inert}]$ and define g as

$$\begin{array}{rcccl} g & : & \mathsf{ul}(j^+) & \to & \{\mathsf{Inert}, \mathsf{Abs}\} \\ & & z & \to & \begin{cases} f(z) & z \in \mathsf{dom}(f) \\ \mathsf{Inert} & \text{otherwise} \end{cases} \end{array}$$

    By *i.h.* there exists a tight type derivation $\Theta \rhd_U T; \Upsilon \vdash^{(0,0,|j^+|_{\mathsf{nd}})} j^+ : N$ such that $\mathsf{inert}(T)$, $\mathsf{dom}(T) = \mathsf{a}(j^+)$, $\mathsf{dom}(\Upsilon) = \mathsf{ul}(j^+)$ and $\Upsilon(y) \in g(z)$ for every $z \in \mathsf{dom}(\Upsilon)$. Finally, if $x \in \mathsf{dom}(\Upsilon)$, then we can derive Φ as follows:

$$\dfrac{\dfrac{}{x : M \vdash^{(0,0,0)} x : M}\ \mathsf{ax}_T \quad \Theta \rhd_U T; \Upsilon \vdash^{(0,0,|j^+|_{\mathsf{nd}})} j^+ : N}{(T \biguplus \{x : (M \uplus \Upsilon(x))\}); (\Upsilon \setminus\setminus \{x\}) \vdash^{(0,0,|j^+|_{\mathsf{nd}}+1)} xj^+ : M}\ \mathsf{app}_i$$

    and if $x \notin \mathsf{dom}(\Upsilon)$, then we can derive Φ as follows:

$$\dfrac{\dfrac{}{x : M \vdash^{(0,0,0)} x : M}\ \mathsf{ax}_T \quad \Theta \rhd_U T; \Upsilon \vdash^{(0,0,|j^+|_{\mathsf{nd}})} j^+ : N}{(T \biguplus \{x : M\}); \Upsilon \vdash^{(0,0,|j^+|_{\mathsf{nd}}+1)} xj^+ : M}\ \mathsf{app}_i$$

– Let $s \in \mathsf{Val}$. Then $\mathsf{a}(t) = \{x\}$, $\mathsf{u}(t) = \emptyset$ and $|t|_{\mathsf{nd}} = 1$. Moreover, note that $\mathsf{dom}(f) = \emptyset$. Let $N \coloneqq [\mathsf{abs}]$. By Lemma 9.1.12 (Tight typability of values), there exists a tight type derivation $\Theta \triangleright_U T \vdash^{(0,0,|s|_{\mathsf{nd}})} s : N$ such that $\mathsf{dom}(T) = \mathsf{nv}(s)$. Since $\mathsf{nv}(s) = \emptyset$ then the statement holds by defining $\Gamma \coloneqq \{x : M\}$, $\Pi \coloneqq \emptyset$, and Φ as follows

$$\dfrac{\dfrac{}{x : M \vdash^{(0,0,0)} x : M}\ \mathsf{ax}_T \qquad \Theta \triangleright_U T \vdash^{(0,0,|s|_{\mathsf{nd}})} s : N}{x : M \vdash^{(0,0,1)} xs : M}\ \mathsf{app}_i$$

- Let $u = k^+$. Since $k^+ \notin \mathsf{Var}$ then $\mathsf{a}(t) = \mathsf{a}(k^+) \cup \mathsf{a}(s)$, $\mathsf{u}(t) = \mathsf{u}(k^+) \cup \mathsf{u}(s)$, and $|t|_{\mathsf{nd}} = |k^+|_{\mathsf{nd}} + |s|_{\mathsf{nd}} + 1$. Let g be defined as

$$\begin{array}{rcccl} g & : & \mathsf{ul}\,(k^+) & \to & \{\mathsf{Inert}, \mathsf{Abs}\} \\ & & z & \to & \begin{cases} f(z) & z \in \mathsf{dom}(f) \\ \mathsf{Inert} & \text{otherwise} \end{cases} \end{array}$$

We can apply the *i.h.* on k^+, M and g to obtain a tight type derivation $\Psi \triangleright_U \Delta; \Sigma \vdash^{(0,0,|k^+|_{\mathsf{nd}})} k^+ : M$ such that $\mathsf{dom}(\Delta) = \mathsf{a}(k^+)$, $\mathsf{inert}(\Delta)$, $\mathsf{dom}(\Sigma) = \mathsf{ul}\,(k^+)$, and $\Sigma(z) \in g(z)$ for every $z \in \mathsf{dom}(\Sigma)$.
We now proceed by case analysis on the shape of s:
– Let $s = y \in \mathsf{Var}$. Then $\mathsf{a}(k^+) = \mathsf{a}(u)$, $\mathsf{u}(t) = \mathsf{u}(k^+) \cup \{y\}$ and $|t|_{\mathsf{nd}} = |k^+|_{\mathsf{nd}} + 1$.
Let

$$N \coloneqq \begin{cases} [\mathsf{abs}] & y \in \mathsf{dom}(f) \text{ and } f(y) = \mathsf{Abs} \\ [\mathsf{inert}] & \text{otherwise} \end{cases}$$

The statement follows by defining

$$\Gamma \coloneqq \begin{cases} \Delta & y \notin \mathsf{a}(u) \\ \Delta \uplus \{y : N\} & y \in \mathsf{a}(y) \end{cases} \qquad \Pi \coloneqq \begin{cases} \Sigma \uplus \{y : N\} & y \notin \mathsf{a}(u) \\ \Sigma & y \in \mathsf{a}(y) \end{cases}$$

and Φ as follows

$$\dfrac{\Psi \triangleright_U \Delta; \Sigma \vdash^{(0,0,|k^+|_{\mathsf{nd}})} k^+ : M \qquad \dfrac{}{y : N \vdash^{(0,0,0)} y : N}\ \mathsf{ax}_T}{\Gamma; \Pi \vdash^{(0,0,|k^+|_{\mathsf{nd}}+1)} k^+ y : M}\ \mathsf{app}_i$$

– Let $s = j^+$. Let h be defined as

$$\begin{array}{rcccl} h & : & \mathsf{ul}\,(j^+) & \to & \{\mathsf{Inert}, \mathsf{Abs}\} \\ & & z & \to & \begin{cases} f(z) & z \in \mathsf{dom}(f) \\ \mathsf{Inert} & \text{otherwise} \end{cases} \end{array}$$

We can then apply the *i.h.* on j^+, $N \coloneqq [\mathsf{inert}]$ and h to obtain a tight type derivation $\Theta \triangleright_U T; \Upsilon \vdash^{(0,0,|j^+|_{\mathsf{nd}})} j^+ : N$ such that $\mathsf{dom}(T) = \mathsf{a}(j^+)$, $\mathsf{inert}(T)$, $\mathsf{dom}(\Upsilon) = \mathsf{ul}\,(j^+)$, and $\Upsilon(z) \in h(z)$ for every $z \in \mathsf{dom}(\Upsilon)$. Therefore, the statement holds by defining $\Pi \coloneqq (\Sigma \setminus\setminus T) \uplus (\Upsilon \setminus\setminus \Delta)$,

$$\begin{array}{rcl} \Gamma & \coloneqq & \Delta \uplus T \\ & & \uplus \{\Sigma(x) : x \in (\mathsf{dom}(\Sigma) \cap \mathsf{dom}(T))\} \\ & & \uplus \{\Upsilon(x) : x \in (\mathsf{dom}(\Upsilon) \cap \mathsf{dom}(\Delta))\}, \end{array}$$

and Φ as follows

$$\frac{\Psi \rhd_U \Delta; \Sigma \vdash^{(0,0,|k^+|_{\mathsf{nd}})} k^+ : M \quad \Theta \rhd_U T; \Upsilon \vdash^{(0,0,|j^+|_{\mathsf{nd}})} j^+ : N}{\Gamma; \Pi \vdash^{(0,0,|k^+|_{\mathsf{nd}}+|j^+|_{\mathsf{nd}}+1)} k^+ j^+ : M}\ \mathsf{app}_i$$

– Let $s \in \mathsf{Val}$. Then $\mathsf{a}(t) = \mathsf{a}(k^+)$, $\mathsf{u}(t) = \mathsf{u}(k^+)$ and $|t|_{\mathsf{nd}} = |k^+|_{\mathsf{nd}} + 1$. Moreover, we can apply Lemma 9.1.12 (Tight typability of values) to obtain a tight type derivation $\Theta \rhd_U \Sigma \vdash^{(0,0,|s|_{\mathsf{nd}})} s : [\mathsf{abs}]$ such that $\mathsf{dom}(\Sigma) = \mathsf{nv}(s) = \emptyset$. Therefore, the statement follows by defining $\Gamma \coloneqq \Delta$, $\Pi \coloneqq \Sigma$ and Φ as follows

$$\frac{\Psi \rhd_U \Delta; \Sigma \vdash^{(0,0,|k^+|_{\mathsf{nd}})} k^+ : M \quad \Theta \rhd_U \emptyset \vdash^{(0,0,|s|_{\mathsf{nd}})} s : [\mathsf{abs}]}{\Pi; \Sigma \vdash^{(0,0,|k^+|_{\mathsf{nd}}+|s|_{\mathsf{nd}}+1)} us : M}\ \mathsf{app}_i$$

□

(Click here to go back to main chapter.)

Lemma 13.6.15 (Tight typability of generalized variables)**.**
Let $p \in \mathcal{PR}$ be such that $\mathsf{genVar}_{\#}(p)$ and $M \in \mathsf{Tight}$. Then there exists tight type derivation $\Phi \rhd_U \Gamma \vdash^{(0,0,|p|_{\mathsf{nd}})} p : M$ such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.

Proof. *(Click here to go back to main chapter.)*
We proceed by induction on the derivation of $\mathsf{genVar}_x(p)$:

- Let $p = (x, \epsilon)$ and $M \in \mathsf{Tight}$. Then the statement holds by simply deriving Φ as follows:

$$\frac{\dfrac{}{x : M \vdash^{(0,0,0)} x : M}\ \mathsf{ax}_T}{x : M \vdash^{(0,0,0)} (x, \epsilon) : M}\ \mathsf{Lift}$$

- Let $\mathsf{genVar}_x(p)$ be derived as

$$\frac{\mathsf{genVar}_y(q)}{\mathsf{genVar}_x(q@[y{\leftarrow}x])}\ \mathsf{GV}_{\mathsf{HER}}$$

where $p = q@[y{\leftarrow}x]$. By Lemma 8.2.4 (Properties of generalized variables), $\mathsf{nv}(q) = \{y\}$, $\mathsf{nv}(p) = \{x\}$ and $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |x|_{\mathsf{nd}} = |q|_{\mathsf{nd}}$. In addtition, by *i.h.* on $\mathsf{genVar}_y(q)$, for every $M \in \mathsf{Tight}$ there exists a tight type derivation $\Psi \rhd_U \Pi \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M$ such that $\mathsf{dom}(\Pi) = \mathsf{nv}(q) = \{y\}$. Moreover, Lemma 9.1.4 (Typing properties of generalized variables) gives us that $\Pi(y) = M$. Therefore, we can derive Φ as follows

$$\frac{\Psi \rhd_U y : M \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M \quad \dfrac{}{x : M \vdash^{(0,0,0)} x : M}\ \mathsf{ax}_T}{x : M \vdash^{(0,0,|q|_{\mathsf{nd}})} q@[y{\leftarrow}x] : M}\ \mathrm{ES}$$

- Let $\mathsf{genVar}_x(p)$ be derived as

$$\frac{\mathsf{genVar}_x(p) \quad y \neq x}{\mathsf{genVar}_x(p@[y{\leftarrow}t])}\ \mathsf{GV}_{\mathsf{GC}}$$

where $p = p@[y{\leftarrow}t]$. By Lemma 8.2.4 (Properties of generalized variables), $\mathsf{nv}(q) = \{x\}$, $\mathsf{nv}(p) = \{x\}$ and $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}}$. In addtition, by *i.h.* on $\mathsf{genVar}_x(q)$, for every $M \in \mathsf{Tight}$ there exists a tight type derivation $\Psi \rhd_U \Pi \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M$ such that $\mathsf{dom}(\Pi) = \mathsf{nv}(q) = \{x\}$. Therefore, we can derive Φ as follows

$$\frac{\Psi \rhd_U \Pi \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M \quad \Pi(y) = \mathbf{0}}{\Pi \vdash^{(0,0,|q|_{\mathsf{nd}})} q@[y{\leftarrow}x] : M}\ \mathrm{ES}_{\mathrm{gc}}$$

□

(Click here to go back to main chapter.)

Lemma 13.6.16 (Tight typability of useful abstraction programs)**.**
Let $p \in \mathcal{PR}$ be such that $\mathsf{uabs}(p)$. Then there exists tight type derivation $\Phi \rhd_U \Gamma \vdash^{(0,0,|p|_{\mathsf{nd}})} p : [\mathsf{abs}]$ such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.

Proof. (Click here to go back to main chapter.)
We proceed by induction on the derivation of $\mathsf{uabs}(p)$:

- Let $\mathsf{uabs}(p)$ be derived as
$$\frac{}{\mathsf{uabs}(v, \epsilon)}\ \mathsf{A}_{\mathsf{Lift}}$$
where $p = (v, \epsilon)$. The statement follows from Lemma 9.1.2 (Typing properties of values).
- Let $\mathsf{uabs}(p)$ be derived as
$$\frac{\mathsf{genVar}_x(q)}{\mathsf{uabs}(q@[x{\leftarrow}v])}\ \mathsf{A}_{\mathsf{GV}}$$
where $p = q@[x{\leftarrow}v]$. Note that $\mathsf{nv}(q) = \{x\}$—by Lemma 8.2.4 (Properties of generalized variables)—that $\mathsf{nv}(p) = \mathsf{nv}(t)$, and that $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |t|_{\mathsf{nd}}$. Let $M \coloneqq [\mathsf{abs}]$. By Lemma 9.1.14 (Tight typability of generalized variables), there exists a tight type derivation $\Psi \rhd_U \Pi \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M$ such that $\mathsf{dom}(\Pi) = \mathsf{nv}(q)$; by Lemma 9.1.4 (Typing properties of generalized variables), $\Pi(x) = M$. Moreover, by Lemma 9.1.12 (Tight typability of values) there exists a tight type derivation $\Theta \rhd_U \Delta \vdash^{(0,0,|t|_{\mathsf{nd}})} t : M$ such that $\mathsf{dom}(\Delta) = \mathsf{nv}(t)$. Therefore, we can derive Φ as follows
$$\frac{\Psi \rhd_U \{x : M\} \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M \quad \Theta \rhd_U \Delta \vdash^{(0,0,|t|_{\mathsf{nd}})} t : M}{\Delta \vdash^{(0,0,|q|_{\mathsf{nd}}+|t|_{\mathsf{nd}})} q@[x{\leftarrow}t] : M}\ \mathrm{ES}$$
- Let $\mathsf{uabs}(p)$ be derived as
$$\frac{\mathsf{uabs}(q)}{\mathsf{uabs}(q@[x{\leftarrow}t])}\ \mathsf{A}_{\mathsf{GC}}$$
where $p = q@[x{\leftarrow}t]$. By Lemma 8.2.5, we have that $\mathsf{nv}(q) = \mathsf{nv}(p) = \emptyset$, hence $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}}$. Moreover, by *i.h.*, there exists a tight type derivation $\Psi \rhd_U \Pi \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{abs}]$ such that $\mathsf{dom}(\Pi) = \mathsf{nv}(q)$. Note that by Lemma 8.2.5 (Properties of useful abstraction programs), $\emptyset = \mathsf{nv}(q) = \mathsf{dom}(\Pi)$. Therefore, we can derive Φ as follows
$$\frac{\Psi \rhd_U \emptyset \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{abs}]}{\emptyset \vdash^{(0,0,|q|_{\mathsf{nd}})} q@[x{\leftarrow}t] : [\mathsf{abs}]}\ \mathrm{ES}_{\mathrm{gc}}$$

□

(Click here to go back to main chapter.)

Lemma 13.6.17 (Tight typability of useful inert programs)**.**
Let $p \in \mathcal{PR}$ be such that $\mathsf{uinert}(p)$, and let $f : \mathsf{ul}(p) \to \{\mathsf{Inert}, \mathsf{Abs}\}$ be a choice function. Then there exists

- *type context Γ such that $\mathsf{dom}(\Gamma) = \mathsf{a}(p)$ and $\mathsf{inert}(\Gamma)$;*
- *type context Π such that $\mathsf{dom}(\Pi) = \mathsf{ul}(p)$ and $\Pi(z) = f(z)$, for every $z \in \mathsf{dom}(\Pi)$;*
- *tight type derivation $\Phi \rhd_U \Gamma; \Pi \vdash^{(0,0,|p|_{\mathsf{nd}})} p : [\mathsf{inert}]$.*

Proof. (Click here to go back to main chapter.)

Let $f : \mathsf{ul}\,(p) \to \{\mathsf{Inert}, \mathsf{Abs}\}$. We proceed by induction on the derivation of $\mathsf{uinert}(p)$:

- Let $\mathsf{uinert}(p)$ be derived as

$$\frac{}{\mathsf{uinert}(i^+, \epsilon)}\ \mathsf{I}_{\mathsf{Lift}}$$

where $p = (i^+, \epsilon)$. Note that $\mathsf{u}(p) = \mathsf{u}(i^+)$ and $\mathsf{a}(p) = \mathsf{a}(i^+)$. The statement holds by taking the type derivation yielded by applying Lemma 9.1.13 (Tight typability of useful inert terms) on i^+, with respect to f and $M \coloneqq [\mathsf{inert}]$, and using it as a premise for typing rule Lift.

- Let $\mathsf{uinert}(p)$ be derived as

$$\frac{\mathsf{uinert}(q) \quad x \in \mathsf{nv}(q)}{\mathsf{uinert}(q@[x{\leftarrow}i])}\ \mathsf{I}_{\mathsf{I}}$$

where $p = q@[x{\leftarrow}i]$. Moreover, let us define

$$\begin{array}{rcccl} g & : & \mathsf{ul}\,(q) & \to & \{\mathsf{Inert}, \mathsf{Abs}\} \\ & & z & \to & \begin{cases} f(z) & z \in \mathsf{dom}(f) \\ \mathsf{Inert} & \text{otherwise} \end{cases} \end{array}$$

By *i.h.* on $\mathsf{uinert}(q)$, there exists a tight type derivation $\Psi \rhd_U \Delta; \Sigma \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{inert}]$ such that $\mathsf{dom}(\Delta) = \mathsf{a}(q)$, $\mathsf{inert}(\Delta)$, $\mathsf{dom}(\Sigma) = \mathsf{ul}\,(q)$ and $\Upsilon(z) \in g(z)$ for every $z \in \mathsf{dom}(\Upsilon)$. We proceed by case analysis on the shape of i:

- Let $i = y \in \mathsf{Var}$. If $x \in \mathsf{a}(q)$, then the statement follows by defining $\Gamma \coloneqq (\Delta \setminus\setminus \{x\}) \uplus \{y : \Delta(x)\}$, $\Pi \coloneqq \Sigma$ and Φ as follows

$$\frac{\Psi \rhd_U (\Delta \setminus\setminus \{x\}); x : \Delta(x); \Sigma \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{inert}] \quad \overline{y : \Delta(x) \vdash^{(0,0,0)} y : \Delta(x)}\ \mathsf{ax}_T}{((\Delta \setminus\setminus \{x\}) \uplus \{y : \Delta(x)\}); \Sigma \vdash^{(0,0,|q|_{\mathsf{nd}})} q@[x{\leftarrow}y] : [\mathsf{inert}]}\ \mathrm{ES}$$

On the other hand, if $x \notin \mathsf{a}(q)$, then the statement follows by defining $\Gamma \coloneqq \Delta$, $\Pi \coloneqq (\Sigma \setminus\setminus \{x\}) \uplus \{y : \Sigma(x)\}$ and Φ as follows:

$$\frac{\Psi \rhd_U \Delta; (\Sigma \setminus\setminus \{x\}); x : \Sigma(x) \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{inert}] \quad \overline{y : \Sigma(x) \vdash^{(0,0,0)} y : \Sigma(x)}\ \mathsf{ax}_T}{\Delta; ((\Sigma \setminus\setminus \{x\}) \uplus \{y : \Sigma(x)\}) \vdash^{(0,0,|q|_{\mathsf{nd}})} q@[x{\leftarrow}y] : [\mathsf{inert}]}\ \mathrm{ES}$$

- Let $i \notin \mathsf{Var}$, and let

$$\begin{array}{rcccl} h & : & \mathsf{ul}\,(i) & \to & \{\mathsf{Inert}, \mathsf{Abs}\} \\ & & z & \to & \begin{cases} f(z) & z \in \mathsf{dom}(f) \\ \mathsf{Inert} & \text{otherwise} \end{cases} \end{array}$$

We further split the analysis on whether $x \in \mathsf{a}(q)$:

  - Let $x \in \mathsf{a}(q)$. We can apply Lemma 9.1.13 (Tight typability of useful inert terms) on i, $M \coloneqq \Delta(x)$ and h, to obtain a tight type derivation $\Theta \rhd_U T; \Upsilon \vdash^{(0,0,|i|_{\mathsf{nd}})} i : M$ such that $\mathsf{dom}(T) = \mathsf{a}(i)$, $\mathsf{inert}(T)$, $\mathsf{dom}(\Upsilon) = \mathsf{ul}\,(i)$, and $\Upsilon(z) \in h(z)$ for every $z \in \mathsf{dom}(\Upsilon)$. The statement then follows by defining $\Gamma \coloneqq (\Delta \setminus\setminus \{x\}) \uplus T$, $\Pi \coloneqq \Sigma \uplus \Upsilon$ and Φ as follows:

$$\frac{\Psi \rhd_U (\Delta \setminus\setminus \{x\}); x : \Delta(x); \Sigma \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{inert}] \quad \Theta \rhd_U T; \Upsilon \vdash^{(0,0,|i|_{\mathsf{nd}})} i : M}{(\Delta \setminus\setminus \{x\}) \uplus T; \Sigma \uplus \Upsilon \vdash^{(0,0,|q|_{\mathsf{nd}}+|i|_{\mathsf{nd}})} q@[x{\leftarrow}i] : [\mathsf{inert}]}\ \mathrm{ES}$$

∗ Let $x \notin \mathsf{a}(q)$. Then it must be that $x \in \mathsf{ul}\,(q)$. We can apply Lemma 9.1.13 (Tight typability of useful inert terms) on i, $M \coloneqq \Sigma(x)$ and h, to obtain a tight type derivation $\Theta \rhd_U T; \Upsilon \vdash^{(0,0,|i|_{\mathsf{nd}})} i : M$ such that $\mathsf{dom}(T) = \mathsf{a}(i)$, $\mathsf{inert}(T)$, $\mathsf{dom}(\Upsilon) = \mathsf{ul}\,(i)$ and $\Upsilon(z) \in h(z)$ for every $z \in \mathsf{dom}(\Upsilon)$. The statement then follows by defining $\Gamma \coloneqq \Delta \uplus T$, $\Pi \coloneqq (\Sigma \setminus\setminus \{x\}) \uplus \Upsilon$ and Φ as follows:

$$\frac{\Psi \rhd_U \Delta; (\Sigma \setminus\setminus \{x\}); x : \Sigma(x) \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{inert}] \quad \Theta \rhd_U T; \Upsilon \vdash^{(0,0,|i|_{\mathsf{nd}})} i : M}{\Delta \uplus T; (\Sigma \setminus\setminus \{x\}) \uplus \Upsilon \vdash^{(0,0,|q|_{\mathsf{nd}}+|i|_{\mathsf{nd}})} q@[x{\leftarrow}i] : [\mathsf{inert}]}\ \mathrm{ES}$$

- Let $\mathsf{uinert}(p)$ be derived as

$$\frac{\mathsf{genVar}_x(q)}{\mathsf{uinert}(q@[x{\leftarrow}i^+])}\ \mathsf{I}_{\mathsf{GV}}$$

where $p = q@[x{\leftarrow}i^+]$. By Lemma 8.2.4 (Properties of generalized variables), we have that $\{x\} = \mathsf{nv}(q)$, and so $\mathsf{nv}(p) = \mathsf{nv}(i^+)$ and $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |i^+|_{\mathsf{nd}}$.
Let $M \coloneqq [\mathsf{inert}]$. By Lemma 9.1.14 (Tight typability of generalized variables) on $\mathsf{genVar}_x(q)$ and M, there exists a type derivation $\Psi \rhd_U x : M \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M$.
Moreover, it is easy to verify that $\mathsf{ul}\,(p) = \mathsf{ul}\,(i^+)$. Hence, we can apply Lemma 9.1.13 (Tight typability of useful inert terms) on i^+, M and f, to obtain a tight type derivation $\Theta \rhd_U T; \Upsilon \vdash^{(0,0,|i^+|_{\mathsf{nd}})} i^+ : M$ such that $\mathsf{dom}(T) = \mathsf{a}(i^+)$, $\mathsf{inert}(T)$, $\mathsf{dom}(\Upsilon) = \mathsf{ul}\,(i^+)$ and $\Upsilon(z) \in f(z)$ for every $z \in \mathsf{dom}(\Upsilon)$. Therefore, the statement follows by defining $\Gamma \coloneqq T$, $\Pi \coloneqq \Upsilon$ and Φ as follows

$$\frac{\Psi \rhd_U x : M \vdash^{(0,0,|q|_{\mathsf{nd}})} q : M \quad \Theta \rhd_U T; \Upsilon \vdash^{(0,0,|i^+|_{\mathsf{nd}})} i^+ : M}{T; \Upsilon \vdash^{(0,0,|q|_{\mathsf{nd}}+|i^+|_{\mathsf{nd}})} q@[x{\leftarrow}i^+] : M}\ \mathrm{ES}$$

- Let $\mathsf{uinert}(p)$ be derived as

$$\frac{\mathsf{uinert}(q) \quad x \in \mathsf{ul}\,(q)}{\mathsf{uinert}(q@[x{\leftarrow}v])}\ \mathsf{I}_{\mathsf{U}}$$

where $p = q@[x{\leftarrow}v]$. Note that $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}} + |v|_{\mathsf{nd}}$. In addition, since $\mathsf{u}(v) = \mathsf{a}(v) = \emptyset$, then it is easy to show that $\mathsf{ul}\,(p) = \mathsf{ul}\,(q) \setminus \{x\}$. Let

$$g \ : \ \mathsf{ul}\,(q) \ \to \ \{\mathsf{Inert}, \mathsf{Abs}\}$$
$$z \ \to \ \begin{cases} f(z) & z \neq x \text{ and } z \in \mathsf{dom}(f) \\ \mathsf{Abs} & z = x \end{cases}$$

Then we can apply the *i.h.* on $\mathsf{uinert}(q)$ and g to get a tight type derivation $\Psi \rhd_U \Delta; \Sigma \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{inert}]$ such that $\mathsf{dom}(\Delta) = \mathsf{a}(q)$, $\mathsf{inert}(\Delta)$, $\mathsf{dom}(\Sigma) = \mathsf{ul}\,(q)$ and $\Sigma(z) \in g(z)$ for every $z \in \mathsf{dom}(\Sigma)$. Note that we can rewrite $\Sigma = (\Sigma \setminus\setminus \{x\}); (x : \Sigma(x))$.
Moreover, by Lemma 9.1.12 (Tight typability of values), there exists a tight type derivation $\Theta \rhd_U T \vdash^{(0,0,|v|_{\mathsf{nd}})} v : \Sigma(x)$ such that $\mathsf{dom}(T) = \mathsf{nv}(v) = \emptyset$.
The statement then follows by defining $\Gamma \coloneqq \Delta$, $\Pi \coloneqq \Sigma$ and Φ as follows:

$$\frac{\Psi \rhd_U \Delta; (\Sigma \setminus\setminus \{x\}); (x : \Sigma(x)) \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{inert}] \quad \Theta \rhd_U \emptyset \vdash^{(0,0,|v|_{\mathsf{nd}})} v : \Sigma(x)}{\Delta; (\Sigma \setminus\setminus \{x\}) \vdash^{(0,0,|q|_{\mathsf{nd}}+|v|_{\mathsf{nd}})} p@[x{\leftarrow}v] : [\mathsf{inert}]}\ \mathrm{ES}$$

- Let $\mathsf{uinert}(p)$ be derived as

$$\frac{\mathsf{uinert}(q) \quad x \notin (\mathsf{u}(q) \cup \mathsf{a}(q))}{\mathsf{uinert}(q@[x{\leftarrow}t])}\ \mathsf{I}_{\mathsf{GC}}$$

where $p = q@[x{\leftarrow}t]$. Note that $|p|_{\mathsf{nd}} = |q|_{\mathsf{nd}}$, $\mathsf{u}(p) = \mathsf{u}(q)$ and $\mathsf{a}(p) = \mathsf{a}(q)$, and so $\mathsf{dom}(f) = \mathsf{ul}\,(q)$. Thus, we can apply the *i.h.* on $\mathsf{uinert}(q)$ with respect to f to obtain a tight type derivation $\Psi \triangleright_U \Delta; \Sigma \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{inert}]$ such that $\mathsf{dom}(\Delta) = \mathsf{a}(q)$, $\mathsf{inert}(\Delta)$, $\mathsf{dom}(\Sigma) = \mathsf{ul}\,(q)$ and $\Sigma(z) \in f(z)$ for every $z \in \mathsf{dom}(\Sigma)$. Therefore, the statement holds by defining $\Gamma \coloneqq \Delta$, $\Pi \coloneqq \Sigma$, and Φ as follows:

$$\frac{\Psi \triangleright_U \Delta; \Sigma \vdash^{(0,0,|q|_{\mathsf{nd}})} q : [\mathsf{inert}] \quad (\Delta; \Sigma)(x) = \mathbf{0}}{\Delta; \Sigma \vdash^{(0,0,|q|_{\mathsf{nd}})} q@[x{\leftarrow}t] : [\mathsf{inert}]}\ \mathrm{ES}_{\mathrm{gc}}$$

□
□

(Click here to go back to main chapter.)

Proposition 13.6.18 (Tight typability of Useful Open CbNeed-normal forms)**.**
Let $p \in \mathcal{PR}$ be such that $\mathsf{unorm}(p)$. Then there exists a tight type derivation $\Phi \triangleright_U \Gamma \vdash^{(0,0,|p|_{\mathsf{nd}})} p : M$ such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.

Proof. *(Click here to go back to main chapter.)*
Let p be in $\rightarrow_{\mathsf{und}}$-normal form. We proceed by case analysis according to what is given by Proposition 8.2.7 (Syntactic characterization of Useful Open CbNeed-normal forms):

- If $\mathsf{genVar}_{\#}(p)$, then the statement follows by Lemma 9.1.14 (Tight typability of generalized variables).
- If $\mathsf{uabs}(p)$, then the statement follows by Lemma 9.1.15 (Tight typability of useful abstraction programs).
- Let $\mathsf{uinert}(p)$ and let $f : \mathsf{ul}\,(p) \to \{\mathsf{Inert}, \mathsf{Abs}\}$ be any given choice function. Then by Lemma 9.1.16 (Tight typability of useful inert programs) on $\mathsf{uinert}(p)$ with respect to f, there exists a tight type derivation $\Phi \triangleright_U \Gamma; \Pi \vdash^{(0,0,|p|_{\mathsf{nd}})} p : [\mathsf{inert}]$ such that $\mathsf{dom}(\Gamma) = \mathsf{a}(p)$, $\mathsf{inert}(\Gamma)$, $\mathsf{dom}(\Pi) = \mathsf{ul}\,(p)$ and $\Pi(z) \in f(z)$. The statement follows by noting that $\mathsf{dom}(\Gamma; \Pi) = \mathsf{dom}(\Gamma) \cup \mathsf{dom}(\Pi) = \mathsf{a}(p) \cup \mathsf{ul}\,(p) = \mathsf{u}(p) \cup \mathsf{a}(p) = \mathsf{nv}(p)$.

□

(Click here to go back to main chapter.)

Lemma 13.6.19 (Linear Removal for Useful Open CbNeed)**.**
Let $x \in \mathsf{Var}$ and $v \in \mathsf{Val}$ such that $x \notin \mathsf{fv}(v)$.

1. *Let $\mathcal{H}^{@}$ be such that $x \notin \mathsf{a}(\mathcal{H}^{@})$, and let*

$$\Phi \triangleright_U \Gamma; x : M \vdash^{(m,e,r)} \mathcal{H}^{@}\langle v \rangle : N$$

be such that $N \neq \mathbf{0}$, and $\mathsf{tight}_{\Gamma}(\mathsf{a}(\mathcal{H}^{@}))$.
Then there exist type derivations

$$\Psi \triangleright_U \Pi \vdash^{(m',e',r')} v : O$$
$$\Theta \triangleright_U \Delta; x : (M \uplus O) \vdash^{(m'',e'',r'')} \mathcal{H}^{@}\langle x \rangle : N$$

such that

- $\Gamma = \Pi \biguplus \Delta$, *and*
- $(m' + m'', e' + e'', r' + r'') = (m, e + 1, r)$.

2. *Let* $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ *be such that* $x \notin (\mathcal{A} \cup \mathsf{dom}(P))$ *and* $\mathsf{fv}(v) \cap \mathsf{dom}(P) = \emptyset$*, and let*

$$\Phi \rhd_U \Gamma; x : M \vdash^{(m,e,r)} P\langle v \rangle : N$$

be such that $N \neq \mathbf{0}$*, and* $\mathsf{tight}_\Gamma(\mathcal{A})$*.*
Then there exist multi type $O \neq \mathbf{0}$ *and type derivations*

$$\Psi \rhd_U \Pi \vdash^{(m',e',r')} v : O$$
$$\Theta \rhd_U \Delta; x : (M \uplus O) \vdash^{(m'',e'',r'')} P\langle x \rangle : N$$

such that

- $\Gamma = \Pi \biguplus \Delta$*, and*
- $(m' + m'', e' + e'', r' + r'') = (m, e + 1, r)$*.*

Proof. (Click here to go back to main chapter.)

1. By induction on the shape of $\mathcal{H}^{@}$:
   - Let $\mathcal{H} = (\langle\cdot\rangle\, t)$. We proceed by case analysis on the last typing rule in Φ:
     - Let Φ be derived as

$$\frac{\Xi \rhd_U \Gamma_1 \vdash^{(m_1,e_1,r_1)} v : [P \multimap N] \quad \Omega \rhd_U \Gamma_2; x : M \vdash^{(m_2,e_2,r_2)} t : P \quad P \neq \mathbf{0}}{(\Gamma_1 \uplus \Gamma_2); x : M \vdash^{(m_1+m_2+1,e_1+e_2,r_1+r_2)} v\, t : N}\ \mathsf{app}$$

       with $\Gamma = (\Gamma_1 \biguplus \Gamma_2)$ and $(m, e, r) = (m_1 + m_2 + 1, e_1 + e_2, r_1 + r_2)$. Note that $x \notin \mathsf{dom}(\Gamma_1)$—because of the fact that $x \notin \mathsf{fv}(v)$ and by Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system).
       The statement follows by taking $\Psi \coloneqq \Xi$ and deriving Θ as follows

$$\dfrac{\dfrac{[P \multimap N] \notin \mathsf{Tight}}{x : [P \multimap N] \vdash^{(0,1,0)} x : [P \multimap N]}\ \mathsf{ax} \qquad \Omega \rhd_U \Gamma_2; x : M \vdash^{(m_2,e_2,r_2)} t : P}{\Gamma_2; x : (M \uplus [P \multimap N]) \vdash^{(m_2+1,e_2+1,r_2)} x\, t : N}\ \mathsf{app}$$

       verifying in particular that

$$(m' + m'', e' + e'', r' + r'') = (m_1 + m_2 + 1, e_1 + e_2 + 1, r_1 + r_2) = (m, e + 1, r)$$

     - It is easy to verify that if Φ cannot have app_i as its last typing rule, since values are not inert-typable.
     - Let Φ be derived as

$$\frac{\Xi \rhd_U \Gamma \vdash^{(m_1,e_1,r_1)} v : [\mathbf{0} \multimap N]}{\Gamma \vdash^{(m_1+1,e_1,r_1)} v : [\mathbf{0} \multimap N]}\ \mathsf{app}_{\mathrm{gc}}$$

       with $M = [\mathbf{0} \multimap N]$ and $(m, e, r) = (m_1 + 1, e_1, r_1)$. Note that $x \notin \mathsf{dom}(\Gamma)$—by Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system), recalling that $x \notin \mathsf{fv}(v)$.
       The statement follows by taking $\Psi \coloneqq \Xi$ and deriving Θ as follows

$$\dfrac{\dfrac{[\mathbf{0} \multimap N] \notin \mathsf{Tight}}{x : [\mathbf{0} \multimap N] \vdash^{(0,1,0)} x : [\mathbf{0} \multimap N]}\ \mathsf{ax}}{x : [\mathbf{0} \multimap N] \vdash^{(1,1,0)} x\, t : N}\ \mathsf{app}_{\mathrm{gc}}$$

       with $\Delta = \emptyset$ and $(m'', e'', r'') = (1, 1, 0)$ verifying that

$*$ $\Gamma = \emptyset \biguplus \Gamma = \Delta \biguplus \Pi$, and
$*$ $(m' + m'', e' + e'', r' + r'') = (m_1 + 1, e_1 + 1, r_1) = (m, e + 1, r)$.

- Let $\mathcal{H}^{@} = \mathcal{J}^{@} t$. We proceed by case analysis on the last typing rule in Φ:
  - Let Φ be derived as

$$\frac{\Gamma_1; x : M_1 \vdash^{(m_1,e_1,r_1)} \mathcal{J}^{@}\langle v\rangle : [P \multimap N] \quad \Gamma_2; x : M_2 \vdash^{(m_2,e_2,r_2)} t : P \quad P \neq \mathbf{0}}{(\Gamma_1 \biguplus \Gamma_2); x : (M_1 \uplus M_2) \vdash^{(m_1+m_2+1,e_1+e_2,r_1+r_2)} \mathcal{J}^{@}\langle v\rangle t : N}\ \mathsf{app}$$

    with $\Gamma = \Gamma_1 \biguplus \Gamma_2$, $M = M_1 + M_2$ and $(m, e, r) = (m_1 + m_2 + 1, e_1 + e_2, r_1 + r_2)$.
    Since $\mathsf{a}(\mathcal{H}^{@}) = \mathsf{a}(\mathcal{J}^{@})$, then we can infer from $\mathsf{tight}_{\Gamma}(\mathsf{a}(\mathcal{H}^{@}))$ that $\mathsf{tight}_{\Gamma_1}(\mathsf{a}(\mathcal{J}^{@}))$.
    Hence, we can apply the *i.h.* on Ψ to obtain type derivations $\Psi' \triangleright_U \Pi' \vdash^{(m_1',e_1',r_1')} v : O'$ and $\Theta' \triangleright_U \Delta'; x : (M_1 \uplus O') \vdash^{(m_1'',e_1'',r_1'')} \mathcal{J}^{@}\langle x\rangle : [P \multimap N]$ such that $\Gamma_1 = \Pi' \biguplus \Delta'$ and $(m_1' + m_1'', e_1' + e_1'', r_1' + r_1'') = (m_1, e_1 + 1, r_1)$.
    Therefore, the statement follows by taking $\Psi \coloneqq \Psi'$ and deriving Θ as follows

$$\frac{\Delta'; x : (M_1 \uplus O') \vdash^{(m_1'',e_1'',r_1'')} \mathcal{J}^{@}\langle x\rangle : [P \multimap N] \quad \Gamma_2; x : M_2 \vdash^{(m_2,e_2,r_2)} t : P}{(\Delta' \biguplus \Gamma_2); x : (M_1 \uplus O' \uplus M_2) \vdash^{(m_1''+m_2+1,e_1''+e_2,r_1''+r_2)} \mathcal{J}^{@}\langle x\rangle t : N}\ \mathsf{app}$$

    verifying that
    $*$ $\Gamma = \Gamma_1 \biguplus \Gamma_2 = (\Pi' \biguplus \Delta') \biguplus \Gamma_2 = \Pi' \biguplus (\Delta' \biguplus \Gamma_2) = \Pi \biguplus \Delta$, and
    $*$ $(m' + m'', e' + e'', r' + r'') = (m_1' + (m_1'' + m_2 + 1), e_1' + (e_1'' + e_2), r_1' + (r_1'' + r_2)) = (m_1 + m_2 + 1, e_1 + e_2 + 1, r_1 + r_2) = (m, e + 1, r)$.
  - The case where app_i is the last typing rule in Φ can be proven similarly to app.
  - The case where $\mathsf{app}_{\mathrm{gc}}$ is the last typing rule in Φ can be proven similarly to the two previous cases.
- Let $\mathcal{H}^{@} = i\mathcal{J}^{@}$. We proceed by case analysis on the last typing rule in Φ:
  - Suppose Φ is derived as follows:

$$\frac{\Xi \triangleright_U \Gamma_1; x : M_1 \vdash^{(m_1,e_1,r_1)} i : [P \multimap N] \quad \Omega \triangleright_U \Gamma_2; x : M_2 \vdash^{(m_2,e_2,r_2)} \mathcal{J}^{@}\langle v\rangle : P \quad P \neq \mathbf{0}}{(\Gamma_1 \uplus \Gamma_2); x : (M_1 \uplus M_2) \vdash^{(m_1+m_2+1,e_1+e_2,r_1+r_2)} i\mathcal{J}^{@}\langle v\rangle : N}\ \mathsf{app}$$

    with $\Gamma = (\Gamma_1 \biguplus \Gamma_2)$. Note that since $x \notin \mathsf{a}(\mathcal{H}^{@})$, then $x \notin \mathsf{a}(i)$ and so $\mathsf{tight}_{\Gamma}(\mathsf{a}(\mathcal{H}^{@}))$ would imply that $\mathsf{tight}_{\Gamma}(\mathsf{a}(i))$. However, we would now be able to apply Lemma 9.1.3 (Typing properties of useful inert terms) on Ξ to obtain that $[P \multimap N] \in \mathsf{Inert}$, which is absurd. Hence, this case is impossible.
  - Let Φ be derived as

$$\frac{\Xi \triangleright_U \Gamma_1; x : M_1 \vdash^{(m_1,e_1,r_1)} i : [\mathsf{inert}]_{i\in I} \quad \Omega \triangleright_U \Gamma_2; x : M_2 \vdash^{(m_2,e_2,r_2)} \mathcal{J}^{@}\langle v\rangle : [\mathsf{tight}] \quad I \neq \emptyset}{(\Gamma_1 \uplus \Gamma_2); x : (M_1 \uplus M_2) \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2+1)} i\mathcal{J}^{@}\langle v\rangle : [\mathsf{inert}]_{i\in I}}\ \mathsf{app}_i$$

    with $\Gamma = (\Gamma_1 \biguplus \Gamma_2)$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2 + 1)$.
    Note that $\mathsf{tight}_{\Gamma}(\mathsf{a}(\mathcal{H}^{@}))$ implies that $\mathsf{tight}_{\Gamma_2}(\mathsf{a}(\mathcal{J}^{@}))$. Hence, we can apply the *i.h.* on Ω to obtain type derivations $\Psi' \triangleright_U \Pi' \vdash^{(m_2',e_2',r_2')} v : O'$ and $\Theta' \triangleright_U \Delta'; x : (M_2 \uplus O') \vdash^{(m_2'',e_2'',r_2'')} \mathcal{J}^{@}\langle x\rangle : [\mathsf{tight}]$ such that $\Gamma_2 = \Pi' \biguplus \Delta'$ and $(m_2' + m_2'', e_2' + e_2'', r_2' + r_2'') = (m_2, e_2 + 1, r_2)$.
    Therefore, the statement follows by taking $\Psi \coloneqq \Psi'$ and deriving Θ as follows

$$\frac{\Xi \triangleright_U \Gamma_1; x : M_1 \vdash^{(m_1,e_1,r_1)} i : [\mathsf{inert}]_{i\in I} \quad \Theta' \triangleright_U \Delta'; x : (M_2 \uplus O') \vdash^{(m_2'',e_2'',r_2'')} \mathcal{J}^{@}\langle x\rangle : [\mathsf{tight}]}{(\Gamma_1 \uplus \Delta'); x : (M_1 \uplus M_2 \uplus O') \vdash^{(m_1+m_2'',e_1+e_2'',r_1+r_2''+1)} i\mathcal{J}^{@}\langle x\rangle : [\mathsf{inert}]_{i\in I}}\ \mathsf{app}_i$$

    verifying that

* $\Gamma = \Gamma_1 \biguplus \Gamma_2 = \Gamma_1 \biguplus (\Pi' \biguplus \Delta') = (\Gamma_1 \biguplus \Delta') \biguplus \Pi' = \Delta \biguplus \Pi$, and
* $(m' + m'', e' + e'', r' + r'') = (m'_2 + (m_1 + m''_2), e'_2 + (e_1 + e''_2), r'_2 + (r_1 + r''_2 + 1)) = (m_1 + m_2, e_1 + e_2 + 1, r_1 + r_2 + 1) = (m, e + 1, r)$.

– The case where $\mathsf{app}_{\mathrm{gc}}$ is the last typing rule in Φ is impossible, which can be proven similarly to the case of app.

2. By induction on the derivation of $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$:
   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{}{(\mathcal{H}^{@}, \epsilon) \in \mathcal{E}^{@}_{\mathsf{u}(\mathcal{H}^{@}),\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_1}$$

   where $P = (\mathcal{H}^{@}, \epsilon)$, $\mathcal{U} = \mathsf{u}(\mathcal{H}^{@})$ and $\mathcal{A} = \mathsf{a}(\mathcal{H}^{@})$. Then Φ must be of the form

$$\frac{\Phi' \rhd_U \Gamma; x : M \vdash^{(m,e,r)} \mathcal{H}^{@}\langle v\rangle : N}{\Gamma; x : M \vdash^{(m,e,r)} (\mathcal{H}^{@}\langle v\rangle, \epsilon) : N}\ \mathsf{Lift}$$

   and we can apply Lemma 9.1.18.1 (Useful linear removal for term contexts) on Φ' to obtain type derivations $\Psi' \rhd_U \Pi' \vdash^{(m'_1,e'_1,r'_1)} v : O'$ and $\Theta' \rhd_U \Delta'; x : (M \uplus O') \vdash^{(m''_1,e''_1,r''_1)} \mathcal{H}^{@}\langle x\rangle : N$ such that $\Gamma = \Pi' \biguplus \Delta'$ and $(m'_1 + m''_1, e'_1 + e''_1, r'_1 + r''_1) = (m, e + 1, r)$.
   Therefore, the statement follows by taking $\Psi \coloneqq \Psi'$ and deriving Θ as follows

$$\frac{\Theta' \rhd_U \Delta'; x : (M \uplus O') \vdash^{(m''_1,e''_1,r''_1)} \mathcal{H}^{@}\langle x\rangle : N}{\Delta'; x : (M \uplus O') \vdash^{(m''_1,e''_1,r''_1)} \mathcal{H}^{@}\langle x\rangle : N}\ \mathsf{Lift}$$

   - Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle y\rangle @ [y \leftarrow \mathcal{H}^{@}] \in \mathcal{E}^{@}_{\mathcal{V} \cup \mathsf{u}(\mathcal{H}^{@}),\mathcal{B} \cup \mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_2}$$

   where $P = Q\langle y\rangle @ [y \leftarrow \mathcal{H}^{@}]$, $\mathcal{U} = \mathcal{V} \cup \mathsf{u}(\mathcal{H}^{@})$ and $\mathcal{A} = \mathcal{B} \cup \mathsf{a}(\mathcal{H}^{@})$. We may safely assume that $x \neq y$—by α-conversion. We proceed by case analysis on the last typing rule in Φ:
   – Let Φ be of the form

$$\frac{\Xi \rhd_U \Gamma_1; x : M_1; y : P \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : N \quad \Omega \rhd_U \Gamma_2; x : M_2 \vdash^{(m_2,e_2,r_2)} \mathcal{H}^{@}\langle v\rangle : P \quad P \neq \mathbf{0}}{(\Gamma_1 \biguplus \Gamma_2); x : (M_1 \uplus M_2) \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle y\rangle @ [y \leftarrow \mathcal{H}^{@}\langle v\rangle] : N}\ \mathrm{ES}$$

   with $\Gamma = \Gamma_1 \biguplus \Gamma_2$, $M = M_1 \biguplus M_2$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$.
   Note that $\mathsf{tight}_{\Gamma}(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi}(\mathcal{B})$. Thus, we can apply the Lemma 9.1.18.1 (Useful linear removal for term contexts) on Ω to obtain type derivations $\Psi' \rhd_U \Pi' \vdash^{(m'_2,e'_2,r'_2)} v : O'$ and $\Theta' \rhd_U \Delta'; x : (M_2 \biguplus O') \vdash^{(m''_2,e''_2,r''_2)} \mathcal{H}^{@}\langle x\rangle : P$ such that $\Gamma_2 = \Pi' \biguplus \Delta'$ and $(m'_2 + m''_2, e'_2 + e''_2, r'_2 + r''_2) = (m_2, e_2 + 1, r_2)$.
   Therefore, the statement follows by taking $\Psi \coloneqq \Psi'$ and deriving Θ as follows

$$\frac{\Xi \rhd_U \Gamma_1; x : M_1; y : P \vdash^{(m_1,e_1,r_1)} P\langle y\rangle : N \quad \Theta' \rhd_U \Delta'; x : (M_2 \biguplus O') \vdash^{(m''_2,e''_2,r''_2)} \mathcal{H}^{@}\langle x\rangle : P}{(\Gamma_1 \biguplus \Delta'); x : (M_1 \uplus M_2 \uplus O') \vdash^{(m_1+m''_2,e_1+e''_2,r_1+r''_2)} Q\langle y\rangle @ [y \leftarrow \mathcal{H}^{@}\langle v\rangle] : N}\ \mathrm{ES}$$

   verifying that
   * $M = M_1 \uplus M_2 \uplus O' = M \uplus O$,
   * $\Gamma = \Gamma_1 \biguplus \Gamma_2 = \Gamma_1 \biguplus (\Pi' \biguplus \Delta') = (\Gamma_1 \biguplus \Delta') \biguplus \Pi' = \Delta \biguplus \Pi$, and

$*$ $(m' + m'', e' + e'', r' + r'') = (m'_2 + (m_1 + m''_2), e'_2 + (e_1 + e''_2), r'_2 + (r_1 + r''_2)) = (m_1 + m_2, e_1 + e_2 + 1, r_1 + r_2) = (m, e + 1, r)$.

– Suppose Φ is of the form

$$\frac{\Phi' \rhd_U \Gamma; x : M \vdash^{(m,e,r)} Q\langle y\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}\langle v\rangle] : N}\ \mathrm{ES}_{\mathrm{gc}}$$

However, an application of Lemma 13.6.8.2 (Plugged variables and domain of type contexts) on Φ' would gives us that $y \in \mathsf{dom}(\Gamma; x : M)$, making this case absurd.

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}z] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{V},y,z),\mathsf{upd}(\mathcal{B},y,z)}}\ \mathsf{E}_{\mathsf{VAR}}$$

where $P = Q@[y{\leftarrow}z]$, $\mathcal{U} = \mathsf{upd}(\mathcal{V}, y, z)$ and $\mathcal{A} = \mathsf{upd}(\mathcal{B}, y, z)$. Note that $x \neq y$—since $x \notin \mathsf{dom}(P)$—and that $\mathsf{fv}(v) \cap \mathsf{dom}(P) = \emptyset$—by hypothesis. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Xi \rhd_U \Gamma_1; x : M_1; y : P \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : N \quad \Omega \rhd_U \Gamma_2; x : M_2 \vdash^{(m_2,e_2,r_2)} z : P \quad P \neq \mathbf{0}}{(\Gamma_1 \biguplus \Gamma_2); x : (M_1 \uplus M_2) \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle v\rangle@[y{\leftarrow}z] : N}\ \mathrm{ES}$$

with $\Gamma = \Gamma_1 \biguplus \Gamma_2$, $M = M_1 \biguplus M_2$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Two further determine the content of Φ, we need to split the analysis into two sub-cases:

$*$ Let $z = x$. Then Φ is of the form

$$\frac{\Xi \rhd_U \Gamma_1; x : M_1; y : M_2 \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : N \quad \overline{\Omega \rhd_U x : M_2 \vdash^{(m_2,e_2,r_2)} x : M_2}}{\Gamma_1; x : (M_1 \uplus M_2) \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle v\rangle@[y{\leftarrow}x] : N}\ \mathrm{ES}$$

with $P = M_2$ and $\Gamma_2 = \emptyset$. Note that if $y \in \mathcal{B}$, then $\mathcal{A} = (\mathcal{B} \setminus \{y\}) \cup \{x\}$ and so $M_2 \in \mathsf{Tight}$ by hypothesis. Thus, $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Gamma_1; x : M_1; y : M_2}(\mathcal{B})$ and then we can apply the *i.h.* on Ξ to obtain type derivations $\Psi' \rhd_U \Pi' \vdash^{(m'_1,e'_1,r'_1)} v : O'$ and $\Theta' \rhd_U \Delta'; x : (M_1 \uplus O'); y : M_2 \vdash^{(m''_1,e''_1,r''_1)} Q\langle x\rangle : N$ such that $\Gamma_1; y : M_2 = \Pi' \biguplus (\Delta'; y : M_2)$ and $(m'_1 + m''_1, e'_1 + e''_1, r'_1 + r''_1) = (m_1, e_1, r_1)$. Note that $y \notin \mathsf{fv}(v)$ by hypothesis, and so $y \notin \mathsf{dom}(\Pi') \subseteq \mathsf{fv}(v)$—by Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system).
Therefore, the statement follows by taking $\Psi := \Psi'$ and deriving Θ as follows

$$\frac{\Theta' \rhd_U \Delta'; x : (M_1 \uplus O'); y : M_2 \vdash^{(m''_1,e''_1,r''_1)} Q\langle x\rangle : N \quad \overline{\Omega \rhd_U x : M_2 \vdash^{(m_2,e_2,r_2)} x : M_2}}{\Delta'; x : (M_1 \uplus O' \uplus M_2) \vdash^{(m''_1+m_2,e''_1+e_2,r''_1+r_2)} Q\langle x\rangle@[y{\leftarrow}x] : N}\ \mathrm{ES}$$

verifying that

· $M = M_1 \uplus O' \uplus M_2 = M \uplus O$,

· $\Gamma = \Gamma_1 \biguplus \Gamma_2 = ((\Pi' \biguplus (\Delta'; y : M_2)) \setminus\setminus \{y\}) \biguplus \emptyset = (\Pi' \biguplus \Delta') = \Pi \biguplus \Delta$, and

· $(m' + m'', e' + e'', r' + r'') = (m'_1 + (m''_1 + m_2), e'_1 + (e''_1 + e_2), r'_1 + (r''_1 + r_2)) = (m_1 + m_2, e_1 + e_2 + 1, r_1 + r_2) = (m, e + 1, r)$.

∗ Let $z \neq x$. Then Φ is of the form

$$\frac{\Xi \rhd_U \Gamma_1'; z : Q; y : P; x : M \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : N \quad \overline{\Omega \rhd_U z : P \vdash^{(m_2,e_2,r_2)} z : P}}{\Gamma_1'; z : (Q \uplus P); x : M \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle v\rangle@[y{\leftarrow}z] : N}\ \text{ES}$$

That is, $\Gamma_1 = \Gamma_1'; z : (Q \uplus P)$, $\Gamma_2 = \{z : P\}$, $M_1 = M$ and $M_2 = \mathbf{0}$.
Note that if $y \in \mathcal{B}$, then $\mathcal{A} = (\mathcal{B} \backslash \{y\}) \cup \{z\}$ and so $(Q \uplus P) \in \mathsf{Tight}$ by hypothesis. Thus, $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Gamma_1; x : M_1; y : P; z : Q}(\mathcal{B})$ and then we can apply the *i.h.* on Ξ to obtain type derivations $\Psi' \rhd_U \Pi'; z : Q_1 \vdash^{(m_1',e_1',r_1')} v : O$ and

$$\Theta' \rhd_U \Delta'; z : Q_2; y : P; x : (M \uplus O') \vdash^{(m_1'',e_1'',r_1'')} Q\langle x\rangle : N$$

such that $Q = Q_1 \uplus Q_2$, $\Gamma_1'; z : Q; y : P = (\Pi'; z : Q_1) \biguplus (\Delta'; z : Q_2; y : P)$ and $(m_1' + m_1'', e_1' + e_1'', r_1' + r_1'') = (m_1, e_1 + 1, r_1)$. Note that $y \notin \mathsf{dom}(\Pi'; z : Q_1)$—by Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system) and the fact that $\mathsf{fv}(v) \cap \mathsf{dom}(P) = \emptyset$.
Therefore, the statement follows by taking $\Psi := \Psi'$ and deriving Θ as follows

$$\frac{\Theta' \rhd_U \Delta'; z : Q_2; y : P; x : (M \uplus O') \vdash^{(m_1'',e_1'',r_1'')} Q\langle x\rangle : N \quad \overline{\Omega \rhd_U z : P \vdash^{(m_2,e_2,r_2)} z : P}}{\Delta'; z : (Q_2 \uplus P); x : (M \uplus O') \vdash^{(m_1''+m_2,e_1''+e_2,r_1''+r_2)} Q\langle x\rangle@[y{\leftarrow}z] : N}\ \text{ES}$$

verifying that

- $M \uplus O' = M \uplus O$,
- $\Gamma = \Gamma_1 \biguplus \Gamma_2 = (\Gamma_1'; z : (Q \uplus P)) \biguplus \emptyset = ((\Gamma_1'; z : Q; y : P) \setminus\setminus \{y\}) \biguplus \{z : P\} = (((\Pi'; z : Q_1) \biguplus (\Delta'; z : Q_2; y : P)) \setminus\setminus \{y\}) \biguplus \{z : P\} = (\Pi'; z : Q_1) \biguplus (\Delta'; z : (Q_2 \uplus P)) = \Pi \biguplus \Delta$, and
- $(m' + m'', e' + e'', r' + r'') = (m_1' + (m_1'' + m_2), e_1' + (e_1'' + e_2), r_1' + (r_1'' + r_2)) = (m_1 + m_2, e_1 + 1 + e_2, r_1 + r_2) = (m, e + 1, r)$.

– Let Φ be of the form

$$\frac{\Phi' \rhd_U \Gamma; x : M \vdash^{(m,e,r)} Q\langle v\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle v\rangle@[y{\leftarrow}z] : N}\ \text{ES}_{\text{gc}}$$

Since $\mathsf{tight}_\Gamma(\mathcal{A})$ and $y \notin \mathsf{dom}(\Gamma; x : M)$ imply that $\mathsf{tight}_\Gamma(\mathcal{B})$, then we can apply the *i.h.* on Φ' and easily prove the statement.

• Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad y \in (\mathcal{U} \cup \mathcal{A})}{Q@[y{\leftarrow}i^+] \in \mathcal{E}^{@}_{(\mathcal{U}\backslash\{y\}) \cup \mathsf{u}(i^+), (\mathcal{A}\backslash\{y\}) \cup \mathsf{a}(i^+)}}\ \mathsf{E_I}$$

where $P = Q@[x{\leftarrow}i^+]$, $\mathcal{U} = (\mathcal{U} \setminus \{x\}) \cup \mathsf{u}(i^+)$ and $\mathcal{A} = (\mathcal{A} \setminus \{x\}) \cup \mathsf{a}(i^+)$. Note that $x \neq y$—since $x \notin \mathsf{dom}(P)$—and that $\mathsf{fv}(v) \cap \mathsf{dom}(P) = \emptyset$—by hypothesis—yielding that $y \notin \mathsf{fv}(v)$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Xi \rhd_U \Gamma_1; y : P; x : M_1 \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : N \quad \Omega \rhd_U \Gamma_2; x : M_2 \vdash^{(m_2,e_2,r_2)} i^+ : P \quad P \neq \mathbf{0}}{(\Gamma_1 \biguplus \Gamma_2); x : (M_1 \uplus M_2) \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle v\rangle@[y{\leftarrow}i^+] : N}\ \text{ES}$$

with $\Gamma = \Gamma_1 \biguplus \Gamma_2$, $M = M_1 \biguplus M_2$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$.
Note that $x \notin \mathcal{A}$, and so $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Gamma_2; x:M_2}(\mathsf{a}(i^+))$. Hence, we can apply Lemma 9.1.3 (Typing properties of useful inert terms) on Ω to obtain that $P \in \mathsf{Inert}$. In turn, this allows us to infer from $\mathsf{tight}_\Gamma(\mathcal{A})$ that $\mathsf{tight}_{\Gamma_1; y:P}(\mathcal{B})$. Thus, we can apply the *i.h.* on Ξ to obtain type derivations $\Psi' \rhd_U \Pi' \vdash^{(m_1', e_1', r_1')} v : O'$ and

$$\Theta' \rhd_U \Delta'; y : P; x : (M_1 \uplus O') \vdash^{(m_1'', e_1'', r_1'')} Q\langle x \rangle : N$$

such that $\Gamma_1; y : P = \Pi' \biguplus (\Delta'; y : P)$ and $(m_1' + m_1'', e_1' + e_1'', r_1' + r_1'') = (m_1, e + 1, r_1)$.
Note that $x \notin \mathsf{dom}(\Pi')$—by Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system) and given that $x \notin \mathsf{fv}(v)$.
Therefore, the statement follows by taking $\Psi \coloneqq \Psi'$ and deriving Θ as follows

$$\frac{\Theta' \rhd_U \Delta'; y : P; x : (M_1 \uplus O') \vdash^{(m_1'', e_1'', r_1'')} Q\langle x \rangle : N \quad \Omega \rhd_U \Gamma_2; x : M_2 \vdash^{(m_2, e_2, r_2)} i^+ : P}{(\Delta' \biguplus \Gamma_2); x : (M_1 \uplus O' \uplus M_2) \vdash^{(m_1'' + m_2, e_1'' + e_2, r_1'' + r_2)} Q\langle x \rangle @ [y \leftarrow i^+] : N}\ \mathrm{ES}$$

verifying that
* $M_1 \uplus O' \uplus M_2 = M \uplus O$,
* $\Gamma = \Gamma_1 \biguplus \Gamma_2 = ((\Pi' \biguplus (\Delta'; y : P)) \setminus\setminus \{y\}) \biguplus \Gamma_2 = \Pi' \biguplus (\Delta' \biguplus \Gamma_2) = \Pi \biguplus \Delta$, and
* $(m' + m'', e' + e'', r' + r'') = (m_1' + (m_1'' + m_2), e_1' + (e_1'' + e_2), r_1' + (r_1'' + r_2)) = (m_1 + m_2, e_1 + 1 + e_2, r_1 + r_2) = (m, e + 1, r)$.

– Let Φ be of the form

$$\frac{\Phi' \rhd_U \Gamma; x : M \vdash^{(m,e,r)} Q\langle v \rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle v \rangle @ [y \leftarrow i^+] : N}\ \mathrm{ES}_{\mathrm{gc}}$$

Since $\mathsf{tight}_\Gamma(\mathcal{A})$ and $y \notin \mathsf{dom}(\Gamma; x : M)$ imply that $\mathsf{tight}_\Gamma(\mathcal{B})$, then we can apply the *i.h.* on Φ' and easily prove the statement.

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{Q@[y \leftarrow t] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E}_{\mathsf{GC}}$$

where $P = Q@[y \leftarrow t]$. Note that $x \neq y$—since $x \notin \mathsf{dom}(P)$—and that $\mathsf{fv}(v) \cap \mathsf{dom}(P) = \emptyset$—by hypothesis—yielding that $y \notin \mathsf{fv}(v)$. We proceed by case analysis on the last typing rule in Φ:
– Let Φ be of the form

$$\frac{\Xi \rhd_U \Gamma_1; y : P; x : M_1 \vdash^{(m_1, e_1, r_1)} Q\langle v \rangle : N \quad \Omega \rhd_U \Gamma_2; x : M_2 \vdash^{(m_2, e_2, r_2)} t : P \quad P \neq \mathbf{0}}{(\Gamma_1 \biguplus \Gamma_2); x : (M_1 \uplus M_2) \vdash^{(m_1 + m_2, e_1 + e_2, r_1 + r_2)} Q\langle v \rangle @ [y \leftarrow t] : N}\ \mathrm{ES}$$

with $\Gamma = \Gamma_1 \biguplus \Gamma_2$, $M = M_1 \biguplus M_2$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$.
Note that the fact that $\mathsf{tight}_\Gamma(\mathcal{A})$ and that $y \notin \mathcal{A}$ imply that $\mathsf{tight}_{\Gamma_1; y:P}(\mathcal{A})$. Hence, we can apply the *i.h.* on Ξ to obtain type derivations $\Psi' \rhd_U \Pi' \vdash^{(m_1', e_1', r_1')} v : O$ and

$$\Theta' \rhd_U \Delta'; y : P; x : (M_1 \uplus O') \vdash^{(m_1'', e_1'', r_1'')} Q\langle x \rangle : N$$

such that $\Gamma_1; y : P = \Pi' \biguplus (\Delta'; y : P)$ and $(m_1' + m_1'', e_1' + e_1'', r_1' + r_1'') = (m_1, e_1 + 1, r_1)$.
Note that $y \notin \mathsf{dom}(\Pi')$—by Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system), and given that $y \notin \mathsf{fv}(v)$.

Therefore, the statement follows by taking $\Psi \coloneqq \Psi'$ and deriving Θ as follows

$$\frac{\Theta' \rhd_U \Delta'; y : P; x : (M_1 \uplus O') \vdash^{(m_1'', e_1'', r_1'')} Q\langle x \rangle : N \quad \Omega \rhd_U \Gamma_2; x : M_2 \vdash^{(m_2, e_2, r_2)} t : P}{(\Delta' \biguplus \Gamma_2); x : (M_1 \uplus O' \uplus M_2) \vdash^{(m_1'' + m_2, e_1'' + e_2, r_1'' + r_2)} Q\langle x \rangle @ [y \leftarrow t] : N}\ \text{ES}$$

verifying that
* $M_1 \uplus O' \uplus M_2 = M \uplus O$,
* $\Gamma = \Gamma_1 \biguplus \Gamma_2 = ((\Pi' \biguplus (\Delta'; y : P)) \setminus\setminus \{y\}) \biguplus \Gamma_2 = \Pi' \biguplus (\Delta' \biguplus \Gamma_2) = \Pi \biguplus \Delta$, and
* $(m' + m'', e' + e'', r' + r'') = (m_1' + (m_1'' + m_2), e_1' + (e_1'' + e_2), r_1' + (r_1'' + r_2)) = (m_1 + m_2, e_1 + 1 + e_2, r_1 + r_2) = (m, e + 1, r)$.

– Let Φ be of the form

$$\frac{\Phi' \rhd_U \Gamma; x : M \vdash^{(m,e,r)} Q\langle v \rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle v \rangle @ [y \leftarrow t] : N}\ \text{ES}_{\text{gc}}$$

We can apply the *i.h.* on Φ' and easily prove the statement.

• Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad x \in (\mathcal{V} \setminus \mathcal{B})}{Q @ [y \leftarrow w] \in \mathcal{E}^{@}_{\mathcal{V} \setminus \{x\},\mathcal{B}}}\ \mathsf{E_U}$$

where $P = Q@[y \leftarrow w]$, $\mathcal{U} = \mathcal{V} \setminus \{x\}$ and $\mathcal{A} = \mathcal{B}$. Note that $x \neq y$—since $x \notin \mathsf{dom}(P)$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Xi \rhd_U \Gamma_1; x : M_1; y : P \vdash^{(m_1, e_1, r_1)} Q\langle v \rangle : N \quad \Omega \rhd_U \Gamma_2; x : M_2 \vdash^{(m_2, e_2, r_2)} w : P \quad P \neq \mathbf{0}}{(\Gamma_1 \biguplus \Gamma_2); x : (M_1 \uplus M_2) \vdash^{(m_1 + m_2, e_1 + e_2, r_1 + r_2)} Q\langle v \rangle @ [y \leftarrow w] : N}\ \text{ES}$$

with $\Gamma = \Gamma_1 \biguplus \Gamma_2$, $M = M_1 \biguplus M_2$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that the facts that $\mathsf{tight}_\Gamma(\mathcal{A})$ and that $y \notin \mathcal{A}$ imply that $\mathsf{tight}_{\Gamma_1; y : P}(\mathcal{A})$. Hence, we can apply the *i.h.* on Ξ to obtain type derivations $\Psi' \rhd_U \Pi' \vdash^{(m_1', e_1', r_1')} v : O$ and

$$\Theta' \rhd_U \Delta'; y : P; x : (M_1 \uplus O') \vdash^{(m_1'', e_1'', r_1'')} Q\langle x \rangle : N$$

such that $\Gamma_1; y : P = \Pi' \biguplus (\Delta'; y : P)$ and $(m_1' + m_1'', e_1' + e_1'', r_1' + r_1'') = (m_1, e_1 + 1, r_1)$. Note that $y \notin \mathsf{dom}(\Pi')$—by Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system) and given that $y \notin \mathsf{fv}(v)$.
Therefore, the statement follows by taking $\Psi \coloneqq \Psi'$ and deriving Θ as follows

$$\frac{\Theta' \rhd_U \Delta'; y : P; x : (M_1 \uplus O') \vdash^{(m_1'', e_1'', r_1'')} Q\langle x \rangle : N \quad \Omega \rhd_U \Gamma_2; x : M_2 \vdash^{(m_2, e_2, r_2)} w : P}{(\Delta' \biguplus \Gamma_2); x : (M_1 \uplus O' \uplus M_2) \vdash^{(m_1'' + m_2, e_1'' + e_2, r_1'' + r_2)} Q\langle x \rangle @ [y \leftarrow w] : N}\ \text{ES}$$

verifying that
* $M_1 \uplus O' \uplus M_2 = M \uplus O$,
* $\Gamma = \Gamma_1 \biguplus \Gamma_2 = ((\Pi' \biguplus (\Delta'; y : P)) \setminus\setminus \{y\}) \biguplus \Gamma_2 = \Pi' \biguplus (\Delta' \biguplus \Gamma_2) = \Pi \biguplus \Delta$, and
* $(m' + m'', e' + e'', r' + r'') = (m_1' + (m_1'' + m_2), e_1' + (e_1'' + e_2), r_1' + (r_1'' + r_2)) = (m_1 + m_2, e_1 + 1 + e_2, r_1 + r_2) = (m, e + 1, r)$.

– Let Φ be of the form

$$\frac{\Phi' \rhd_U \Gamma; x : M \vdash^{(m,e,r)} Q\langle v\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle v\rangle@[y{\leftarrow}w] : N}\ \mathrm{ES}_{\mathrm{gc}}$$

Since $\mathsf{tight}_\Gamma(\mathcal{A})$ and $y \notin \mathsf{dom}(\Gamma; x : M)$ imply that $\mathsf{tight}_\Gamma(\mathcal{B})$, then we can apply the *i.h.* on Φ' and easily prove the statement.

- Let $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad y \notin (\mathcal{U} \cup \mathcal{A})}{Q\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle] \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}}\ \mathsf{E_{NL}}$$

where $P = Q\langle y\rangle@[y{\leftarrow}\langle\cdot\rangle]$. We may assume that $x \neq y$—by α-conversion. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Xi \rhd_U \Gamma_1; x : M; y : P \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : N \quad \Omega \rhd_U \Gamma_2 \vdash^{(m_2,e_2,r_2)} v : P \quad P \neq \mathbf{0}}{(\Gamma_1 \uplus \Gamma_2); x : M \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle y\rangle@[y{\leftarrow}v] : N}\ \mathrm{ES}$$

with $\Gamma = \Gamma_1 \uplus \Gamma_2$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that $x \notin \mathsf{dom}(\Gamma_2)$, because $x \notin \mathsf{fv}(v)$—and applying Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system).
Moreover, since $x, y \notin \mathcal{A}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Gamma_1; x : M; y : P}(\mathcal{A})$. Consequently, we can apply Lemma 13.6.8.2 (Plugged variables and domain of type contexts) on Ξ to obtain that

$$(\Gamma_1; x : M; y : P)(y) = P \notin \mathsf{Abs}$$

Given that values cannot be inert-typed, then $P \notin \mathsf{Tight}$.
Note that we can rewrite $\Omega \rhd_U \Gamma_2; x : \mathbf{0} \vdash^{(m_2,e_2,r_2)} v : P$ and can thus make the statement follow by taking $\Psi \coloneqq \Psi' \rhd_U \Pi' \vdash^{(m'_2,e'_2,r'_2)} v : O'$, with $\Pi' = \Gamma_2$, $(m'_2, e'_2, r'_2) = (m_2, e_2, r_2)$ and $O' = P$, and deriving Θ as follow:

$$\frac{\Theta' \rhd_U \Gamma_1; x : M; y : O' \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : N \quad \dfrac{O' \notin \mathsf{Tight}}{x : O' \vdash^{(0,1,0)} x : O'}\ \mathsf{ax}}{\Gamma_1; x : (M \uplus O') \vdash^{(m_1,e_1+1,r_1)} Q\langle y\rangle@[y{\leftarrow}x] : N}\ \mathrm{ES}$$

verifying that
  * $M \uplus O' = M \uplus O$,
  * $\Gamma = \Gamma_1 \uplus \Gamma_2 = \Pi \uplus \Delta$
  * $(m' + m'', e' + e'', r' + r'') = (m_2 + m_1, e_2 + e_1 + 1, r_2 + r_1) = (m, e + 1, r)$

– Suppose Φ is of the form

$$\frac{\Phi' \rhd_U \Gamma; x : M \vdash^{(m,e,r)} Q\langle y\rangle : N \quad \Gamma(y) = \mathbf{0}}{\Gamma; x : M \vdash^{(m,e,r)} Q\langle y\rangle@[y{\leftarrow}v] : N}\ \mathrm{ES}_{\mathrm{gc}}$$

However, since $x \notin \mathcal{A} = \mathcal{B}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ would imply that $\mathsf{tight}_\Gamma(\mathcal{B})$, and so we would be able to apply Lemma 13.6.8.2 (Plugged variables and domain of type contexts) on Φ' to obtain that $y \in \mathsf{dom}(\Gamma; x : M)$. Since that is absurd, then so is this case.

□

(Click here to go back to main chapter.)

The following is required to apply Lemma 9.1.18 (Linear Removal for Useful Open CbNeed) in the proof of Proposition 9.1.20.2 (Quantitative Subject Expansion for Useful Open CbNeed-exponential case) to obtain the right indices.

Lemma 13.6.20 (Merging multi types of Useful Open CbNeed type derivations)**.**
Let $v \in \mathsf{Val}$. For any two type derivations

$$\Phi_N \rhd_U \Gamma_N \vdash^{(m_N,e_N,r_N)} v : N$$
$$\Phi_O \rhd_U \Gamma_O \vdash^{(m_O,e_O,r_O)} v : O$$

there exists type derivation

$$\Phi_N \rhd_U \Gamma_N \biguplus \Gamma_O \vdash^{(m_N+m_O,e_N+e_O,r_N+r_O)} v : N \uplus O$$

Proof.

Trivial, given that the many typing rule is the only one that can derive a multi type on the right—*i.e.*, the derived type of the subject, in this case M—for values.

□

Lemma 13.6.21 (Quantitative Subject Expansion for $\to_{\mathsf{um}}$ in term contexts)**.**
Let $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} (\mathcal{H}\langle u\rangle, [x \leftarrow s]) : M$ such that $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}))$. Then there exists $\Phi' \rhd_U \Gamma \vdash^{(m+1,e,r)} \mathcal{H}\langle(\lambda x.u)s\rangle : M$.

Proof. (Click here to go back to main chapter.)

By structural induction on $\mathcal{H}$:

- *Empty context*; *i.e.*, $\mathcal{H} = \langle\cdot\rangle$. We proceed by case analysis on the last typing rule in Φ:
  - Let Φ be of the form

$$\dfrac{\dfrac{\Psi \rhd_U \Pi; x : N \vdash^{(m_1,e_1,r_1)} u : M}{\Pi; x : N \vdash^{(m_1,e_1,r_1)} (u, \epsilon) : M}\,\mathsf{Lift} \qquad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} s : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} (u, [x \leftarrow s]) : M}\,\mathrm{ES}$$

  with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. We can then derive Φ' as follows

$$\dfrac{\dfrac{\dfrac{\Psi \rhd_U \Pi; x : N \vdash^{(m_1,e_1,r_1)} u : M}{\Pi \vdash^{(m_1,e_1,r_1)} \lambda x.u : N \multimap M}\,\mathsf{fun}}{\Pi \vdash^{(m_1,e_1,r_1)} \lambda x.u : [N \multimap M]}\,\mathsf{many} \qquad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} s : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2+1,e_1+e_2,r_1+r_2)} (\lambda x.u)s : M}\,\mathsf{app}$$

  verifying that $(m_1 + m_2 + 1, e_1 + e_2, r_1 + r_2) = (m, e + 1, r)$.

– Let Φ be of the form

$$\dfrac{\dfrac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} u : M}{\Gamma \vdash^{(m,e,r)} (u,\epsilon) : M}\ \mathsf{Lift} \qquad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} (u,[x{\leftarrow}s]) : M}\ \mathrm{ES}_{\mathrm{gc}}$$

We can then derive Φ' as follows

$$\dfrac{\dfrac{\dfrac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} u : M}{\Gamma \vdash^{(m,e,r)} \lambda x.u : \mathbf{0} \multimap M}\ \mathsf{fun}}{\Gamma \vdash^{(m,e,r)} \lambda x.u : [\mathbf{0} \multimap M]}\ \mathsf{many}}{\Gamma \vdash^{(m+1,e,r)} (\lambda x.u)s : M}\ \mathsf{app}_{\mathrm{gc}}$$

- *Application left*; *i.e.*, $\mathcal{H} = \mathcal{J}t$. We proceed by case analysis on the last typing rule in Φ:
  – Let Φ be of the form

$$\dfrac{\dfrac{\Psi \rhd_U \Pi; x : N \vdash^{(m_1,e_1,r_1)} \mathcal{J}\langle u\rangle t : M}{\Pi; x : N \vdash^{(m_1,e_1,r_1)} (\mathcal{J}\langle u\rangle t,\epsilon) : M}\ \mathsf{Lift} \qquad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} s : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} (\mathcal{J}\langle u\rangle t,[x{\leftarrow}s]) : M}\ \mathrm{ES}$$

with $\Gamma = \Pi \uplus \Delta$ and $(m,e,r) = (m_1+m_2, e_1+e_2, r_1+r_2)$. We proceed by case analysis on the last typing rule in Ψ:

  * Let Ψ be of the form

$$\dfrac{\Xi \rhd_U \Sigma; x : N \vdash^{(m',e',r')} \mathcal{J}\langle u\rangle : [P \multimap M] \quad \Omega \rhd_U T \vdash^{(m'',e'',r'')} t : P}{\Pi; x : N \vdash^{(m'+m'',e'+e'',r'+r'')} \mathcal{J}\langle u\rangle t : M}\ \mathsf{app}$$

with $\Pi = \Sigma \uplus T$ and $(m'+m'', e'+e'', r'+r'') = (m_1, e_1, r_1)$. Since we may assume that $x \notin \mathsf{fv}(t)$ by α-conversion, then we may assume as well that $x \notin \mathsf{dom}(T)$—by Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system).
We can now derive an auxiliary type derivation Z as follows

$$\dfrac{\dfrac{\Xi \rhd_U \Sigma; x : N \vdash^{(m',e',r')} \mathcal{J}\langle u\rangle : [P \multimap M]}{\Sigma; x : N \vdash^{(m',e',r')} (\mathcal{J}\langle u\rangle,\epsilon) : [P \multimap M]}\ \mathsf{Lift} \qquad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} s : N}{\Sigma \uplus \Delta \vdash^{(m'+m_2,e'+e_2,r'+r_2)} (\mathcal{J}\langle u\rangle,[x{\leftarrow}s]) : [P \multimap M]}\ \mathrm{ES}$$

on which we can apply the *i.h.* to obtain

$$Z' \rhd_U \Sigma \uplus \Delta \vdash^{(m'+m_2+1,e'+e_2,r'+r_2)} \mathcal{J}\langle(\lambda x.u)s\rangle : M$$

We can finally derive Φ' as follows

$$\dfrac{Z' \rhd_U \Sigma \uplus \Delta \vdash^{(m'+m_2+1,e'+e_2,r'+r_2)} \mathcal{J}\langle(\lambda x.u)s\rangle : [P \multimap M] \quad \Omega \rhd_U T \vdash^{(m'',e'',r'')} t : P}{\Sigma \uplus \Delta \uplus T \vdash^{(m'+m_2+1+m'',e'+e_2+e'',r'+r_2+r'')} \mathcal{J}\langle(\lambda x.u)s\rangle : [P \multimap M]}\ \mathsf{app}$$

verifying that
  · $\Sigma \uplus \Delta \uplus T = \Pi \uplus \Delta = \Gamma$, and
  · $(m'+m_2+1+m'', e'+e_2+e'', r'+r_2+r'') = (m_1+m_2+1, e_1+e_2, r_1+r_2) = (m+1, e, r)$.

∗ Let Φ be of the form

$$\dfrac{\dfrac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} \mathcal{J}\langle u\rangle t : M}{\Gamma \vdash^{(m,e,r)} (\mathcal{J}\langle u\rangle t, \epsilon) : M}\ \mathsf{Lift} \qquad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} (\mathcal{J}\langle u\rangle t, [x{\leftarrow}s]) : M}\ \mathrm{ES}_{\mathrm{gc}}$$

We proceed by case analysis on the last typing rule in Φ. The cases where app_i and $\mathsf{app}_{\mathrm{gc}}$ are proven similarly, and are left to the reader: Let Ψ be derived as

$$\dfrac{\Theta \rhd_U \Gamma_1 \vdash^{(m_1,e_1,r_1)} \mathcal{J}\langle u\rangle : [P \multimap M] \quad \Xi \rhd_U \Gamma_2 \vdash^{(m_2,e_2,r_2)} t : P}{\Gamma_1 \biguplus \Gamma_2 \vdash^{(m_1+m_2+1,e_1+e_2,r_1+r_2)} \mathcal{J}\langle u\rangle t : M}\ \mathsf{app}$$

with $\Gamma = \Gamma_1 \biguplus \Gamma_2$ and $(m, e, r) = (m_1 + m_2 + 1, e_1 + e_2, r_1 + r_2)$.
We can now derive an auxiliary type derivation Z as follows

$$\dfrac{\dfrac{\Theta \rhd_U \Gamma_1 \vdash^{(m_1,e_1,r_1)} \mathcal{J}\langle u\rangle : [P \multimap M]}{\Gamma_1 \vdash^{(m_1,e_1,r_1)} (\mathcal{J}\langle u\rangle, \epsilon) : [P \multimap M]}\ \mathsf{Lift} \qquad \Gamma_1(x) = \mathbf{0}}{\Gamma_1 \vdash^{(m_1,e_1,r_1)} (\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : [P \multimap M]}\ \mathrm{ES}_{\mathrm{gc}}$$

on which we can apply the *i.h.* to obtain $Z' \rhd_U \Gamma_1 \vdash^{(m_1+1,e_1,r_1)} \mathcal{J}\langle(\lambda x.u)s\rangle : [P \multimap M]$, and finally derive Φ' as follows

$$\dfrac{Z' \rhd_U \Gamma_1 \vdash^{(m_1+1,e_1,r_1)} \mathcal{J}\langle(\lambda x.u)s\rangle : [P \multimap M] \quad \Omega \rhd_U \Gamma_2 \vdash^{(m_2,e_2,r_2)} t : P}{\Gamma_1 \biguplus \Gamma_2 \vdash^{(m_1+1+m_2+1,e_1+e_2,r_1+r_2)} \mathcal{J}\langle(\lambda x.u)s\rangle t : M}\ \mathsf{app}$$

verifying that $(m_1 + 1 + m_2 + 1, e_1 + e_2, r_1 + r_2) = (m + 1, e, r)$.

- *Application right*; *i.e.*, $\mathcal{H} = i\mathcal{J}$. We may assume that $x \notin \mathsf{fv}(i)$ by α-conversion. We proceed by case analysis on the last typing rule in Φ:
  – Let Φ be of the form

$$\dfrac{\dfrac{\Psi \rhd_U \Pi; x : N \vdash^{(m_1,e_1,r_1)} i\mathcal{J}\langle u\rangle : M}{\Pi; x : N \vdash^{(m_1,e_1,r_1)} (i\mathcal{J}\langle u\rangle, \epsilon) : M}\ \mathsf{Lift} \qquad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} s : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} (i\mathcal{J}\langle u\rangle, [x{\leftarrow}s]) : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. By proceeding by case analysis on the last typing rule in Φ, it is easy to verify that the sub-type derivation corresponding to t can be applied with Lemma 9.1.3 (Typing properties of useful inert terms) to infer that $M \in \mathsf{Inert}$. Thus, let $M = [\mathsf{inert}]_{i\in I}$, and note that Ψ must be of the form

$$\dfrac{\Xi \rhd_U \Pi_1 \vdash^{(m'_1,e'_1,r'_1)} i : [\mathsf{inert}]_{i\in I} \quad \Omega \rhd_U \Pi_2; x : N \vdash^{(m''_1,e''_1,r''_1)} \mathcal{J}\langle u\rangle : [\mathsf{tight}]}{(\Pi_1 \biguplus \Pi_2); x : N \vdash^{(m'_1+m''_1,e'_1+e''_1,r'_1+r''_1+1)} i\mathcal{J}\langle u\rangle : [\mathsf{inert}]_{i\in I}}\ \mathsf{app}_i$$

where $\Pi = \Pi_1 \biguplus \Pi_2$ and $(m_1, e_1, r_1) = (m'_1 + m''_1, e'_1 + e''_1, r'_1 + r''_1 + 1)$. Note that $x \notin \mathsf{dom}(\Pi_1)$ because $x \notin \mathsf{fv}(t)$—and by an application of Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system).

We can now derive an auxiliary type derivation Z as follows

$$\dfrac{\dfrac{\Omega \rhd_U \Pi_2; x : N \vdash^{(m_1'',e_1'',r_1'')} \mathcal{J}\langle u\rangle : [\mathsf{tight}]}{\Pi_2; x : N \vdash^{(m_1'',e_1'',r_1'')} (\mathcal{J}\langle u\rangle, \epsilon) : [\mathsf{tight}]}\ \mathsf{Lift} \qquad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} s : N}{(\Pi_2 \uplus \Delta); x : N \vdash^{(m_1''+m_2,e_1''+e_2,r_1''+r_2)} (\mathcal{J}\langle u\rangle, [x\leftarrow s]) : [\mathsf{tight}]}\ \mathrm{ES}$$

on which we can apply the *i.h.* to obtain a type derivation

$$Z' \rhd_U (\Pi_2 \biguplus \Delta); x : N \vdash^{(m_1''+m_2+1,e_1''+e_2,r_1''+r_2)} \mathcal{J}\langle(\lambda x.u)s\rangle : [\mathsf{tight}]$$

and finally derive Φ' as follows

$$\dfrac{\Xi \rhd_U \Pi_1 \vdash^{(m_1',e_1',r_1')} i : [\mathsf{inert}]_{i\in I} \qquad Z'}{(\Pi_1 \uplus \Pi_2 \uplus \Delta); x : N \vdash^{(m_1'+m_1''+m_2+1,e_1'+e_1''+e_2,r_1'+r_1''+r_2+1)} i\mathcal{J}\langle u\rangle : [\mathsf{inert}]_{i\in I}}\ \mathsf{app}_i$$

verifying that
- $\Pi_1 \uplus \Pi_2 \uplus \Delta = \Pi \uplus \Delta = \Gamma$, and
- $(m_1' + m_1'' + m_2 + 1, e_1' + e_1'' + e_2, r_1' + r_1'' + r_2 + 1) = (m_1 + m_2 + 1, e_1 + e_2, r_1 + r_2) = (m + 1, e, r)$.

– The case where Φ ends in the application of an $\mathrm{ES}_{\mathrm{gc}}$ rule is proven very similarly to the previous case, where ES is replaced with $\mathrm{ES}_{\mathrm{gc}}$ both in the analysis of the shape of Φ and of Z, and changes are made accordingly.

□

(Click here to go back to main chapter.)

Proposition 13.6.22 (Quantitative Subject Expansion for Useful Open CbNeed)**.**
Let $\Phi' \rhd_U \Gamma \vdash^{(m,e,r)} p' : M$ be a tight type derivation.
1. Multiplicative*: if $p \to_{\mathsf{um}} p'$, then there exists a type derivation $\Phi \rhd_U \Gamma \vdash^{(m+1,e,r)} p : M$.*
2. Exponential*: if $p \to_{\mathsf{ue}} p'$, then there exists a type derivation $\Phi \rhd_U \Gamma \vdash^{(m,e+1,r)} p : M$.*

Proof. *(Click here to go back to main chapter.)*
1. We prove this by means of a weaker statement:

$$\text{Let } p' = P\langle(\lambda x.t)u\rangle \to_{\mathsf{um}} P\langle t, [x\leftarrow u]\rangle = p, \text{ with } P \in \mathcal{E}_{\mathcal{U},\mathcal{A}}, \text{ and let}$$

$$\Phi \rhd_U \Gamma \vdash^{(m,e,r)} p : M$$

be a type derivation such that $\mathsf{tight}_\Gamma(\mathcal{A})$. Then there exists

$$\Phi' \rhd_U \Gamma \vdash^{(m+1,e,r)} p' : M$$

We proceed by induction on the derivation of P:
- Let $P = (\mathcal{H}, \epsilon)$. Note that then $\Phi \rhd_U \Gamma \vdash^{(m,e,r)} (\mathcal{H}\langle t\rangle, [x\leftarrow u]) : M$. Note that $\mathsf{tight}_\Gamma(\mathsf{a}(\mathcal{H}))$, and so the statement follows by application of Lemma 9.1.19 (Quantitative Subject Expansion for $\to_{\mathsf{um}}$ in term contexts) on Φ.

- Let P be derived as
$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}z] \in \mathcal{E}_{\mathsf{upd}(\mathcal{V},y,z),\mathsf{upd}(\mathcal{B},y,z)}}\ \mathsf{M_{VAR}}$$
where $P = Q@[y{\leftarrow}z]$, $\mathcal{U} = \mathsf{upd}(\mathcal{V}, y, z)$ and $\mathcal{A} = \mathsf{upd}(\mathcal{B}, y, z)$. We proceed by case analysis on the last typing rule in Φ:
  - Let Φ be of the form
$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle t, [x{\leftarrow}u]\rangle : M \quad \Theta \rhd_U z : N \vdash^{(m_2,e_2,r_2)} z : N \quad N \neq \mathbf{0}}{\Pi \biguplus \{z : N\} \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle t, [x{\leftarrow}u]\rangle@[y{\leftarrow}z] : M}\ \mathrm{ES}$$
where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that if $y \in \mathcal{B}$, then $\mathcal{A} = (\mathcal{B} \setminus \{y\}) \cup \{z\}$ and so $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $N \in \mathsf{Tight}$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi' \rhd_U \Pi; y : N \vdash^{(m_1+1,e_1,r_1)} Q\langle(\lambda x.t)u, \epsilon\rangle : M$. The statement follows by deriving Φ' as an application of the ES rule with Ψ' and Θ as premises.
  - Let Φ be of the form
$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle t, [x{\leftarrow}u]\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle t, [x{\leftarrow}u]\rangle@[y{\leftarrow}z] : M}\ \mathrm{ES_{gc}}$$
Note that since $y \notin \mathsf{dom}(\Gamma)$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathcal{B})$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi' \rhd_U \Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u, \epsilon\rangle : M$. The statement follows by deriving Φ' as follows
$$\frac{\Psi' \rhd_U \Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle@[y{\leftarrow}z] : M}\ \mathrm{ES_{gc}}$$

- Let P be derived as
$$\frac{Q \in \mathcal{E}_{\mathcal{U},\mathcal{A}} \quad y \in (\mathcal{U} \cup \mathcal{A})}{Q@[y{\leftarrow}i^+] \in \mathcal{E}_{(\mathcal{U}\setminus\{x\})\cup\mathsf{u}(i^+),(\mathcal{A}\setminus\{x\})\cup\mathsf{a}(i^+)}}\ \mathsf{M_I}$$
where $P = Q@[y{\leftarrow}i^+]$, $\mathcal{U} = (\mathcal{U} \setminus \{x\}) \cup \mathsf{u}(i^+)$, and $\mathcal{A} = (\mathcal{A} \setminus \{x\}) \cup \mathsf{a}(i^+)$. We proceed by case analysis on the last typing rule in Φ:
  - Let Φ be of the form
$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle t, [x{\leftarrow}u]\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} i^+ : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle t, [x{\leftarrow}u]\rangle@[y{\leftarrow}i^+] : M}\ \mathrm{ES}$$
where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Pi(\mathsf{a}(i^+))$, and so we can apply the Lemma 9.1.3 (Typing properties of useful inert terms) on Θ to obtain that $N \in \mathsf{Inert}$. Thus, $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi;y:N}(\mathcal{B})$ and so we can apply the *i.h.* on Ψ to obtain type derivation $\Psi' \rhd_U \Pi; y : N \vdash^{(m_1+1,e_1,r_1)} Q\langle(\lambda x.t)u\rangle : M$. The statement follows by taking Ψ' and Θ as premises for the ES rule, thus deriving Φ'.
  - Let Φ be of the form
$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle t, [x{\leftarrow}u]\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle t, [x{\leftarrow}u]\rangle@[y{\leftarrow}i^+] : M}\ \mathrm{ES_{gc}}$$

Note that since $y \notin \mathsf{dom}(\Gamma)$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathcal{B})$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi' \rhd_U \Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle : M$. The statement follows by deriving Φ' as follows

$$\frac{\Psi' \rhd_U \Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle @ [y \leftarrow i^+] : M}\ \mathrm{ES_{gc}}$$

- Let P be derived as
$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q@[y \leftarrow s] \in \mathcal{E}_{\mathcal{V},\mathcal{B}}}\ \mathsf{M_{GC}}$$
where $P = Q@[y \leftarrow s], \mathcal{U} = \mathcal{V}$ and $\mathcal{A} = \mathcal{B}$. We proceed by case analysis on the last typing rule in Φ:
  - Let Φ be of the form
$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle t, [x \leftarrow u]\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} s : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle t, [x \leftarrow u]\rangle @ [y \leftarrow s] : M}\ \mathrm{ES}$$
where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that since $y \notin \mathcal{B}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi;y:N}(\mathcal{B})$. Hence, we can apply the *i.h.* on Ψ to obtain type derivation $\Psi \rhd_U \Pi; y : N \vdash^{(m_1+1,e_1,r_1)} Q\langle(\lambda x.t)u\rangle : M$. The statement follows by taking Ψ' and Θ as premises for the ES rule, thus deriving Φ'.
  - Let Φ be of the form
$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle t, [x \leftarrow u]\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle t, [x \leftarrow u]\rangle @ [y \leftarrow s] : M}\ \mathrm{ES_{gc}}$$
Note that $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathcal{B})$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi \rhd_U \Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle : M$. The statement follows by deriving Φ' as follows:
$$\frac{\Psi \rhd_U \Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle @ [y \leftarrow s] : M}\ \mathrm{ES_{gc}}$$

- Let P be derived as
$$\frac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \setminus \mathcal{B})}{Q@[y \leftarrow v] \in \mathcal{E}_{\mathcal{V}\setminus\{x\},\mathcal{B}}}\ \mathsf{M_U}$$
where $P = Q@[y \leftarrow v], \mathcal{U} = \mathcal{V} \setminus \{x\}$ and $\mathcal{A} = \mathcal{B}$. We proceed by case analysis on the last typing rule in Φ:
  - Let Φ be of the form
$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle t, [x \leftarrow u]\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} v : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle t, [x \leftarrow u]\rangle @ [y \leftarrow v] : M}\ \mathrm{ES}$$
where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that since $y \notin \mathcal{B}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi;y:N}(\mathcal{B})$. Hence, we can apply the *i.h.* on Ψ to obtain type derivation $\Psi \rhd_U \Pi; y : N \vdash^{(m_1+1,e_1,r_1)} Q\langle(\lambda x.t)u\rangle : M$. The statement follows by taking Ψ' and Θ as premises for the ES rule, thus deriving Φ'.

– Let Φ be of the form

$$\dfrac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle t, [x{\leftarrow}u]\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle t, [x{\leftarrow}u]\rangle @ [y{\leftarrow}s] : M}\ \mathrm{ES_{gc}}$$

Note that $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathcal{B})$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi \rhd_U \Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle : M$. The statement follows by deriving Φ' as follows:

$$\dfrac{\Psi \rhd_U \Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m+1,e,r)} Q\langle(\lambda x.t)u\rangle @ [y{\leftarrow}s] : M}\ \mathrm{ES_{gc}}$$

- Let P be derived as

$$\dfrac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}] \in \mathcal{E}_{\mathcal{V}\cup \mathsf{u}(\mathcal{H}),\mathcal{B}\cup \mathsf{a}(\mathcal{H})}}\ \mathsf{M_{HER}}$$

where $P = Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}]$, $\mathcal{U} = \mathcal{V} \cup \mathsf{u}(\mathcal{H})$ and $\mathcal{A} = \mathcal{B} \cup \mathsf{a}(\mathcal{H})$. Note that then $p' = Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}\langle(\lambda x.t)u\rangle] \to_{\mathsf{um}} (Q\langle y\rangle @ [x{\leftarrow}\mathcal{H}\langle t\rangle]) @ [x{\leftarrow}u] = p$. Moreover, since x is bound in $\lambda x.t$, then note that $x \notin \mathsf{fv}(Q\langle y\rangle)$.
We proceed by case analysis on the shape of Φ:
– Let Φ be of the form

$$\dfrac{\dfrac{\begin{array}{cc}\vdots\,\Psi & \vdots\,\Theta\\ \Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : M & \Delta; x : O \vdash^{(m_2,e_2,r_2)} \mathcal{H}\langle t\rangle : N\end{array}}{(\Pi \uplus \Delta); x : O \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}\langle t\rangle] : M}\ \mathrm{ES} \quad \begin{array}{c}\vdots\,\Xi\\ \Sigma \vdash^{(m_3,e_3,r_3)} u : O\end{array}}{\Pi \uplus \Delta \uplus \Sigma \vdash^{(m_1+m_2+m_3,e_1+e_2+e_3,r_1+r_2+r_3)} (Q\langle y\rangle @ [x{\leftarrow}\mathcal{H}\langle t\rangle]) @ [x{\leftarrow}u] : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \uplus \Delta \uplus \Sigma$ and $(m, e, r) = (m_1 + m_2 + m_3, e_1 + e_2 + e_3, r_1 + r_2 + r_3)$. Note that since $x \notin \mathsf{fv}(Q\langle y\rangle)$, then $x \notin \mathsf{dom}(\Pi)$—by Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system).
Let us now derive an auxiliary type derivation Z as follows

$$\dfrac{\dfrac{\Theta \rhd_U \Delta; x : O \vdash^{(m_2,e_2,r_2)} \mathcal{H}\langle t\rangle : N}{\Delta; x : O \vdash^{(m_2,e_2,r_2)} (\mathcal{H}\langle t\rangle, \epsilon) : N}\ \mathsf{Lift} \quad \Xi \rhd_U \Sigma \vdash^{(m_3,e_3,r_3)} u : O}{\Delta \uplus \Sigma \vdash^{(m_2+m_3,e_2+e_3,r_2+r_3)} (\mathcal{H}\langle t\rangle, [x{\leftarrow}u]) : N}\ \mathrm{ES}$$

We can apply Lemma 9.1.19 (Quantitative Subject Expansion for $\to_{\mathsf{um}}$ in term contexts) on Z to obtain type derivation $Z' \rhd_U \Delta \uplus \Sigma \vdash^{(m_2+m_3+1,e_2+e_3,r_2+r_3)} \mathcal{H}\langle(\lambda x.t)u\rangle : N$. Therefore, the statement follows by deriving Φ' as follows

$$\dfrac{\begin{array}{cc}\vdots\,\Psi & \vdots\,Z'\\ \Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : M & \Delta \uplus \Sigma \vdash^{(m_2+m_3+1,e_2+e_3,r_2+r_3)} \mathcal{H}\langle(\lambda x.t)u\rangle : N\end{array}}{\Pi \uplus \Delta \uplus \Sigma \vdash^{(m_1+m_2+m_3+1,e_1+e_2+e_3,r_1+r_2+r_3)} Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}\langle(\lambda x.t)u\rangle] : M}\ \mathrm{ES}$$

noting that $Q\langle y\rangle @ [y{\leftarrow}\mathcal{H}\langle(\lambda x.t)u\rangle] = P\langle(\lambda x.t)u\rangle$.

– The case where Φ is of the form

$$\dfrac{\dfrac{\vdots\ \Psi \qquad \vdots\ \Theta}{\Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : M \qquad \Delta \vdash^{(m_2,e_2,r_2)} \mathcal{H}\langle t\rangle : N}\ \text{ES}}{\dfrac{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle t\rangle] : M \qquad (\Pi \biguplus \Delta)(x) = \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} (Q\langle y\rangle@[x{\leftarrow}\mathcal{H}\langle t\rangle])@[x{\leftarrow}u] : M}\ \text{ES}_{\text{gc}}}$$

with $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$, can be proven analogously to the previous case.

– Suppose Φ were of the form

$$\dfrac{\dfrac{\Psi \triangleright_U \Gamma \vdash^{(m,e,r)} Q\langle y\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle t\rangle] : M}\ \text{ES}_{\text{gc}} \qquad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} (Q\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle t\rangle])@[x{\leftarrow}u] : M}\ \text{ES}_{\text{gc}}$$

However, we would have that $y \notin \mathsf{fv}(Q\langle y\rangle)$—by Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system) on Ψ—which contradicts the fact that $y \in \mathsf{fv}(Q\langle y\rangle)$ given by Lemma 13.6.8 (Plugged variables and domain of type contexts). Hence, this case is impossible.

– Suppose Φ were of the form

$$\dfrac{\dfrac{\vdots\ \Psi}{\Pi; x : O \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle : M \qquad \Pi(y) = \mathbf{0}}\ \text{ES}_{\text{gc}} \qquad \begin{array}{c}\vdots\ \Xi\\ \Sigma \vdash^{(m_3,e_3,r_3)} u : O \qquad O \neq \mathbf{0}\end{array}}{\Pi; x : O \vdash^{(m_1,e_1,r_1)} Q\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle t\rangle] : M \qquad \Pi \biguplus \Sigma \vdash^{(m,e,r)} (Q\langle y\rangle@[y{\leftarrow}\mathcal{H}\langle t\rangle])@[x{\leftarrow}u] : M}\ \text{ES}$$

However, since $x \in \mathsf{dom}(\Pi; x : O)$, then we would have that $x \in \mathsf{fv}(Q\langle y\rangle)$—by Lemma 9.1.1 (Relevance of the Useful Open CbNeed type system) on Ψ—which is absurd.

2. We prove this by means of a weaker statement:

Let $p' = P\langle x\rangle \to_{\mathsf{ue}} P\langle v\rangle = p$, with $P \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}}$ and $P(x) = v \in \mathsf{Val}$, and let $\Phi \triangleright_U \Gamma \vdash^{(m,e,r)} p : M$ be a type derivation such that $\mathsf{tight}_\Gamma(\mathcal{A})$. Then there exists $\Phi' \triangleright_U \Gamma \vdash^{(m,e+1,r)} p' : M$

We proceed by induction on the derivation of P:

- The case where $P = (\mathcal{H}^{@}, \epsilon)$ is impossible, since it would be that $\mathsf{dom}(P) = \emptyset$.
- Suppose P is derived as

$$\dfrac{Q \in \mathcal{E}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}] \in \mathcal{E}^{@}_{\mathcal{V}\cup\mathsf{u}(\mathcal{H}^{@}),\mathcal{B}\cup\mathsf{a}(\mathcal{H}^{@})}}\ \mathsf{E}_{\mathsf{AX}_2}$$

where $P = Q\langle y\rangle@[y{\leftarrow}\mathcal{H}^{@}]$, $\mathcal{U} = \mathcal{V} \cup \mathsf{u}(\mathcal{H}^{@})$ and $\mathcal{A} = \mathcal{B} \cup \mathsf{a}(\mathcal{H}^{@})$. But then $x \notin \mathsf{dom}(P)$, which is absurd.

- Let P be derived as

$$\dfrac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}z] \in \mathcal{E}^{@}_{\mathsf{upd}(\mathcal{V},y,z),\mathsf{upd}(\mathcal{B},y,z)}}\ \mathsf{E}_{\mathsf{VAR}}$$

where $P = Q@[y{\leftarrow}z]$, $\mathcal{U} = \mathsf{upd}(\mathcal{V}, y, z)$ and $\mathcal{A} = \mathsf{upd}(\mathcal{B}, y, z)$. Note that $x \in \mathsf{dom}(Q)$, since $z \notin \mathsf{Val}$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : M \quad \Theta \rhd_U z : N \vdash^{(m_2,e_2,r_2)} z : N \quad N \neq \mathbf{0}}{\Pi \uplus \{z : N\} \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle v\rangle@[y{\leftarrow}z] : M}\ \text{ES}$$

where $\Gamma = \Pi \uplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that if $y \in \mathcal{B}$, then $\mathcal{A} = (\mathcal{B} \setminus \{y\}) \cup \{z\}$ and so $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $N \in \mathsf{Tight}$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi' \rhd_U \Pi; y : N \vdash^{(m_1,e_1+1,r_1)} Q\langle x\rangle : M$. The statement follows by deriving Φ' as an application of the ES rule having Ψ' and Θ as premises.

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle v\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle v\rangle@[y{\leftarrow}z] : M}\ \text{ES}_{\text{gc}}$$

Note that since $y \notin \mathsf{dom}(\Gamma)$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathcal{B})$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi' \rhd_U \Gamma \vdash^{(m,e+1,r)} Q\langle x, \epsilon\rangle : M$. The statement follows by deriving Φ' as follows

$$\frac{\Psi' \rhd_U \Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle@[y{\leftarrow}z] : M}\ \text{ES}_{\text{gc}}$$

- Let P be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{U},\mathcal{A}} \quad y \in (\mathcal{U} \cup \mathcal{A})}{Q@[y{\leftarrow}i^+] \in \mathcal{E}^{@}_{(\mathcal{U}\setminus\{x\})\cup \mathsf{u}(i^+),(\mathcal{A}\setminus\{x\})\cup\mathsf{a}(i^+)}}\ \mathsf{E_I}$$

where $P = Q@[y{\leftarrow}i^+]$, $\mathcal{U} = (\mathcal{U} \setminus \{x\}) \cup \mathsf{u}(i^+)$, and $\mathcal{A} = (\mathcal{A} \setminus \{x\}) \cup \mathsf{a}(i^+)$. Note that since $i^+ \notin \mathsf{Val}$, then $x \neq y$ and $x \in \mathsf{dom}(Q)$. We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} i^+ : N \quad N \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle v\rangle@[y{\leftarrow}i^+] : M}\ \text{ES}$$

where $\Gamma = \Pi \uplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Pi(\mathsf{a}(i^+))$, and so we can apply the Lemma 9.1.3 (Typing properties of useful inert terms) on Θ to obtain that $N \in \mathsf{Inert}$. Thus, $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi;y:N}(\mathcal{B})$ and so we can apply the *i.h.* on Ψ to obtain type derivation $\Psi' \rhd_U \Pi; y : N \vdash^{(m_1,e_1+1,r_1)} Q\langle x\rangle : M$. The statement follows by taking Ψ' and Θ as premises for the ES rule, thus deriving Φ'.

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle v\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle v\rangle@[y{\leftarrow}i^+] : M}\ \text{ES}_{\text{gc}}$$

Note that since $y \notin \mathsf{dom}(\Gamma)$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathcal{B})$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi' \rhd_U \Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle : M$. The statement follows by deriving Φ' as follows

$$\frac{\Psi' \rhd_U \Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle@[y{\leftarrow}i^+] : M}\ \text{ES}_{\text{gc}}$$

- Let P be derived as
$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q@[y{\leftarrow}s] \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}}}\ \mathsf{E_{GC}}$$
where $P = Q@[y{\leftarrow}s]$, $\mathcal{U} = \mathcal{V}$ and $\mathcal{A} = \mathcal{B}$.
We distinguish two cases, namely the one where $x = y$ and $s = v$—*i.e.*, when $p = Q\langle v\rangle@[x{\leftarrow}v]$—and one where $x \neq y$. Let us first assume that $x \neq y$ and proceed by case analysis on the last typing rule in Φ:
  - Let Φ be of the form
$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} s : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle v\rangle@[y{\leftarrow}s] : M}\ \mathrm{ES}$$
where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that since $y \notin \mathcal{B}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi;y:N}(\mathcal{B})$. Hence, we can apply the *i.h.* on Ψ to obtain type derivation $\Psi \rhd_U \Pi; y : N \vdash^{(m_1,e_1+1,r_1)} Q\langle x\rangle : M$. The statement follows by taking Ψ' and Θ as premises for the ES rule, thus deriving Φ'.
  - Let Φ be of the form
$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle v\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle v\rangle@[y{\leftarrow}s] : M}\ \mathrm{ES_{gc}}$$
Note that $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathcal{B})$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi \rhd_U \Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle : M$. The statement follows by deriving Φ' as follows:
$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle@[y{\leftarrow}s] : M}\ \mathrm{ES_{gc}}$$

This completes the case where $x \neq y$. Let us now assume that $x = y$ and $s = v$. That is,
$$p' = Q\langle x\rangle@[x{\leftarrow}v] \to_{\mathsf{ue}} Q\langle v^\alpha\rangle@[x{\leftarrow}v] = p$$
We proceed by case analysis on the last typing rule in Φ:
  - Let Φ be of the form
$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} v : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle v\rangle@[x{\leftarrow}v] : M}\ \mathrm{ES}$$
where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that $x \notin (\mathcal{B} \cup \mathsf{dom}(Q))$ because $x = y$, and that we can safely assume that $\mathsf{fv}(v) \cap \mathsf{dom}(Q) = \emptyset$. Hence, we can apply Lemma 9.1.18.2 on Ψ to obtain multi type O and type derivations
$$\Xi \rhd_U \Pi' \vdash^{(m'_1,e'_1,r'_1)} v : O$$
and
$$\Omega \rhd_U \Pi''; x : (N \uplus O) \vdash^{(m''_1,e''_1,r''_1)} Q\langle x\rangle : M$$
such that $\Pi = \Pi' \biguplus \Pi''$ and $(m'_1{+}m''_1, e'_1{+}e''_1, r'_1{+}r''_1) = (m_1, e_1{+}1, r_1)$. Next, we apply Lemma 13.6.20 (Merging multi types of Useful Open CbNeed type derivations) on Ξ and Θ to obtain
$$Z \rhd_U \Pi' \biguplus \Delta \vdash^{(m'_1+m_2,e'_1+e_2,r'_1+r_2)} v : N \uplus O$$

Finally, the statement follows by deriving Φ' as follows:

$$\frac{\Omega \qquad Z \qquad (N \uplus O) \neq \mathbf{0}}{\Pi'' \biguplus \Pi' \biguplus \Delta \vdash^{(m_1''+m_1'+m2, e_1''+e_1'+e_2, r_1''+r_1'+r_2)} Q\langle x\rangle@[x{\leftarrow}v] : M}\ \text{ES}$$

where $\Pi'' \biguplus \Pi' \biguplus \Delta = \Pi \biguplus \Delta = \Gamma$ and

$$\begin{array}{rl} & (m_1'' + m_1' + m2, e_1'' + e_1' + e_2, r_1'' + r_1' + r_2) \\ = & (m_1 + m_2, r_1 + r2, (e_1 + 1) + e_2) \\ = & (m, r, e + 1) \end{array}$$

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle v\rangle : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle v\rangle@[x{\leftarrow}v] : M}\ \text{ES}_{\text{gc}}$$

By application of Lemma 9.1.18.2 on Ψ, there exist multi type $O \neq \mathbf{0}$ and type derivations

$$\Theta \rhd_U \Pi; x : O \vdash^{(m',e',r')} Q\langle x\rangle : M$$

and

$$\Xi \rhd_U \Delta \vdash^{(m'',e'',r'')} v : O$$

such that $\Gamma = \Pi \biguplus \Delta$ and $(m, e + 1, r) = (m' + m'', e' + e'', r' + r'')$.
Therefore, we can derive Φ' as follows:

$$\frac{\Theta \rhd_U \Pi; x : O \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Xi \rhd_U \Delta \vdash^{(m'',e'',r'')} v : O \quad O \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m'+m'', e'+e'', r'+r'')} Q\langle x\rangle@[x{\leftarrow}v] : M}\ \text{ES}_{\text{gc}}$$

- Let P be derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \in (\mathcal{V} \setminus \mathcal{B})}{Q@[y{\leftarrow}w] \in \mathcal{E}^{@}_{\mathcal{V}\setminus\{x\},\mathcal{B}}}\ \mathsf{E_U}$$

where $P = Q@[y{\leftarrow}w]$, $\mathcal{U} = \mathcal{V} \setminus \{x\}$ and $\mathcal{A} = \mathcal{B}$.
We distinguish two cases, namely the one where $x = y$ and $s = v$—*i.e.*, when $p = Q\langle v\rangle@[x{\leftarrow}v]$—and one where $x \neq y$. Let us first assume that $x \neq y$ and proceed by case analysis on the last typing rule in Φ:
– Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; y : N \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} w : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2, e_1+e_2, r_1+r_2)} Q\langle v\rangle@[y{\leftarrow}w] : M}\ \text{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. Note that since $y \notin \mathcal{B}$, then $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_{\Pi;y:N}(\mathcal{B})$. Hence, we can apply the *i.h.* on Ψ to obtain type derivation $\Psi \rhd_U \Pi; y : N \vdash^{(m_1,e_1+1,r_1)} Q\langle x\rangle : M$. The statement follows by taking Ψ' and Θ as premises for the ES rule, thus deriving Φ'.
– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle v\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle v\rangle@[y{\leftarrow}s] : M}\ \text{ES}_{\text{gc}}$$

Note that $\mathsf{tight}_\Gamma(\mathcal{A})$ implies that $\mathsf{tight}_\Gamma(\mathcal{B})$. Hence, we can apply the *i.h.* on Ψ to obtain $\Psi \rhd_U \Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle : M$. The statement follows by deriving Φ' as follows:

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle : M \quad \Gamma(y) = \mathbf{0}}{\Gamma \vdash^{(m,e+1,r)} Q\langle x\rangle @ [y \leftarrow s] : M}\ \mathrm{ES_{gc}}$$

This completes the case where $x \neq y$. Let us now assume that $x = y$ and $s = v$. That is,

$$p' = Q\langle x\rangle @ [x \leftarrow v] \rightarrow_{\mathsf{ue}} Q\langle v^\alpha\rangle @ [x \leftarrow v]$$

We proceed by case analysis on the last typing rule in Φ:

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Pi; x : N \vdash^{(m_1,e_1,r_1)} Q\langle v\rangle : M \quad \Theta \rhd_U \Delta \vdash^{(m_2,e_2,r_2)} v : N \quad N \neq \mathbf{0}}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,e_1+e_2,r_1+r_2)} Q\langle v\rangle @ [x \leftarrow v] : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \biguplus \Delta$ and $(m, e, r) = (m_1 + m_2, e_1 + e_2, r_1 + r_2)$. We may assume that $x \notin (\mathcal{B} \cup \mathsf{dom}(\Gamma))$ and that $\mathsf{fv}(v) \cap \mathsf{dom}(\Gamma) = \emptyset$—by α-conversion. We can then apply Lemma 9.1.18.2 on Ψ to obtain multi type O and type derivations

$$\Xi \rhd_U \Pi' \vdash^{(m'_1,e'_1,r'_1)} : O$$

and

$$\Omega \rhd_U \Pi''; x : (N \uplus O) \vdash^{(m''_1,e''_1,r''_1)} : M$$

such that $\Pi = \Pi' \biguplus \Pi''$ and $(m'_1 + m''_1, e'_1 + e''_1, r'_1 + r''_1) = (m_1, e_1 + 1, r_1)$. Next, we apply Lemma 13.6.20 (Merging multi types of Useful Open CbNeed type derivations) on Ξ and Θ to obtain

$$Z \rhd_U \Pi' \biguplus \Delta \vdash^{(m'_1+m_2,e'_1+e_2,r'_1+r_2)} v : N \uplus O$$

Finally, the statement is verified by the following derivation of Φ'

$$\frac{\Omega \qquad Z \qquad (N \uplus O) \neq \mathbf{0}}{\Pi'' \biguplus \Pi' \biguplus \Delta \vdash^{(m''_1+m'_1+m2,e''_1+e'_1+e_2,r''_1+r'_1+r_2)} Q\langle x\rangle @ [x \leftarrow v] : M}\ \mathrm{ES}$$

where $\Pi'' \biguplus \Pi' \biguplus \Delta = \Pi \biguplus \Delta = \Gamma$ and

$$\begin{array}{ll} & (m''_1 + m'_1 + m2, e''_1 + e'_1 + e_2, r''_1 + r'_1 + r_2) \\ = & (m_1 + m_2, r_1 + r2, (e_1 + 1) + e_2) \\ = & (m, r, e + 1) \end{array}$$

– Let Φ be of the form

$$\frac{\Psi \rhd_U \Gamma \vdash^{(m,e,r)} Q\langle v\rangle : M \quad \Gamma(x) = \mathbf{0}}{\Gamma \vdash^{(m,e,r)} Q\langle v\rangle @ [x \leftarrow v] : M}\ \mathrm{ES_{gc}}$$

By application of Lemma 9.1.18.2 on Ψ, there exist multi type $O \neq \mathbf{0}$ and type derivations

$$\Theta \rhd_U \Pi; x : O \vdash^{(m',e',r')} Q\langle x\rangle : M$$

and

$$\Xi \rhd_U \Delta \vdash^{(m'',e'',r'')} v : O$$

such that $\Gamma = \Pi \uplus \Delta$ and $(m, e+1, r) = (m'+m'', e'+e'', r'+r'')$. Therefore, we can derive Φ' as follows:

$$\frac{\Theta \rhd_U \Pi; x : O \vdash^{(m',e',r')} Q\langle x\rangle : M \quad \Xi \rhd_U \Delta \vdash^{(m'',e'',r'')} v : O \quad O \neq \mathbf{0}}{\Pi \uplus \Delta \vdash^{(m'+m'',e'+e'',r'+r'')} Q\langle x\rangle @ [x \leftarrow v] : M}\ \mathrm{ES_{gc}}$$

- Suppose P is derived as

$$\frac{Q \in \mathcal{E}^{@}_{\mathcal{V},\mathcal{B}} \quad y \notin (\mathcal{V} \cup \mathcal{B})}{Q\langle y\rangle @ [y \leftarrow \mathcal{H}] \in \mathcal{E}^{@}_{\mathcal{V} \cup \mathsf{u}(\mathcal{H}), \mathcal{B} \cup \mathsf{a}(\mathcal{H})}}\ \mathsf{E_{NL}}$$

where $P = Q\langle y\rangle @ [y \leftarrow \mathcal{H}]$, $\mathcal{U} = \mathcal{V} \cup \mathsf{u}(\mathcal{H})$ and $\mathcal{A} = \mathcal{B} \cup \mathsf{a}(\mathcal{H})$. But then $x \notin \mathsf{dom}(P)$, which is absurd.

□

(Click here to go back to main chapter.)

Theorem 13.6.23 (Tight Completeness for Useful Open CbNeed)**.**
Given $p \in \mathcal{PR}$ such that $d : p \to^*_{\mathsf{und}} q$ for some q in $\to_{\mathsf{und}}$-normal form, there exists a tight type derivation $\Phi \rhd_U \Gamma \vdash^{(|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})} p : M$ where $\mathsf{dom}(\Gamma) = \mathsf{nv}(q)$.

Proof. (Click here to go back to main chapter.)
By induction on the length $|d|$ of the reduction sequence $d : p \to^*_{\mathsf{und}} q$:

- *Base case*: Let $k := 0$. Then p is in $\to_{\mathsf{und}}$-normal form and $p = q$. The statement follows by application of Proposition 9.1.17 (Tight typability of Useful Open CbNeed-normal forms), which yields a tight type derivation $\Phi \rhd_U \Gamma \vdash^{(0,0,|p|_{\mathsf{nd}})} t : M$ such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$.
- *Inductive case*: Let $k > 0$. That is, d is of the following form

$$d : p \to_{\mathsf{und}} \underbrace{r \to^{k-1}_{\mathsf{und}} q}_{d'}$$

By *i.h.*, there exists tight type derivation

$$\Psi \rhd_U \Gamma \vdash^{(|d'|_{\mathsf{m}}, |d'|_{\mathsf{e}}, |q|_{\mathsf{nd}})} r : M$$

such that $\mathsf{dom}(\Gamma) = \mathsf{nv}(p)$. We proceed by case analyis on the kind of reduction step in $p \to_{\mathsf{und}} r$:

- *Multiplicative step*: Let $p \to_{\mathsf{um}} r$. By Proposition 9.1.20.1 (Quantitative Subject Expansion for Useful Open CbNeed - Multiplicative), there exists (tight) type derivation

$$\Psi' \rhd_U \Gamma \vdash^{(|d'|_{\mathsf{m}}+1, |d'|_{\mathsf{e}}, |q|_{\mathsf{nd}})} p : M$$

Therefore, the statement follows by taking $\Phi := \Psi'$, noting in particular that

$$(|d'|_{\mathsf{m}} + 1, |d'|_{\mathsf{e}}, |q|_{\mathsf{nd}}) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})$$

– *Exponential step*: Let $p \to_{\mathsf{ue}} r$. By Proposition 9.1.20.2 (Quantitative Subject Expansion for Useful Open CbNeed - Exponential), there exists (tight) type derivation

$$\Psi' \rhd_U \Gamma \vdash^{(|d'|_{\mathsf{m}}, |d'|_{\mathsf{e}}+1, |q|_{\mathsf{nd}})} p : M$$

Therefore, the statement follows by taking $\Phi \coloneqq \Psi'$, noting in particular that

$$(|d'|_{\mathsf{m}}, |d'|_{\mathsf{e}} + 1, |q|_{\mathsf{nd}}) = (|d|_{\mathsf{m}}, |d|_{\mathsf{e}}, |q|_{\mathsf{nd}})$$

□

(Click here to go back to main chapter.)

13.7 Proofs of Chapter 10 (Strong CbV)

The Value Substitution Calculus

Irrelevance of $\to_{\mathsf{e_{var}}}$.

Lemma 13.7.1 (Strong normalization of $\to_{\mathsf{e_{var}}}$)**.**
The reduction relation $\to_{\mathsf{e_{var}}}$ is strongly normalizing.

Proof.
Trivial, since each $\to_{\mathsf{e_{var}}}$-step strictly decreases the number of ESs.

□

Lemma 13.7.2 ($\to_{\mathsf{e_{var}}}$ preserves $\to_{\mathsf{vsc}_\lambda}$-normal forms)**.**
Let $t, u \in \Lambda_{\mathsf{L}}$ be such that $t \to_{\mathsf{e_{var}}} u$. Then, t is in $\to_{\mathsf{vsc}_\lambda}$-normal form if and only if u is in $\to_{\mathsf{vsc}_\lambda}$-normal form.

Proof.
Trivial, by induction on the definition of $t \to_{\mathsf{e_{var}}} u$.

□

Lemma 13.7.3 (Swaps: Rewriting properties of $\to_{\mathsf{e_{var}}}$)**.**
1. Swapping with respect to $\to_{\mathsf{m}}$: $\to_{\mathsf{e_{var}}}\to_{\mathsf{m}} \subseteq \to_{\mathsf{m}}\to_{\mathsf{e_{var}}}$
2. Swapping with respect to $\to_{\mathsf{e}_\lambda}$: $\to_{\mathsf{e_{var}}}\to_{\mathsf{e}_\lambda} \subseteq \to_{\mathsf{e}_\lambda}\to^*_{\mathsf{e_{var}}} \cup \to_{\mathsf{e}_\lambda}\to_{\mathsf{e}_\lambda}$
3. Swapping with respect to $\to^+_{\mathsf{e}_\lambda}$: $\to^*_{\mathsf{e_{var}}}\to^+_{\mathsf{e}_\lambda} \subseteq \to^+_{\mathsf{e}_\lambda}\to^*_{\mathsf{e_{var}}}$

Proof.
The only non trivial point is the third one, that we now prove. The hypothesis is $t \to^k_{\mathsf{e_{var}}}\to^h_{\mathsf{e}_\lambda} u$ for some $k \geq 0, h \geq 1$. We recall that $\to_{\mathsf{e}_\lambda} \cup \to_{\mathsf{e_{var}}}$ is strongly normalizing. The proof is by lexicographic induction on (n, k), where n is the length of the longest $\to_{\mathsf{e}_\lambda} \cup \to_{\mathsf{e_{var}}}$ sequence from t. Now the case $k = h = 1$ is given by the second point. Then consider $t \to^k_{\mathsf{e_{var}}}\to_{\mathsf{e_{var}}}\to_{\mathsf{e}_\lambda}\to^h_{\mathsf{e}_\lambda} u$, and apply the swap of the second point to the central pair. There are two cases:

1. $\to_{\mathsf{e_{var}}}\to_{\mathsf{e}_\lambda} \subseteq \to_{\mathsf{e}_\lambda}\to^*_{\mathsf{e_{var}}}$, and so $t \to^k_{\mathsf{e_{var}}}\to_{\mathsf{e}_\lambda}\to^*_{\mathsf{e_{var}}}\to^h_{\mathsf{e}_\lambda} u$. Now by *i.h.* (on k) applied to the prefix $\to^k_{\mathsf{e_{var}}}\to_{\mathsf{e}_\lambda}$ we obtain $t \to^+_{\mathsf{e}_\lambda}\to^*_{\mathsf{e_{var}}}\to^*_{\mathsf{e_{var}}}\to^h_{\mathsf{e}_\lambda} u$. By *i.h.* (on the longest sequence), the suffix $\to^*_{\mathsf{e_{var}}}\to^*_{\mathsf{e_{var}}}\to^h_{\mathsf{e}_\lambda}$ turns into $\to^+_{\mathsf{e}_\lambda}\to^*_{\mathsf{e_{var}}}$, giving $t \to^+_{\mathsf{e}_\lambda}\to^+_{\mathsf{e}_\lambda}\to^*_{\mathsf{e_{var}}} u$, that is, the statement holds.
2. $\to_{\mathsf{e_{var}}}\to_{\mathsf{e}_\lambda} \subseteq \to_{\mathsf{e}_\lambda}\to_{\mathsf{e}_\lambda}$, and so $t \to^k_{\mathsf{e_{var}}}\to_{\mathsf{e}_\lambda}\to_{\mathsf{e}_\lambda}\to^h_{\mathsf{e}_\lambda} u$. Now by *i.h.* (on k) applied to the whole sequence, we obtain $t \to^+_{\mathsf{e}_\lambda}\to^*_{\mathsf{e_{var}}} u$.

□

Proposition 13.7.4 (Irrelevance of $\to_{\mathsf{e_{var}}}$)**.**
Let $d : t \to^*_{\mathsf{vsc}} u$. Then there is $d' : t \to^*_{\mathsf{vsc}_\lambda}\to^*_{\mathsf{e_{var}}} u$ with $|d'|_{\mathsf{m}} = |d|_{\mathsf{m}}$.
Moreover, $\to_{\mathsf{vsc}_\lambda}$ is weakly normalizing (resp. strongly normalizing) on t if and only if $\to_{\mathsf{vsc}}$ is weakly normalizing (resp. strongly normalizing) on t.

Proof. (Click here to go back to main chapter.)

- *Postponement*: By induction on the length $|d|$ of the evaluation $d : t \to^*_{\mathsf{vsc}} u$, using Lemma 13.7.3 (Swaps: Rewriting properties of $\to_{\mathsf{e_{var}}}$).

- *Termination:* Clearly, if $\to_{\mathsf{vsc}_\lambda} \cup \to_{\mathsf{e_{var}}}$ is strongly normalizing on t then $\to_{\mathsf{vsc}_\lambda}$ is strongly normalizing on t, since $\to_{\mathsf{vsc}_\lambda} \subseteq \to_{\mathsf{vsc}_\lambda} \cup \to_{\mathsf{e_{var}}}$.
  Conversely, suppose that $\to_{\mathsf{vsc}_\lambda} \cup \to_{\mathsf{e_{var}}}$ is not strongly normalizing on t. Then, there is an infinite sequence of terms $(t_i)_{i\in\mathbb{N}}$ such that $t_i \,(\to_{\mathsf{vsc}_\lambda} \cup \to_{\mathsf{e_{var}}})\, t_{i+1}$ and $t = t_0$. Hence, for any $n \in \mathbb{N}$ there is an evaluation $d_n \colon t \,(\to_{\mathsf{vsc}_\lambda} \cup \to_{\mathsf{e_{var}}})^* \, t_n$ with $|d_n|_{\mathsf{m}} = n$ and, by postponement irrelevance, there is an evaluation $d'_n \colon t \to^*_{\mathsf{vsc}_\lambda} u_n$ with $|d'_n|_{\mathsf{m}} = n$. Thus $\to_{\mathsf{vsc}_\lambda}$ is not strongly normalizing on t (by Konig's lemma).
  If $\to_{\mathsf{vsc}_\lambda} \cup \to_{\mathsf{e_{var}}}$ is weakly normalizing on t then $t \,(\to_{\mathsf{vsc}_\lambda} \cup \to_{\mathsf{e_{var}}})^* \, u$ for some u normal for $\to_{\mathsf{vsc}_\lambda} \cup \to_{\mathsf{e_{var}}}$. By postponement irrelevance, $t \to^*_{\mathsf{vsc}_\lambda} s \to^*_{\mathsf{e_{var}}} u$ where s is $\to_{\mathsf{vsc}_\lambda}$-normal—by Lemma 13.7.2 ($\to_{\mathsf{e_{var}}}$ preserves $\to_{\mathsf{vsc}_\lambda}$-normal forms). Therefore, $\to_{\mathsf{vsc}_\lambda}$ is weakly normalizing on t.
  If $\to_{\mathsf{vsc}_\lambda}$ is weakly normalizing on t then $t \to^*_{\mathsf{vsc}_\lambda} s$ for some s that is $\to_{\mathsf{vsc}_\lambda}$-normal. Since $\to_{\mathsf{e_{var}}}$ is strongly normalizing—by Lemma 13.7.2 ($\to_{\mathsf{e_{var}}}$ preserves $\to_{\mathsf{vsc}_\lambda}$-normal forms)—$s \to^*_{\mathsf{e_{var}}} u$ for some u that is $\to_{\mathsf{e_{var}}}$-normal. By Lemma 13.7.2 ($\to_{\mathsf{e_{var}}}$ preserves $\to_{\mathsf{vsc}_\lambda}$-normal forms), u is also $\to_{\mathsf{vsc}_\lambda}$-normal. Therefore, $\to_{\mathsf{vsc}_\lambda} \cup \to_{\mathsf{e_{var}}}$ is weakly normalizing on t.

□

(Click here to go back to main chapter.)

Syntactic characterization of VSC-normal forms.

Lemma 13.7.5 (Shape of strong fireballs)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be a strong fireball. Then exactly one of the following holds:
- *Either t is a strong inert term, or*
- *t is a value fireball.*

Proof.
Proving that at least one of the two holds is left for the reader. We now prove that only one of them holds:
- Let t be a strong inert term. We prove that t is not a value fireball by structural induction on t:
  - *Variable*: Trivial.
  - *Application*: Trivial.
  - ES; *i.e.*, $t = i_s[x\leftarrow j_s]$: Then i_s is not a value fireball —by *i.h.*—, and so neither is t.
- Let $t = S\langle\lambda x.f_s\rangle$, with $S = [x_1\leftarrow i_{s,1}] \dots [x_n\leftarrow i_{s,n}]$, with $n \geq 0$. We prove that t is not a strong inert term by induction on n:
  - *Empty substitution context*; *i.e.*, $S = \langle\cdot\rangle$: Trivial.
  - *Non-empty substitution context*; *i.e.*, $S = S'[x\leftarrow i_{s,n+1}]$: Since $S'\langle\lambda x.f\rangle$ is not a strong inert term —by *i.h.*—, then neither is $S'\langle\lambda x.f\rangle[x\leftarrow i_{s,n+1}] = t$. □

Lemma 13.7.6 (Substitution of strong inert terms preserves strong fireballs)**.**
Let f_s be a strong fireball. Then for every strong inert term i_s, $f_s\{x\leftarrow i_s\}$ is a strong fireball.

Proof.
Trivial, by induction on the syntactic structure of f_s.

□

Proposition 13.7.7 (Syntactic characterization of VSC-normal forms)**.**

Let $t \in \Lambda_{\mathsf{L}}$. t is in $\to_{\mathsf{vsc}_\lambda}$-normal form (resp. $\to_{\mathsf{vsc}}$-normal form) if and only if t is a strong fireball (resp. strong super fireball).

Proof. (Click here to go back to main chapter.)

- Let t be in $\to_{\mathsf{vsc}_\lambda}$-normal form. We prove that t is a strong fireball by structural induction on t:
  - *Variable.* Trivial.
  - *Abstraction*; Let $t \coloneqq \lambda x.u$. Since t is $\to_{\mathsf{vsc}_\lambda}$-normal, so is u. By *i.h.*, u is a strong fireball, and then so is t.
  - *Application*: $t \coloneqq t_1 t_2$. Since t is $\to_{\mathsf{vsc}_\lambda}$-normal, so are t_1 and t_2. By *i.h.*, t_1 and t_2 are strong fireballs. Note that $t_1 \neq S\langle\lambda x.u\rangle$, otherwise $t \mapsto_{\mathsf{m}} S\langle u[x\leftarrow t_2]\rangle$ which contradicts $\to_{\mathsf{vsc}_\lambda}$-normality of t. Thus, t_1 is a strong inert term—by Lemma 13.7.5 (Shape of strong fireballs)—and so t is a strong fireball.
  - *Explicit substitution*: Let $t \coloneqq t_1[x\leftarrow t_2]$.Since t is $\to_{\mathsf{vsc}_\lambda}$-normal, then so are t_1 and t_2. By *i.h.*, t_1 and t_2 are strong fireballs. Note that $t_2 \neq S\langle v\rangle$, otherwise $t \mapsto_{\mathsf{e}_\lambda} S\langle t_1\{x\leftarrow v\}\rangle$ which contradicts the $\to_{\mathsf{vsc}_\lambda}$-normality of t. Thus t_2 is a strong inert term—by Lemma 13.7.5 (Shape of strong fireballs)—and so t is a strong fireball.
- Let t be a strong fireball. We prove that t is in $\to_{\mathsf{vsc}_\lambda}$-normal form by structural induction on the definition of strong fireball.
  - *Variable.* Trivial.
  - *Abstraction*: Let $t \coloneqq \lambda x.f_s$. By *i.h.*, f_s is $\to_{\mathsf{vsc}_\lambda}$-normal, and hence so is t.
  - *Application*: Let $t \coloneqq i_s f_s$. By *i.h.*, i_s and f_s are $\to_{\mathsf{vsc}_\lambda}$-normal. By Lemma 13.7.5 (Shape of strong fireballs), note that i_s is not of the form $S\langle\lambda x.u\rangle$, and so t is also in $\to_{\mathsf{vsc}_\lambda}$-normal form.
  - *Explicit substitution*: Let $t \coloneqq f_s[x\leftarrow i_s]$ (it includes the case when f_s is a strong inert term). By *i.h.*, both f_s and i_s are $\to_{\mathsf{vsc}_\lambda}$-normal. By Lemma 13.7.5 (Shape of strong fireballs), i_s is not of the form $S\langle v\rangle$, and so t is also in $\to_{\mathsf{vsc}_\lambda}$-normal form.
- Let t be in $\to_{\mathsf{vsc}}$-normal form. We prove that t is a strong super fireball by structural induction on t. The proof is analogous to the case where t is in $\to_{\mathsf{vsc}_\lambda}$-normal form, but for the following case.
  - *Explicit substitution*: Let $t \coloneqq t_1[x\leftarrow t_2]$. Since t is $\to_{\mathsf{vsc}}$-normal, so are t_1 and t_2. By *i.h.*, t_1 and t_2 are strong super fireballs. Note that $t_2 \neq S\langle v\rangle$, otherwise $t \mapsto_{\mathsf{e}_\lambda} S\langle t_1\{x\leftarrow v\}\rangle$ which contradicts the $\to_{\mathsf{vsc}}$-normality of t. Moreover, $t_2 \neq S\langle y\rangle$, otherwise $t \mapsto_{\mathsf{e}_{\mathsf{var}}} S\langle t_1\{x\leftarrow y\}\rangle$ which contradicts the $\to_{\mathsf{vsc}}$-normality of t. Thus, t_2 is a compound strong inert term and so t is a strong super fireball
- Let t be a strong super fireball. We prove that t is in $\to_{\mathsf{vsc}}$-normal form by structural induction on the definition of strong fireball. The proof is analogous to the one where we proved that strong fireballs are normals, except for the following case:
  - *Explicit substitution*: Let $t \coloneqq f_s[x\leftarrow i_s]$ (it includes the case when f_s is a strong inert term), where f_s and i_s are super and i_s is also compound (as t is a strong super fireball). By *i.h.*, f_s and i_s are $\to_{\mathsf{vsc}}$-normal. Since i_s is not of the form either $S\langle v\rangle$—by Lemma 13.7.5 (Shape of strong fireballs)—or $S\langle y\rangle$ (as it is compound), t is also $\to_{\mathsf{vsc}}$-normal.

□

(Click here to go back to main chapter.)

Strong bisimulation of $\equiv$ and $\to_{\mathsf{vsc}}$

Proposition 13.7.8 (Strong Bisimulation of $\equiv$ and $\to_{\mathsf{vsc}}$)**.**
Let $\mathsf{a} \in \{\mathsf{m}, \mathsf{e}_\lambda, \mathsf{e}_{\mathsf{var}}, \mathsf{sm}, \mathsf{se}_\lambda, \mathsf{se}_{\mathsf{var}}\}$. *If* $t \equiv u$ *and* $t \to_{\mathsf{a}} t'$, *then there exists* $u' \in \Lambda_{\mathsf{vsc}}$ *such that* $u \to_{\mathsf{a}} u'$ *and* $t' \equiv u'$.

Proof. *(Click here to go back to main chapter.)*
The proof that $\equiv$ is a strong bisimulation is an easy adaptation of the proof of [AP12].
Irrelevance of $\equiv$ follows from Lemma 13.7.9.1 (Postponement of $\equiv$s) below. This, in turn, relies on the fact that $\equiv$ is a strong bisimulation, although the proof of Lemma 13.7.9.1 does not use irrelevance.

□

(Click here to go back to main chapter.)

Lemma 13.7.9 (Properties of $\equiv$)**.**
1. Postponement*: if* $d\colon t(\equiv\to_{\mathsf{vsc}}\equiv)^* u$ *then there are* $s \equiv u$ *and* $d'\colon t \to^*_{\mathsf{vsc}} s$ *such that* $|d| = |d'|$ *and* $|d|_a = |d'|_a$ *for* $\mathsf{a} \in \{\mathsf{m}, \mathsf{e}_\lambda, \mathsf{e}_{\mathsf{var}}, \mathsf{om}, \mathsf{oe}_\lambda, \mathsf{oe}_{\mathsf{var}}, \mathsf{sm}, \mathsf{se}_\lambda, \mathsf{se}_{\mathsf{var}}\}$. .
2. Normal forms*: if* $t \equiv u$ *then* t *is* a*-normal iff* u *is* a*-normal.*
3. Confluence*:* $\equiv\to_{\mathsf{vsc}}\equiv$ *is confluent and* $\equiv\to_{\mathsf{s}}\equiv$ *is diamond.*

Proof.
All points are standard and follow immediately from the fact that $\equiv$ is a strong bisimulation—by the first part of Proposition 10.2.4 (Strong bisimulation of $\equiv$ and $\to_{\mathsf{vsc}}$)—proofs are easy, see [Acc11](pp. 86-87).

□

The Strong CbV strategy

Lemma 13.7.10 (Properties of rigid terms)**.**
1. *Given* $t \in \Lambda_{\mathsf{vsc}}$ *and a rigid context* R, $R\langle t\rangle$ *is a rigid term.*
2. *Let* r *be a rigid term and* $t \in \Lambda_{\mathsf{vsc}}$ *such that* $r \to_{\mathsf{w}} t$. *Then* t *is rigid.*
3. *Let* r *be a rigid term and* $t \in \Lambda_{\mathsf{vsc}}$ *such that* $r \to_{\mathsf{s}} t$. *Then* t *is rigid.*
4. *Let* $t \neq S\langle\lambda x.u\rangle$ *be in* $\to_{\mathsf{w}}$*-normal form. Then* t *is rigid.*
5. *Let* r *be a rigid term not in* $\to_{\mathsf{s}_\lambda}$*-normal form (resp.* $\to_{\mathsf{s}}$*-normal form). Then there exists a rigid context* R *and* $u, u' \in \Lambda_{\mathsf{vsc}}$ *such that* $r = R\langle u\rangle \to_{\mathsf{s}_\lambda} R\langle u'\rangle$ *(resp.* $R\langle u\rangle \to_{\mathsf{s}} R\langle u'\rangle$*), with* $u \to_{\mathsf{w}_\lambda} u'$ *(resp.* $u \to_{\mathsf{w}} u'$*).*

Proof.
1. By induction on the definition of R.
2. Let O be an open evaluation context such that $r = O\langle u\rangle \to_{\mathsf{w}} O\langle u'\rangle = t$, with $u \mapsto_{\mathsf{m}} u'$, $u \mapsto_{\mathsf{e}_\lambda} u'$ or $u \mapsto_{\mathsf{e}_{\mathsf{var}}} u'$. We prove this for $\to_{\mathsf{wm}}$ by structural induction on O; the proofs for $\to_{\mathsf{we}_\lambda}$ and $\to_{\mathsf{we}_{\mathsf{var}}}$ follow the same schema and are left for the reader.
   - *Empty context*; *i.e.*, $O = \langle\cdot\rangle$. This case is not possible, because it would imply that $r = u = S\langle\lambda x.s\rangle m$, which is not a rigid term.
   - *Application right*; *i.e.*, $O = mO'$. Since $r = mO'\langle u\rangle$, then m is a rigid term, and so $mO'\langle u'\rangle = t$ is rigid too.

- *Application left*; *i.e.*, $O = O'm$. Since $r = O'\langle u\rangle m$ and $O'\langle u'\rangle$ is rigid by *i.h.*, then $O'\langle u'\rangle m = t$ is rigid too.
- ES *left*; *i.e.*, $O = O'[x{\leftarrow}m]$. Since $r = O'\langle u\rangle[x{\leftarrow}m]$, then both $O'\langle u\rangle$ and m are rigid. Moreover, $O'\langle u'\rangle$ is rigid by *i.h.*, and so $O'\langle u'\rangle[x{\leftarrow}m] = t$ is rigid too.
- ES *right*; *i.e.*, $O = m[x{\leftarrow}O']$. Since $r = m[x{\leftarrow}O'\langle u\rangle]$, then both m and $O'\langle u\rangle$ are rigid. Moreover, $O'\langle u'\rangle$ is rigid by *i.h.*, and so $m[x{\leftarrow}O'\langle u'\rangle] = t$ is rigid too.

3. Let S be a strong evaluation context such that $r = S\langle u\rangle \to_{\mathsf{s}} S\langle u'\rangle = t$, with $u \to_{\mathsf{wm}} u'$, $u \to_{\mathsf{we}_\lambda} u'$ or $u \to_{\mathsf{we}_{\mathsf{var}}} u'$. We prove this for $u \to_{\mathsf{wm}} u'$ by structural induction on S; the proofs for $\to_{\mathsf{we}_\lambda}$ and $\to_{\mathsf{we}_{\mathsf{var}}}$ follow the same schema.
   - *Empty context*; *i.e.*, $S = \langle\cdot\rangle$. Then $r = u \to_{\mathsf{wm}} u' = t$ and the statement holds by Lemma 13.7.10.2.
   - *Under λ-abstraction right*; *i.e.*, $S = \lambda x.S'$. This case is not possible, because it would imply that $r = \lambda x.S'\langle u\rangle$, which is not a rigid term.
   - *Strong context,* ES *right*; *i.e.*, $S = m[x{\leftarrow}R]$. Since $r = m[x{\leftarrow}R\langle u\rangle]$, then both m and $R\langle u\rangle$ are rigid terms. Moreover, $R\langle u'\rangle$ is rigid by *i.h.*, and so $m[x{\leftarrow}R\langle u'\rangle] = s$ is rigid too.
   - *Strong context,* ES *left*; *i.e.*, $S = S'[x{\leftarrow}m]$, with m a rigid term. Since $r = S'\langle u\rangle[x{\leftarrow}m]$, then $S'\langle u\rangle$ is rigid. Moreover, $S'\langle u'\rangle$ is rigid by *i.h.*, and so $S'\langle u'\rangle[x{\leftarrow}m] = s$ is rigid too.
   - *Rigid context, application right*; *i.e.*, $S = mS'$, with m a rigid term. Then $mS'\langle u'\rangle = s$ is rigid too.
   - *Rigid context, application left*; *i.e.*, $S = Rm$. Since $r = R\langle u\rangle m$, then $R\langle u\rangle$ is rigid. Moreover, $R\langle u'\rangle$ is rigid by *i.h.*, and so $R\langle u'\rangle m$ is rigid too.
   - *Rigid context,* ES *left*; *i.e.*, $S = R[x{\leftarrow}m]$, with m a rigid term. Since $r = R\langle u\rangle[x{\leftarrow}m]$, then $R\langle u\rangle$ is rigid. Moreover, $R\langle u'\rangle$ is rigid by *i.h.*, and so $R\langle u'\rangle[x{\leftarrow}m] = s$ is rigid too.
   - *Rigid context,* ES *right*; *i.e.*, $S = m[x{\leftarrow}R]$, with m a rigid term. Since $r = m[x{\leftarrow}R\langle u\rangle]$, then $R\langle u\rangle$ is rigid. Moreover, $R\langle u'\rangle$ is rigid by *i.h.*, and so $m[x{\leftarrow}R\langle u'\rangle] = s$ is rigid too.
4. By structural induction on t:
   - *Variable*. Trivial.
   - *λ-abstraction*. Trivial.
   - *Application*; *i.e.*, $t = t_1t_2$. If t_1 is not in $\to_{\mathsf{w}}$-normal form, then neither is t. Moreover, if t_1 is a λ-abstraction in a substitution context, then t is not in $\mapsto_{\mathsf{m}}$-normal form —which is absurd. Thus, t_1 is rigid —by *i.h.*— and then so is t.
   - ES; *i.e.*, $t = t_1[x{\leftarrow}t_2]$. First of all, if t_1 is a λ-abstraction in a substitution context then so is t —which is absurd—, and if t_1 is not in $\to_{\mathsf{w}}$-normal form then neither is t. Therefore, t_1 is a rigid term —by *i.h.*. Second, if t_2 is an abstraction in a substitution context then t is not in $\mapsto_{\mathsf{e}}$-normal form —which is absurd—, and if t_2 is not in $\to_{\mathsf{w}}$-normal form then neither is t. Therefore, t_2 is a rigid term —by *i.h.*. Thus, t is a rigid term as well.
5. We shall leave the respective proof for $\to_{\mathsf{s}}$ to the reader, and proceed by structural induction on r to prove the one for $\to_{\mathsf{s}_\lambda}$:
   - *Variable*. Trivial.
   - *Application*; *i.e.*, $t = t_1t_2$, with t_1 a rigid term. Note that t_1 is not a λ-abstraction in a substitution context.
     Let t_1 be not in $\to_{\mathsf{s}_\lambda}$-normal form. Then by *i.h.* there exists a rigid context R' and terms $s, s' \in \Lambda_{\mathsf{vsc}}$ such that $t_1 = R''\langle u\rangle \to_{\mathsf{s}_\lambda} R'\langle s'\rangle$. Thus, the statement holds by taking $R \coloneqq R't_2$, $u \coloneqq s$ and $u' \coloneqq s'$.
     Let t_1 be in $\to_{\mathsf{s}_\lambda}$-normal form. Then t_2 is not in $\to_{\mathsf{s}_\lambda}$-normal form, implying the existence

of a strong context S and terms $s, s' \in \Lambda_{\mathsf{vsc}}$ such that $t_2 = S\langle s\rangle \to_{\mathsf{s}_\lambda} S\langle s'\rangle$ and $s \to_{\mathsf{w}_\lambda} s'$, and so the statement holds by taking $R := t_1 S$.

- ES; *i.e.*, $t = t_1[x \leftarrow t_2]$, with both t_1 and t_2 rigid terms. If t_1 is not in $\to_{\mathsf{s}_\lambda}$-normal form, then there exist —by *i.h.*— rigid context R' and terms $s, s' \in \Lambda_{\mathsf{vsc}}$ such that $t_1 = R'\langle s\rangle \to_{\mathsf{s}_\lambda} R'\langle s'\rangle$. Thus, the statement holds by taking $R := R'[x \leftarrow t_2]$, $u := s$ and $u' := s'$.
  If t_1 is in $\to_{\mathsf{s}_\lambda}$-normal form, then t_2 is not in $\to_{\mathsf{s}_\lambda}$-normal form and so there exist rigid context R' and terms $s, s' \in \Lambda_{\mathsf{vsc}}$ such that $t_2 = R'\langle s\rangle \to_{\mathsf{s}_\lambda} R'\langle s'\rangle$. Thus, the statement holds by taking $R := t_1[x \leftarrow R']$, $u := s$ and $u' := s'$. □

Lemma 13.7.11 (Properties of the VSC)**.**

1. $\to_{\mathsf{m}}$ *and* $\to_{\mathsf{e}}$ *are strongly normalizing (separately).*
2. $\to_{\mathsf{wm}}$ *and* $\to_{\mathsf{we}}$ *are diamond (separately).*
3. $\to_{\mathsf{wm}}$ *and* $\to_{\mathsf{we}}$ *strongly commute.*
4. $\to_{\mathsf{w}}$ *is diamond.*
5. $\to_{\mathsf{sm}}$ *and* $\to_{\mathsf{se}}$ *are diamond (separately).*
6. $\to_{\mathsf{sm}}$ *and* $\to_{\mathsf{se}}$ *strongly commute.*
7. $\to_{\mathsf{s}}$ *is diamond, and all* vsc*-normalizing reduction sequences* d *from* $t \in \Lambda_{\mathsf{vsc}}$ *(if any) have the same length* $|d|$*, the same number* $|d|_{\mathsf{e}}$ *of* e*-steps, and the same number* $|d|_{\mathsf{m}}$ *of* m*-steps.*

Proof.

1. See [AP12].
2. We prove that $\to_{\mathsf{wm}}$ is diamond, *i.e.* if $u \;{}_{\mathsf{wm}}\!\leftarrow t \to_{\mathsf{wm}} s$ with $u \neq s$ then there exists $t' \in \Lambda_{\mathsf{vsc}}$ such that $u \to_{\mathsf{wm}} t' \;{}_{\mathsf{wm}}\!\leftarrow s$. The proof is by induction on the definition of $\to_{\mathsf{wm}}$. Since there $t \to_{\mathsf{wm}} s \neq u$ and the reduction $\to_{\mathsf{wm}}$ is weak, there are only eight cases:
   - *Step at the Root for* $t \to_{\mathsf{wm}} u$ *and Application Right for* $t \to_{\mathsf{wm}} s$, *i.e.* $t := S\langle \lambda x.\tilde{t}\rangle m \mapsto_{\mathsf{m}} S\langle \tilde{t}[x \leftarrow m]\rangle =: u$ and $t \mapsto_{\mathsf{m}} S\langle \lambda x.\tilde{t}\rangle m' =: s$ with $m \to_{\mathsf{wm}} m'$: then, $u \to_{\mathsf{wm}} S\langle \tilde{t}[x \leftarrow m']\rangle \;{}_{\mathsf{wm}}\!\leftarrow s$;
   - *Step at the Root for* $t \to_{\mathsf{wm}} u$ *and Application Left for* $t \to_{\mathsf{wm}} s$, *i.e.*, for some $n > 0$,

     $$t := (\lambda x.\tilde{t})[x_1 \leftarrow t_1] \dots [x_n \leftarrow t_n] m \mapsto_{\mathsf{m}} \tilde{t}[x \leftarrow m][x_1 \leftarrow t_1] \dots [x_n \leftarrow t_n] =: u$$

     whereas $t \to_{\mathsf{wm}} (\lambda x.\tilde{t})[x_1 \leftarrow t_1] \dots [x_j \leftarrow t'_j] \dots [x_n \leftarrow t_n] m =: s$ with $t_j \to_{\mathsf{wm}} t'_j$ for some $1 \leq j \leq n$: then,

     $$u \to_{\mathsf{wm}} \tilde{t}[x \leftarrow m][x_1 \leftarrow t_1] \dots [x_j \leftarrow t'_j] \dots [x_n \leftarrow t_n] \;{}_{\mathsf{wm}}\!\leftarrow s;$$

   - *Application Left for* $t \to_{\mathsf{wm}} u$ *and Application Right for* $t \to_{\mathsf{wm}} s$, *i.e.* $t := m\tilde{t} \to_{\mathsf{wm}} m'\tilde{t} =: u$ and $t \to_{\mathsf{wm}} m\tilde{t'} =: s$ with $m \to_{\mathsf{wm}} m'$ and $\tilde{t} \to_{\mathsf{wm}} \tilde{t'}$: then, $u \to_{\mathsf{wm}} m'\tilde{t'} \;{}_{\mathsf{wm}}\!\leftarrow s$;
   - *Application Left for both* $t \to_{\mathsf{wm}} u$ *and* $t \to_{\mathsf{wm}} s$, *i.e.* $t := m\tilde{t} \to_{\mathsf{wm}} m'\tilde{t} =: u$ and $t \to_{\mathsf{wm}} m''\tilde{t} =: s$ with $m' \;{}_{\mathsf{wm}}\!\leftarrow m \to_{\mathsf{wm}} m''$: by *i.h.*, there exists $m_0 \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{wm}} m_0 \;{}_{\mathsf{m}}\!\leftarrow m''$, hence $u \to_{\mathsf{wm}} m_0\tilde{t} \;{}_{\mathsf{m}}\!\leftarrow s$;
   - *Application Right for both* $t \to_{\mathsf{wm}} u$ *and* $t \to_{\mathsf{wm}} s$, *i.e.* $t := \tilde{t}m \to_{\mathsf{wm}} \tilde{t}m' =: u$ and $t \to_{\mathsf{wm}} \tilde{t}m'' =: s$ with $m' \;{}_{\mathsf{wm}}\!\leftarrow m \to_{\mathsf{wm}} m''$: by *i.h.*, there exists $m_0 \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{wm}} m_0 \;{}_{\mathsf{wm}}\!\leftarrow m''$, hence $u \to_{\mathsf{wm}} \tilde{t}m_0 \;{}_{\mathsf{wm}}\!\leftarrow s$;
   - ES *left for* $t \to_{\mathsf{wm}} u$ *and* ES *right for* $t \to_{\mathsf{wm}} s$, *i.e.* $t := m[x \leftarrow \tilde{t}] \to_{\mathsf{wm}} m'[x \leftarrow \tilde{t}] =: u$ and $t \to_{\mathsf{wm}} m[x \leftarrow \tilde{t'}] =: s$ with $m \to_{\mathsf{wm}} m'$ and $\tilde{t} \to_{\mathsf{wm}} \tilde{t'}$: then,

     $$u \to_{\mathsf{wm}} m'[x \leftarrow \tilde{t'}] \;{}_{\mathsf{wm}}\!\leftarrow s$$

- ES *left for both* $t \to_{\mathsf{wm}} u$ *and* $t \to_{\mathsf{wm}} s$, *i.e.* $t := m[x\leftarrow\tilde{t}] \to_{\mathsf{wm}} m'[x\leftarrow\tilde{t}] =: u$ and $t \to_{\mathsf{wm}} m''[x\leftarrow\tilde{t}] =: s$ with $m' \,{}_{\mathsf{wm}}\!\leftarrow m \to_{\mathsf{wm}} m''$: by *i.h.*, there exists $m_0 \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{wm}} m_0 \,{}_{\mathsf{wm}}\!\leftarrow m''$, hence $u \to_{\mathsf{sm}} m_0[x\leftarrow\tilde{t}] \,{}_{\mathsf{wm}}\!\leftarrow s$;
- ES *right for both* $t \to_{\mathsf{wm}} u$ *and* $t \to_{\mathsf{wm}} s$, *i.e.* $t := \tilde{t}[x\leftarrow m] \to_{\mathsf{wm}} \tilde{t}[x\leftarrow m'] =: u$ and $t \to_{\mathsf{wm}} \tilde{t}[x\leftarrow m''] =: s$ with $m' \,{}_{\mathsf{wm}}\!\leftarrow m \to_{\mathsf{sm}} m''$: by *i.h.*, there exists $m_0 \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{wm}} m_0 \,{}_{\mathsf{wm}}\!\leftarrow m''$, hence $u \to_{\mathsf{sm}} \tilde{t}[x\leftarrow m_0] \,{}_{\mathsf{wm}}\!\leftarrow s$.

We prove that $\to_{\mathsf{we}}$ is diamond, *i.e.* if $u \,{}_{\mathsf{we}}\!\leftarrow t \to_{\mathsf{we}} s$ with $u \neq s$ then there exists $m \in \Lambda_{\mathsf{vsc}}$ such that $u \to_{\mathsf{we}} t' \,{}_{\mathsf{e}}\!\leftarrow s$. The proof is by induction on the definition of $\to_{\mathsf{we}}$. Since there $t \to_{\mathsf{we}} s \neq u$ and the reduction $\to_{\mathsf{we}}$ is weak, there are only eight cases:

- *Step at the Root for* $t \to_{\mathsf{we}} u$ and ES *left for* $t \to_{\mathsf{we}} s$, *i.e.* $t := m[x\leftarrow S\langle v\rangle] \mapsto_{\mathsf{e}} S\langle m\{x\leftarrow v\}\rangle =: u$ and $t \mapsto_{\mathsf{e}} m'[x\leftarrow S\langle v\rangle] =: s$ with $m \to_{\mathsf{we}} m'$: then,

$$u \to_{\mathsf{we}} S\langle m'[x\leftarrow v]\rangle \,{}_{\mathsf{we}}\!\leftarrow s$$

- *Step at the Root for* $t \to_{\mathsf{we}} u$ and ES *right for* $t \to_{\mathsf{we}} s$, *i.e.*, for some $n > 0$, $t := m[x\leftarrow v[x_1\leftarrow t_1]\dots[x_n\leftarrow t_n]] \mapsto_{\mathsf{e}} m\{x\leftarrow v\}[x_1\leftarrow t_1]\dots[x_n\leftarrow t_n] =: u$ whereas

$$t \to_{\mathsf{we}} m[x\leftarrow v[x_1\leftarrow t_1]\dots[x_j\leftarrow t'_j]\dots[x_n\leftarrow t_n]] =: s$$

  with $t_j \to_{\mathsf{we}} t'_j$ for some $1 \leq j \leq n$: then,

$$u \to_{\mathsf{we}} m\{x\leftarrow v\}[x_1\leftarrow t_1]\dots[x_j\leftarrow t'_j]\dots[x_n\leftarrow t_n] \,{}_{\mathsf{we}}\!\leftarrow s;$$

- *Application Left for* $t \to_{\mathsf{we}} u$ and *Application Right for* $t \to_{\mathsf{we}} s$, *i.e.* $t := m\tilde{t} \to_{\mathsf{we}} m'\tilde{t} =: u$ and $t \to_{\mathsf{we}} m\tilde{t'} =: s$ with $m \to_{\mathsf{we}} m'$ and $\tilde{t} \to_{\mathsf{we}} \tilde{t'}$: then, $u \to_{\mathsf{we}} m'\tilde{t'} \,{}_{\mathsf{we}}\!\leftarrow s$;
- *Application Left for both* $t \to_{\mathsf{we}} u$ *and* $t \to_{\mathsf{we}} s$, *i.e.* $t := m\tilde{t} \to_{\mathsf{we}} m'\tilde{t} =: u$ and $t \to_{\mathsf{we}} m''\tilde{t} =: s$ with $m' \,{}_{\mathsf{we}}\!\leftarrow m \to_{\mathsf{we}} m''$: by *i.h.*, there exists $m_0 \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{we}} m_0 \,{}_{\mathsf{we}}\!\leftarrow m''$, hence $u \to_{\mathsf{we}} m_0\tilde{t} \,{}_{\mathsf{we}}\!\leftarrow s$;
- *Application Right for both* $t \to_{\mathsf{we}} u$ *and* $t \to_{\mathsf{we}} s$, *i.e.* $t := \tilde{t}m \to_{\mathsf{we}} \tilde{t}m' =: u$ and $t \to_{\mathsf{we}} \tilde{t}m'' =: s$ with $m' \,{}_{\mathsf{we}}\!\leftarrow m \to_{\mathsf{we}} m''$: by *i.h.*, there exists $m_0 \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{we}} m_0 \,{}_{\mathsf{we}}\!\leftarrow m''$, hence $u \to_{\mathsf{we}} \tilde{t}m_0 \,{}_{\mathsf{we}}\!\leftarrow s$;
- ES *left for* $t \to_{\mathsf{we}} u$ and ES *right for* $t \to_{\mathsf{we}} s$, *i.e.* $t := m[x\leftarrow\tilde{t}] \to_{\mathsf{we}} m'[x\leftarrow\tilde{t}] =: u$ and $t \to_{\mathsf{we}} m[x\leftarrow\tilde{t'}] =: s$ with $m \to_{\mathsf{we}} m'$ and $\tilde{t} \to_{\mathsf{we}} \tilde{t'}$: then, $u \to_{\mathsf{we}} m'[x\leftarrow\tilde{t'}] \,{}_{\mathsf{we}}\!\leftarrow s$;
- ES *left for both* $t \to_{\mathsf{e}} u$ *and* $t \to_{\mathsf{e}} s$, *i.e.* $t := m[x\leftarrow\tilde{t}] \to_{\mathsf{e}} m'[x\leftarrow\tilde{t}] =: u$ and $t \to_{\mathsf{e}} m''[x\leftarrow\tilde{t}] =: s$ with $m' \,{}_{\mathsf{e}}\!\leftarrow m \to_{\mathsf{e}} m''$: by *i.h.*, there exists $m_0 \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{e}} m_0 \,{}_{\mathsf{e}}\!\leftarrow m''$, hence $u \to_{\mathsf{e}} m_0[x\leftarrow\tilde{t}] \,{}_{\mathsf{e}}\!\leftarrow s$;
- ES *right for both* $t \to_{\mathsf{e}} u$ *and* $t \to_{\mathsf{e}} s$, *i.e.* $t := \tilde{t}[x\leftarrow m] \to_{\mathsf{e}} \tilde{t}[x\leftarrow m'] =: u$ and $t \to_{\mathsf{e}} \tilde{t}[x\leftarrow m''] =: s$ with $m' \,{}_{\mathsf{e}}\!\leftarrow m \to_{\mathsf{e}} m''$: by *i.h.*, there exists $m_0 \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{e}} m_0 \,{}_{\mathsf{e}}\!\leftarrow m''$, hence $u \to_{\mathsf{e}} \tilde{t}[x\leftarrow m_0] \,{}_{\mathsf{e}}\!\leftarrow s$.

Note that in [AP12] it has just been proved the strong confluence of $\to_{\mathsf{vsc}}$, not of $\to_{\mathsf{m}}$ or $\to_{\mathsf{e}}$.

3. We show that $\to_{\mathsf{we}}$ and $\to_{\mathsf{wm}}$ strongly commute, *i.e.* if $u \,{}_{\mathsf{we}}\!\leftarrow t \to_{\mathsf{wm}} s$, then $u \neq s$ and there is $t' \in \Lambda_{\mathsf{vsc}}$ such that $u \to_{\mathsf{wm}} t' \,{}_{\mathsf{we}}\!\leftarrow s$. The proof is by induction on the definition of $t \to_{\mathsf{we}} u$. The proof that $u \neq s$ is left to the reader. Since the $\to_{\mathsf{e}}$ and $\to_{\mathsf{m}}$ cannot reduce under λ's, all vsc-values are m-normal and e-normal. So, there are the following cases.
   - *Step at the Root for* $t \to_{\mathsf{we}} u$ and ES *left for* $t \to_{\mathsf{wm}} s$, *i.e.* $t := m[z\leftarrow S\langle v\rangle] \to_{\mathsf{we}} S\langle m\{z\leftarrow v\}\rangle =: u$ and $t \to_{\mathsf{wm}} m'[z\leftarrow S\langle v\rangle] =: s$ with $m \to_{\mathsf{wm}} m'$: then

$$u \to_{\mathsf{wm}} S\langle m'\{z\leftarrow v\}\rangle \,{}_{\mathsf{we}}\!\leftarrow u$$

- *Step at the Root for* $t \to_{\mathsf{we}} u$ and ES *right for* $t \to_{\mathsf{wm}} s$, *i.e.*

$$\begin{aligned} t &:= m[z\leftarrow v[x_1\leftarrow t_1]\dots[x_n\leftarrow t_n]] \\ &\to_{\mathsf{we}} m\{z\leftarrow v\}[x_1\leftarrow t_1]\dots[x_n\leftarrow t_n] =: u \end{aligned}$$

  and $t \to_{\mathsf{wm}} m[z\leftarrow v[x_1\leftarrow t_1]\dots[x_j\leftarrow t'_j]\dots[x_n\leftarrow t_n]] =: s$ for some $n > 0$, and $t_j \to_{\mathsf{wm}} t'_j$ for some $1 \leq j \leq n$: then, $u \to_{\mathsf{wm}} m\{z\leftarrow v\}[x_1\leftarrow t_1]\dots[x_j\leftarrow t'_j]\dots[x_n\leftarrow t_n]\ {}_{\mathsf{we}}\!\leftarrow s$;
- *Application Left for* $t \to_{\mathsf{e}} u$ and *Application Right for* $t \to_{\mathsf{sm}} s$, *i.e.* $t := m\tilde{t} \to_{\mathsf{we}} m'\tilde{t} =: u$ and $t \to_{\mathsf{wm}} m\tilde{t}' =: s$ with $m \to_{\mathsf{e}} m'$ and $\tilde{t} \to_{\mathsf{wm}} \tilde{t}'$: then, $t \to_{\mathsf{wm}} m'\tilde{t}'\ {}_{\mathsf{we}}\!\leftarrow u$;
- *Application Left for both* $t \to_{\mathsf{we}} u$ *and* $t \to_{\mathsf{wm}} s$, *i.e.* $t := m\tilde{t} \to_{\mathsf{we}} m'\tilde{t} =: u$ and $t \to_{\mathsf{wm}} m''\tilde{t} =: s$ with $m'\ {}_{\mathsf{we}}\!\leftarrow m \to_{\mathsf{wm}} m''$: by *i.h.*, there exists $\tilde{u} \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{wm}} \tilde{u}\ {}_{\mathsf{we}}\!\leftarrow m''$, hence $u \to_{\mathsf{wm}} \tilde{u}\tilde{t}\ {}_{\mathsf{we}}\!\leftarrow s$;
- *Application Left for* $t \to_{\mathsf{we}} u$ and *Step at the Root for* $t \to_{\mathsf{wm}} s$; *i.e.*,

$$t := (\lambda x.\tilde{t})[x_1\leftarrow t_1]\dots[x_n\leftarrow t_n]m \to_{\mathsf{we}} (\lambda x.\tilde{t})[x_1\leftarrow t_1]\dots[x_j\leftarrow t'_j]\dots[x_n\leftarrow t_n]m =: u$$

  with $n > 0$ and $t_j \to_{\mathsf{we}} t'_j$ for some $1 \leq j \leq n$, and

$$t \to_{\mathsf{wm}} \tilde{t}[x\leftarrow m][x_1\leftarrow t_1]\dots[x_n\leftarrow t_n] =: s$$

  Then,

$$u \to_{\mathsf{wm}} \tilde{t}[x\leftarrow m][x_1\leftarrow t_1]\dots[x_j\leftarrow t'_j]\dots[x_n\leftarrow t_n]\ {}_{\mathsf{we}}\!\leftarrow s;$$

- *Application Right for* $t \to_{\mathsf{we}} u$ and *Application Left for* $t \to_{\mathsf{sm}} s$, *i.e.* $t := \tilde{t}m \to_{\mathsf{we}} \tilde{t}m' =: u$ and $t \to_{\mathsf{wm}} \tilde{t}'m =: s$ with $m \to_{\mathsf{we}} m'$ and $\tilde{t} \to_{\mathsf{wm}} \tilde{t}'$: then, $u \to_{\mathsf{wm}} \tilde{t}'m'\ {}_{\mathsf{we}}\!\leftarrow s$;
- *Application Right for both* $t \to_{\mathsf{we}} u$ *and* $t \to_{\mathsf{wm}} s$, *i.e.* $t := \tilde{t}m \to_{\mathsf{we}} \tilde{t}m' =: u$ and $t \to_{\mathsf{wm}} \tilde{t}m'' =: s$ with $m'\ {}_{\mathsf{we}}\!\leftarrow m \to_{\mathsf{wm}} m''$: by *i.h.*, there exists $\tilde{u} \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{wm}} \tilde{u}\ {}_{\mathsf{we}}\!\leftarrow m''$, hence $u \to_{\mathsf{wm}} \tilde{t}\tilde{u}\ {}_{\mathsf{we}}\!\leftarrow s$;
- *Application Right for* $t \to_{\mathsf{we}} u$ and *Step at the Root for* $t \to_{\mathsf{wm}} s$, *i.e.* $t := S\langle\lambda x.\tilde{t}\rangle m \to_{\mathsf{we}} S\langle\lambda x.\tilde{t}\rangle m' =: u$ with $m \to_{\mathsf{we}} m'$, and $t \to_{\mathsf{wm}} S\langle\tilde{t}[x\leftarrow m]\rangle =: s$: then,

$$u \to_{\mathsf{wm}} S\langle\tilde{t}[x\leftarrow m']\rangle\ {}_{\mathsf{we}}\!\leftarrow s$$

- ES *left for* $t \to_{\mathsf{we}} u$ and ES *right for* $t \to_{\mathsf{wm}} s$, *i.e.* $t := m[x\leftarrow\tilde{t}] \to_{\mathsf{we}} m'[x\leftarrow\tilde{t}] =: u$ and $t \to_{\mathsf{wm}} m[x\leftarrow\tilde{t}'] =: s$ with $m \to_{\mathsf{e}} m'$ and $\tilde{t} \to_{\mathsf{wm}} \tilde{t}'$: then, $u \to_{\mathsf{wm}} m'[x\leftarrow\tilde{t}']\ {}_{\mathsf{we}}\!\leftarrow s$;
- ES *left for both* $t \to_{\mathsf{we}} u$ *and* $t \to_{\mathsf{wm}} s$, *i.e.* $t := m[x\leftarrow\tilde{t}] \to_{\mathsf{e}} m'[x\leftarrow\tilde{t}] =: u$ and $t \to_{\mathsf{wm}} m''[x\leftarrow\tilde{t}] =: s$ with $m'\ {}_{\mathsf{we}}\!\leftarrow m \to_{\mathsf{wm}} m''$: by *i.h.*,there is $\tilde{u} \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{wm}} \tilde{u}\ {}_{\mathsf{we}}\!\leftarrow m''$, hence $u \to_{\mathsf{wm}} \tilde{u}[x\leftarrow\tilde{t}]\ {}_{\mathsf{we}}\!\leftarrow s$;
- ES *right for* $t \to_{\mathsf{we}} u$ and ES *left for* $t \to_{\mathsf{wm}} s$, *i.e.* $t := \tilde{t}[x\leftarrow m] \to_{\mathsf{we}} \tilde{t}[x\leftarrow m'] =: u$ and $t \to_{\mathsf{wm}} \tilde{t}'[x\leftarrow m] =: s$ with $m \to_{\mathsf{we}} m'$ and $\tilde{t} \to_{\mathsf{wm}} \tilde{t}'$: then, $u \to_{\mathsf{wm}} \tilde{t}'[x\leftarrow m']\ {}_{\mathsf{we}}\!\leftarrow s$;
- ES *right for both* $t \to_{\mathsf{we}} u$ *and* $t \to_{\mathsf{wm}} s$, *i.e.* $t := \tilde{t}[x\leftarrow m] \to_{\mathsf{we}} \tilde{t}[x\leftarrow m'] =: u$ and $t \to_{\mathsf{wm}} \tilde{t}[x\leftarrow m''] =: s$ with $m\ {}_{\mathsf{e}}\!\leftarrow m' \to_{\mathsf{wm}} m''$: by *i.h.*, there is $\tilde{u} \in \Lambda_{\mathsf{vsc}}$ such that $m \to_{\mathsf{wm}} \tilde{u}\ {}_{\mathsf{we}}\!\leftarrow m''$, hence $u \to_{\mathsf{wm}} \tilde{t}[x\leftarrow\tilde{u}]\ {}_{\mathsf{we}}\!\leftarrow s$.

4. It follows immediately from Lemma 13.7.11.5 (diamond property for $\to_{\mathsf{wm}}$ and $\to_{\mathsf{we}}$), Lemma 13.7.11.2 (strong commutation of $\to_{\mathsf{wm}}$ and $\to_{\mathsf{we}}$), and Hindley-Rosen ([Bar84, p. 3.3.5]).
5. We prove that $\to_{\mathsf{sm}}$ is diamond, *i.e.* if $u\ {}_{\mathsf{sm}}\!\leftarrow t \to_{\mathsf{sm}} s$ with $u \neq s$ then there exists $t' \in \Lambda_{\mathsf{vsc}}$ such that $u \to_{\mathsf{sm}} t'\ {}_{\mathsf{sm}}\!\leftarrow s$ (the proof that $\to_{\mathsf{se}}$ is diamond is analogue). The proof is by structural induction on t, doing case analysis on $t \to_{\mathsf{sm}} u$ and $t \to_{\mathsf{sm}} s$:

- *Under λ-abstraction for both $t \to_{\mathsf{sm}} u$ and $t \to_{\mathsf{sm}} s$; i.e., $t = \lambda x.m \to_{\mathsf{sm}} \lambda x.\tilde{t} = u$* and $t = \lambda x.m \to_{\mathsf{sm}} \lambda x.\tilde{u} = s$, with $m \to_{\mathsf{sm}} \tilde{t}$ and $m \to_{\mathsf{sm}} \tilde{u}$. By *i.h.* there exists m' such that $\tilde{t} \to_{\mathsf{sm}} m' \ {}_{\mathsf{sm}}\!\leftarrow \tilde{u}$ and so $u = \lambda x.\tilde{t} \to_{\mathsf{sm}} \lambda x.m' \ {}_{\mathsf{sm}}\!\leftarrow \lambda x.\tilde{u} = s$.
- *Application right for $t \to_{\mathsf{sm}} u$ and application left for $t \to_{\mathsf{sm}} s$; i.e., $t = m\tilde{t} \to_{\mathsf{sm}} m\tilde{t'} = u$* and $t = m\tilde{t} \to_{\mathsf{sm}} m'\tilde{t} = s$. There are several sub-cases to this:
  - Let $t = m\tilde{t} = mO\langle\tilde{\tilde{t}}\rangle \to_{\mathsf{wm}} mO\langle\tilde{\tilde{t}'}\rangle = u$, with $\tilde{\tilde{t}} \mapsto_{\mathsf{m}} \tilde{\tilde{t}'}$, and $t = R\langle\tilde{m}\rangle\tilde{t} \to_{\mathsf{sm}} R\langle\tilde{m'}\rangle\tilde{t}$, with $\tilde{m} \to_{\mathsf{wm}} \tilde{m'}$. Let $t' \coloneqq R\langle\tilde{m'}\rangle O\langle\tilde{\tilde{t}'}\rangle$, having that

    $$u = R\langle\tilde{m}\rangle O\langle\tilde{\tilde{t}'}\rangle \to_{\mathsf{sm}} t' \ {}_{\mathsf{sm}}\!\leftarrow R\langle\tilde{m'}\rangle O\langle\tilde{\tilde{t}}\rangle = s$$

    Note that $u \to_{\mathsf{sm}} t'$ holds because every rigid context is an open context.
  - Let $t = m\tilde{t} = mO_1\langle\tilde{\tilde{t}}\rangle \to_{\mathsf{wm}} mO_1\langle\tilde{\tilde{t}'}\rangle = u$, with $\tilde{\tilde{t}} \mapsto_{\mathsf{m}} \tilde{\tilde{t}'}$, and $t = O_2\langle\tilde{m}\rangle\tilde{t} \to_{\mathsf{wm}} O_2\langle\tilde{m'}\rangle\tilde{t}$, with $\tilde{m} \mapsto_{\mathsf{m}} \tilde{m'}$. Then the statement holds by Lemma 13.7.11.5 (Properties of the VSC).
  - Let $t = mS\langle\tilde{\tilde{t}}\rangle \to_{\mathsf{sm}} mS\langle\tilde{\tilde{t}'}\rangle = u$, with m a rigid term and $\tilde{\tilde{t}} \to_{\mathsf{wm}} \tilde{\tilde{t}'}$, and $t = R\langle\tilde{m}\rangle\tilde{t} \to_{\mathsf{sm}} R\langle\tilde{m'}\rangle\tilde{t} = s$, with $\tilde{m} \to_{\mathsf{wm}} \tilde{m'}$. Let $t' \coloneqq R\langle\tilde{m'}\rangle S\langle\tilde{\tilde{t}'}\rangle$, having that

    $$u = R\langle\tilde{m}\rangle S\langle\tilde{\tilde{t}'}\rangle \to_{\mathsf{sm}} t' \ {}_{\mathsf{sm}}\!\leftarrow R\langle\tilde{m'}\rangle S\langle\tilde{\tilde{t}}\rangle = s$$

    Note that $t' \ {}_{\mathsf{sm}}\!\leftarrow s$ holds because $R\langle\tilde{m'}\rangle$ is a rigid term —by Lemma 13.7.10.1 (Properties of rigid terms).
  - Let $t = mS\langle\tilde{\tilde{t}}\rangle \to_{\mathsf{sm}} mS\langle\tilde{\tilde{t}'}\rangle = u$, with m a rigid term and $\tilde{\tilde{t}} \to_{\mathsf{wm}} \tilde{\tilde{t}'}$, and $t = O\langle\tilde{m}\rangle\tilde{t} \to_{\mathsf{sm}} O\langle\tilde{m'}\rangle\tilde{t}$, with $\tilde{m} \mapsto_{\mathsf{m}} \tilde{m'}$. Let $t' \coloneqq O\langle\tilde{m'}\rangle S\langle\tilde{\tilde{t}'}\rangle$, having that

    $$u = O\langle\tilde{m}\rangle S\langle\tilde{\tilde{t}'}\rangle \to_{\mathsf{sm}} t' \ {}_{\mathsf{sm}}\!\leftarrow O\langle\tilde{m'}\rangle S\langle\tilde{\tilde{t}}\rangle = s$$

    Note that $t' \ {}_{\mathsf{sm}}\!\leftarrow s$ holds because the fact that m is a rigid term and that $m = O\langle\tilde{m}\rangle \to_{\mathsf{wm}} O\langle\tilde{m'}\rangle$ imply that $O\langle\tilde{m'}\rangle$ is a rigid term —by Lemma 13.7.10.2 (Properties of rigid terms).
- *Application right for both $t \to_{\mathsf{sm}} u$ and $t \to_{\mathsf{sm}} s$; i.e., $t = m\tilde{t} \to_{\mathsf{sm}} m\tilde{t'} = u$* and $t = m\tilde{t} \to_{\mathsf{sm}} m\tilde{t''} = s$. By *i.h.* there exists $\tilde{u} \in \Lambda_{\mathsf{vsc}}$ such that $\tilde{t'} \to_{\mathsf{sm}} \tilde{u} \ {}_{\mathsf{sm}}\!\leftarrow \tilde{t''}$. The analysis of the sub-cases, depending on the open/strong/rigid type contexts involved in $t \to_{\mathsf{sm}} u$ and $t \to_{\mathsf{sm}} s$, follows the same schema as for the previous item, all showing that

  $$u = m\tilde{t'} \to_{\mathsf{sm}} m\tilde{u} \ {}_{\mathsf{sm}}\!\leftarrow m\tilde{t''} = s$$

- *Application left for both $t \to_{\mathsf{sm}} u$ and $t \to_{\mathsf{sm}} s$; i.e., $t = m\tilde{t} \to_{\mathsf{sm}} m'\tilde{t} = u$* and $t = m\tilde{t} \to_{\mathsf{sm}} m''\tilde{t} = s$. By *i.h.* there exists $\tilde{u} \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{sm}} \tilde{u} \ {}_{\mathsf{sm}}\!\leftarrow m''$. The analysis of the sub-cases, depending on the open/strong/rigid type contexts involved in $t \to_{\mathsf{sm}} u$ and $t \to_{\mathsf{sm}} s$, follows the same schema as for the previous item, all showing that

  $$u = m'\tilde{t} \to_{\mathsf{sm}} \tilde{u}\tilde{t} \ {}_{\mathsf{sm}}\!\leftarrow m''\tilde{t} = s$$

- ES *right for $t \to_{\mathsf{sm}} u$ and ES left for $t \to_{\mathsf{sm}} s$; i.e., $t = m[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} m[x{\leftarrow}\tilde{t'}] = u$* and $t = m[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} m'[x{\leftarrow}\tilde{t}]$. There are several sub-cases to this:

– Let $t = m[x{\leftarrow}O\langle\tilde{\tilde{t}}\rangle] \to_{\mathsf{sm}} m[x{\leftarrow}O\langle\tilde{\tilde{t}}'\rangle] = u$, with $\tilde{\tilde{t}} \mapsto_{\mathsf{m}} \tilde{\tilde{t}}'$, and $t = O\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} O\langle\tilde{m}'\rangle[x{\leftarrow}\tilde{t}] = s$, with $\tilde{m} \mapsto_{\mathsf{m}} \tilde{m}'$. Then the statement holds by Lemma 13.7.11.2 (Properties of the VSC).

– Let $t = m[x{\leftarrow}O\langle\tilde{\tilde{t}}\rangle] \to_{\mathsf{sm}} m[x{\leftarrow}O\langle\tilde{\tilde{t}}'\rangle] = u$, with $\tilde{\tilde{t}} \mapsto_{\mathsf{m}} \tilde{\tilde{t}}'$, and $t = S\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} S\langle\tilde{m}'\rangle[x{\leftarrow}\tilde{t}] = s$, with $\tilde{m} \to_{\mathsf{wm}} \tilde{m}'$ and $\tilde{t}$ is a rigid term. Let $t' \coloneqq S\langle\tilde{m}'\rangle[x{\leftarrow}O\langle\tilde{\tilde{t}}'\rangle]$, having that

$$u = S\langle\tilde{m}\rangle[x{\leftarrow}O\langle\tilde{\tilde{t}}'\rangle] \to_{\mathsf{sm}} t' \;{}_{\mathsf{sm}}{\leftarrow} S\langle\tilde{m}'\rangle[x{\leftarrow}O\langle\tilde{\tilde{t}}\rangle] = s$$

Note that $u \to_{\mathsf{sm}} t'$ holds because the fact that$\tilde{t}$ is a rigid term and that $\tilde{t} = O\langle\tilde{\tilde{t}}\rangle \to_{\mathsf{sm}} O\langle\tilde{\tilde{t}}'\rangle$ imply that $O\langle\tilde{\tilde{t}}'\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms).

– Let $t = m[x{\leftarrow}O\langle\tilde{\tilde{t}}\rangle] \to_{\mathsf{sm}} m[x{\leftarrow}O\langle\tilde{\tilde{t}}'\rangle] = u$, with $\tilde{\tilde{t}} \mapsto_{\mathsf{m}} \tilde{\tilde{t}}'$, and $t = R\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} R\langle\tilde{m}'\rangle[x{\leftarrow}\tilde{t}] = s$, with $\tilde{m} \to_{\mathsf{wm}} \tilde{m}'$ and $\tilde{t}$ is a rigid term. Let $t' \coloneqq R\langle\tilde{m}'\rangle[x{\leftarrow}O\langle\tilde{\tilde{t}}'\rangle]$, having that

$$u = R\langle\tilde{m}\rangle[x{\leftarrow}O\langle\tilde{\tilde{t}}'\rangle] \to_{\mathsf{sm}} t' \;{}_{\mathsf{sm}}{\leftarrow} R\langle\tilde{m}'\rangle[x{\leftarrow}O\langle\tilde{\tilde{t}}\rangle] = s$$

Note that $u \to_{\mathsf{sm}} t'$ holds because the fact that $\tilde{t}$ is a rigid term and that $\tilde{t} = O\langle\tilde{\tilde{t}}\rangle \to_{\mathsf{sm}} O\langle\tilde{\tilde{t}}'\rangle$ imply $O\langle\tilde{\tilde{t}}'\rangle$ is a rigid term —by Lemma 13.7.10.2 (Properties of rigid terms).

– Let $t = m[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] \to_{\mathsf{sm}} m[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle] = u$, with $\tilde{\tilde{t}} \to_{\mathsf{wm}} \tilde{\tilde{t}}'$, and $t = O\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} O\langle\tilde{m}'\rangle[x{\leftarrow}\tilde{t}] = s$, with $\tilde{m} \mapsto_{\mathsf{m}} \tilde{m}'$. Let $t' \coloneqq O\langle\tilde{m}'\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle]$, having that

$$u = O\langle\tilde{m}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle] \to_{\mathsf{sm}} t' \;{}_{\mathsf{sm}}{\leftarrow} O\langle\tilde{m}'\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] = s$$

– Let $t = m[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] \to_{\mathsf{sm}} m[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle] = u$, with $\tilde{\tilde{t}} \to_{\mathsf{wm}} \tilde{\tilde{t}}'$, and $t = S\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} S\langle\tilde{m}'\rangle[x{\leftarrow}\tilde{t}] = s$, with $\tilde{m} \to_{\mathsf{wm}} \tilde{m}'$ and $\tilde{t}$ is a rigid term. Let $t' \coloneqq S\langle\tilde{m}'\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle]$, having that

$$u = S\langle\tilde{m}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle] \to_{\mathsf{sm}} t' \;{}_{\mathsf{sm}}{\leftarrow} S\langle\tilde{m}'\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] = s$$

Note that $u \to_{\mathsf{sm}} t'$ holds because the fact that $\tilde{t}$ is a rigid term and that $\tilde{t} = R\langle\tilde{\tilde{t}}\rangle \to_{\mathsf{sm}} R\langle\tilde{\tilde{t}}'\rangle$ imply that $R\langle\tilde{\tilde{t}}'\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms).

– Let $t = m[x{\leftarrow}R_1\langle\tilde{\tilde{t}}\rangle] \to_{\mathsf{sm}} m[x{\leftarrow}R_1\langle\tilde{\tilde{t}}'\rangle] = u$, with $\tilde{\tilde{t}} \to_{\mathsf{wm}} \tilde{\tilde{t}}'$, and $t = R_2\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} R_2\langle\tilde{m}'\rangle[x{\leftarrow}\tilde{t}] = s$, with $\tilde{m} \to_{\mathsf{wm}} \tilde{m}'$ and $\tilde{t}$ is a rigid term. Let $t' \coloneqq R_2\langle\tilde{m}'\rangle[x{\leftarrow}R_1\langle\tilde{\tilde{t}}'\rangle]$, having that

$$u = R_2\langle\tilde{m}\rangle[x{\leftarrow}R_1\langle\tilde{\tilde{t}}'\rangle] \to_{\mathsf{sm}} t' \;{}_{\mathsf{sm}}{\leftarrow} R_2\langle\tilde{m}'\rangle[x{\leftarrow}R_1\langle\tilde{\tilde{t}}\rangle] = s$$

Note that $u \to_{\mathsf{sm}} t'$ holds because the fact that $\tilde{t}$ is a rigid term and that $\tilde{t} = R_1\langle\tilde{\tilde{t}}\rangle \to_{\mathsf{sm}} R_1\langle\tilde{\tilde{t}}'\rangle$ imply that $R_1\langle\tilde{\tilde{t}}'\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms).

– Let $t = m[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] \to_{\mathsf{sm}} m[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle] = u$, with $\tilde{\tilde{t}} \to_{\mathsf{wm}} \tilde{\tilde{t}}'$ and m is a rigid term, and $t = O\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} O\langle\tilde{m'}\rangle[x{\leftarrow}\tilde{t}] = s$, with $\tilde{m} \mapsto_{\mathsf{m}} \tilde{m'}$. Let $t' \coloneqq O\langle\tilde{m'}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle]$, having that

$$u = O\langle\tilde{m}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle] \to_{\mathsf{sm}} t' \;{}_{\mathsf{sm}}{\leftarrow} O\langle\tilde{m'}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] = s$$

Note that $s \;{}_{\mathsf{sm}}{\leftarrow} t'$ holds because the fact that $\tilde{t}$ is a rigid term and that $\tilde{t} = R\langle\tilde{\tilde{t}}\rangle \to_{\mathsf{sm}} R\langle\tilde{\tilde{t}}'\rangle$ imply that $R\langle\tilde{\tilde{t}}'\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms).

– Let $t = m[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] \to_{\mathsf{sm}} m[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle] = u$, with $\tilde{\tilde{t}} \to_{\mathsf{wm}} \tilde{\tilde{t}}'$ and m is a rigid term, and $t = S\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} S\langle\tilde{m'}\rangle[x{\leftarrow}\tilde{t}] = s$, with $\tilde{m} \to_{\mathsf{wm}} \tilde{m'}$ and $\tilde{t}$ is a rigid term. Let $t' \coloneqq S\langle\tilde{m'}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle]$, having that

$$u = S\langle\tilde{m}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle] \to_{\mathsf{sm}} t' \;{}_{\mathsf{sm}}{\leftarrow} S\langle\tilde{m'}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] = s$$

Note that $u \to_{\mathsf{sm}} t'$ holds because the fact that $\tilde{t}$ is a rigid term and that $\tilde{t} = R\langle\tilde{\tilde{t}}\rangle \to_{\mathsf{sm}} R\langle\tilde{\tilde{t}}'\rangle$ imply that $R\langle\tilde{\tilde{t}}'\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms). Moreover, note that $t' \;{}_{\mathsf{sm}}{\leftarrow} s$ holds because the fact that m is a rigid term and that $m = S\langle\tilde{m}\rangle \to_{\mathsf{sm}} S\langle\tilde{m'}\rangle$ imply that $S\langle\tilde{m'}\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms).

– Let $t = m[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] \to_{\mathsf{sm}} m[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle] = u$, with $\tilde{\tilde{t}} \to_{\mathsf{wm}} \tilde{\tilde{t}}'$ and m is a rigid term, and $t = R\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} R\langle\tilde{m'}\rangle[x{\leftarrow}\tilde{t}] = s$, with $\tilde{m} \to_{\mathsf{wm}} \tilde{m'}$ and $\tilde{t}$ is a rigid term. Let $t' \coloneqq R\langle\tilde{m'}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle]$, having that

$$u = R\langle\tilde{m}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}'\rangle] \to_{\mathsf{sm}} t' \;{}_{\mathsf{sm}}{\leftarrow} R\langle\tilde{m'}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] = s$$

Note that $u \to_{\mathsf{sm}} t'$ holds because the fact that $\tilde{t}$ is a rigid term and that $\tilde{t} = R\langle\tilde{\tilde{t}}\rangle \to_{\mathsf{sm}} R\langle\tilde{\tilde{t}}'\rangle$ imply that $R\langle\tilde{\tilde{t}}'\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms). Moreover, note that $t' \;{}_{\mathsf{sm}}{\leftarrow} s$ holds because the fact that m is a rigid term and that $m = R\langle\tilde{m}\rangle \to_{\mathsf{sm}} R\langle\tilde{m'}\rangle$ imply that $R\langle\tilde{m'}\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms).

- ES *right for both* $t \to_{\mathsf{sm}} u$ *and* $t \to_{\mathsf{sm}} s$; *i.e.*, $t = m[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} m[x{\leftarrow}\tilde{t'}] = u$ and $t = m[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} m[x{\leftarrow}\tilde{t''}] = s$. By *i.h.* there exists $\tilde{u} \in \Lambda_{\mathsf{vsc}}$ such that $\tilde{t'} \to_{\mathsf{sm}} \tilde{u} \;{}_{\mathsf{sm}}{\leftarrow} \tilde{t''}$. The analysis of the sub-cases, depending on the open/strong/rigid type contexts involved in $t \to_{\mathsf{sm}} u$ and $t \to_{\mathsf{sm}} s$, follows the same schema as for the previous item, all showing that

$$u = m[x{\leftarrow}\tilde{t'}] \to_{\mathsf{sm}} m[x{\leftarrow}\tilde{u}] \;{}_{\mathsf{sm}}{\leftarrow} m[x{\leftarrow}\tilde{t''}] = s$$

- ES *left for both* $t \to_{\mathsf{sm}} u$ *and* $t \to_{\mathsf{sm}} s$; *i.e.*, $t = m[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} m'[x{\leftarrow}\tilde{t}] = u$ and $t = m''[x{\leftarrow}\tilde{t}]$. By *i.h.* there exists $\tilde{u} \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{sm}} \tilde{u} \;{}_{\mathsf{sm}}{\leftarrow} m''$. The analysis of the sub-cases, depending on the open/strong/rigid type contexts involved in $t \to_{\mathsf{sm}} u$ and $t \to_{\mathsf{sm}} s$, follows the same schema as for the previous item, all showing that

$$u = m'[x{\leftarrow}\tilde{t}] \to_{\mathsf{sm}} \tilde{u}[x{\leftarrow}\tilde{t}] \;{}_{\mathsf{sm}}{\leftarrow} m''[x{\leftarrow}\tilde{t}] = s$$

The proof that $\to_{\mathsf{se}}$ is diamond (*i.e.*, if $u \;{}_{\mathsf{se}}{\leftarrow} t \to_{\mathsf{se}} s$ with $u \neq s$ then there exists $t' \in \Lambda_{\mathsf{vsc}}$ such that $u \to_{\mathsf{se}} t' \;{}_{\mathsf{se}}{\leftarrow} s$) follows the same schema as for $\to_{\mathsf{sm}}$.

6. We show that $\to_{\mathsf{se}}$ and $\to_{\mathsf{sm}}$ strongly commute; *i.e.*, if $u \ {}_{\mathsf{se}}\!\leftarrow t \to_{\mathsf{sm}} s$, then $u \neq s$ and there is $t' \in \Lambda_{\mathsf{vsc}}$ such that $u \to_{\mathsf{sm}} t' \ {}_{\mathsf{se}}\!\leftarrow s$. The proof is by structural induction on t, doing case analysis on $t \ {}_{\mathsf{se}}\!\leftarrow u$ and $t \to_{\mathsf{sm}} s$:
   - *Under λ-abstraction for both $t \ {}_{\mathsf{se}}\!\leftarrow u$ and $t \to_{\mathsf{sm}} s$; i.e., $t = \lambda x.\tilde{t} \ {}_{\mathsf{se}}\!\leftarrow \lambda x.m = u$ and $t = \lambda x.m \to_{\mathsf{sm}} \lambda x.\tilde{u} = s$, with $\tilde{t} \ {}_{\mathsf{we}}\!\leftarrow m \to_{\mathsf{wm}} \tilde{u}$.* By Lemma 13.7.11.5 (Properties of the VSC), there exists m' such that $\tilde{t} \to_{\mathsf{sm}} m' \ {}_{\mathsf{se}}\!\leftarrow \tilde{u}$, and so $u = \lambda x.\tilde{t} \to_{\mathsf{wm}} \lambda x.m' \ {}_{\mathsf{we}}\!\leftarrow \lambda x.\tilde{u}$.
   - *Application right for $u \ {}_{\mathsf{se}}\!\leftarrow t$ and application left for $t \to_{\mathsf{sm}} s$; i.e., $u = m\tilde{t}' \ {}_{\mathsf{se}}\!\leftarrow m\tilde{t} = t$ and $t = m\tilde{t} \to_{\mathsf{sm}} m'\tilde{t} = s$.* There are several sub-cases to this:
     - Let $u = mO\langle \tilde{\tilde{t}}' \rangle \ {}_{\mathsf{se}}\!\leftarrow mO\langle \tilde{\tilde{t}} \rangle = t$, with $\tilde{\tilde{t}}' \ {}_{\mathsf{we}}\!\leftarrow \tilde{\tilde{t}}$, and $t = R\langle \tilde{m} \rangle \tilde{t} \to_{\mathsf{sm}} R\langle \tilde{m'} \rangle \tilde{t} = s$, with $\tilde{m} \to_{\mathsf{wm}} \tilde{m'}$. Let $t' = R\langle \tilde{m'} \rangle O\langle \tilde{\tilde{t}}' \rangle$, having that
     $$u = R\langle \tilde{m} \rangle O\langle \tilde{\tilde{t}}' \rangle \to_{\mathsf{sm}} t' \ {}_{\mathsf{se}}\!\leftarrow R\langle \tilde{m'} \rangle O\langle \tilde{\tilde{t}} \rangle = s$$
     - Let $u = mO_1\langle \tilde{\tilde{t}}' \rangle \ {}_{\mathsf{se}}\!\leftarrow mO_1\langle \tilde{\tilde{t}} \rangle = t$, and $t = O_2\langle \tilde{m} \rangle \tilde{t} \to_{\mathsf{sm}} O_2\langle \tilde{m'} \rangle \tilde{t} = s$, with $\tilde{m} \to_{\mathsf{sm}} \tilde{m'}$. Then the statement holds by Lemma 13.7.11.6 (Properties of the VSC)
     - Let $u = mS\langle \tilde{\tilde{t}}' \rangle \ {}_{\mathsf{se}}\!\leftarrow mS\langle \tilde{\tilde{t}} \rangle = t$, with m a rigid term and $\tilde{\tilde{t}}' \ {}_{\mathsf{we}}\!\leftarrow \tilde{\tilde{t}}$, and $t = R\langle \tilde{m} \rangle \tilde{t} \to_{\mathsf{sm}} R\langle \tilde{m'} \rangle \tilde{t} = s$, with $\tilde{m} \to_{\mathsf{wm}} \tilde{m'}$. Let $t' = R\langle \tilde{m'} \rangle S\langle \tilde{\tilde{t}}' \rangle$, having that
     $$u = R\langle \tilde{m} \rangle S\langle \tilde{\tilde{t}}' \rangle \to_{\mathsf{sm}} t' \ {}_{\mathsf{se}}\!\leftarrow R\langle \tilde{m'} \rangle S\langle \tilde{\tilde{t}} \rangle = s$$
     Note that $t' \ {}_{\mathsf{se}}\!\leftarrow s$ holds because $R\langle \tilde{m'} \rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms).
     - Let $u = mS\langle \tilde{\tilde{t}}' \rangle \ {}_{\mathsf{se}}\!\leftarrow mS\langle \tilde{\tilde{t}} \rangle = t$, with m a rigid term and $\tilde{\tilde{t}}' \ {}_{\mathsf{we}}\!\leftarrow \tilde{\tilde{t}}$, and $t = O\langle \tilde{m} \rangle \tilde{t} \to_{\mathsf{sm}} O\langle \tilde{m'} \rangle \tilde{t} = s$, with $\tilde{m} \mapsto_{\mathsf{m}} \tilde{m'}$. Let $t' = O\langle \tilde{m'} \rangle S\langle \tilde{\tilde{t}}' \rangle$, having that
     $$u = O\langle \tilde{m} \rangle S\langle \tilde{\tilde{t}}' \rangle \to_{\mathsf{sm}} t' \ {}_{\mathsf{se}}\!\leftarrow O\langle \tilde{m'} \rangle S\langle \tilde{\tilde{t}} \rangle = s$$
     Note that $t' \ {}_{\mathsf{se}}\!\leftarrow s$ holds because $O\langle \tilde{m'} \rangle$ is rigid —by Lemma 13.7.10.2 (Properties of rigid terms).
   - *Application left for $u \ {}_{\mathsf{se}}\!\leftarrow t$ and application right for $t \to_{\mathsf{sm}} s$; i.e., $u = m'\tilde{t} \ {}_{\mathsf{se}}\!\leftarrow m\tilde{t} = t$ and $t = m\tilde{t} \to_{\mathsf{sm}} m\tilde{t}' = s$.* There are several sub-cases to this:
     - Let $u = R\langle \tilde{m'} \rangle \tilde{t} \ {}_{\mathsf{se}}\!\leftarrow R\langle \tilde{m} \rangle \tilde{t} = t$, with $\tilde{m'} \ {}_{\mathsf{we}}\!\leftarrow \tilde{m}$, and $t = mO\langle \tilde{\tilde{t}} \rangle \to_{\mathsf{sm}} mO\langle \tilde{\tilde{t}}' \rangle = s$, with $\tilde{\tilde{t}} \mapsto_{\mathsf{m}} \tilde{\tilde{t}}'$. Let $t' = R\langle \tilde{m'} \rangle O\langle \tilde{\tilde{t}}' \rangle$, having that
     $$u = R\langle \tilde{m'} \rangle O\langle \tilde{\tilde{t}} \rangle \to_{\mathsf{sm}} t' \ {}_{\mathsf{se}}\!\leftarrow R\langle \tilde{m} \rangle O\langle \tilde{\tilde{t}}' \rangle = s$$
     - Let $u = O_1\langle \tilde{m'} \rangle \tilde{t} \ {}_{\mathsf{se}}\!\leftarrow O_1\langle \tilde{m} \rangle \tilde{t} = t$, with $\tilde{m'} \ {}_{\mathsf{we}}\!\leftarrow \tilde{m}$, and $t = mO_2\langle \tilde{\tilde{t}} \rangle \to_{\mathsf{sm}} mO_2\langle \tilde{\tilde{t}}' \rangle = s$, with $\tilde{\tilde{t}} \mapsto_{\mathsf{m}} \tilde{\tilde{t}}'$. Then the statement holds by Lemma 13.7.11.6 (Properties of the VSC).
     - Let $u = R\langle \tilde{m'} \rangle \tilde{t} \ {}_{\mathsf{se}}\!\leftarrow R\langle \tilde{m} \rangle \tilde{t} = t$, with $\tilde{m'} \ {}_{\mathsf{we}}\!\leftarrow \tilde{m}$, and $t = mS\langle \tilde{\tilde{t}} \rangle \to_{\mathsf{sm}} mS\langle \tilde{\tilde{t}}' \rangle = s$, with $\tilde{\tilde{t}} \to_{\mathsf{wm}} \tilde{\tilde{t}}'$ and m a rigid term. Let $t' = R\langle \tilde{m'} \rangle S\langle \tilde{\tilde{t}}' \rangle$, having that
     $$u = R\langle \tilde{m'} \rangle S\langle \tilde{\tilde{t}} \rangle \to_{\mathsf{sm}} t' \ {}_{\mathsf{se}}\!\leftarrow R\langle \tilde{m} \rangle S\langle \tilde{\tilde{t}}' \rangle = s$$
     Note that $u \to_{\mathsf{sm}} t'$ holds because $R\langle \tilde{m'} \rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms).

– Let $u = O\langle\tilde{m'}\rangle\tilde{t} \;{}_{\mathsf{se}}\!\leftarrow O\langle\tilde{m}\rangle\tilde{t} = t$, with $\tilde{m'} \;{}_{\mathsf{we}}\!\leftarrow \tilde{m}$, and $t = mS\langle\tilde{\tilde{t}}\rangle \to_{\mathsf{sm}} mS\langle\tilde{\tilde{t}'}\rangle = s$, with m a rigid term and $\tilde{\tilde{t}} \to_{\mathsf{wm}} \tilde{\tilde{t}'}$. Let $t' = O\langle\tilde{m'}\rangle S\langle\tilde{\tilde{t}'}\rangle$, having that

$$u = O\langle\tilde{m'}\rangle S\langle\tilde{\tilde{t}}\rangle \to_{\mathsf{sm}} t' \;{}_{\mathsf{se}}\!\leftarrow O\langle\tilde{m}\rangle S\langle\tilde{\tilde{t}'}\rangle = s$$

Note that $u \to_{\mathsf{sm}} t'$ holds because $O\langle\tilde{m'}\rangle$ is rigid —by Lemma 13.7.10.2 (Properties of rigid terms).

- *Application right for both $u \;{}_{\mathsf{se}}\!\leftarrow t$ and $t \to_{\mathsf{sm}} s$; i.e., $u = m\tilde{t'} \;{}_{\mathsf{se}}\!\leftarrow m\tilde{t} = t$ and $t = m\tilde{t} \to_{\mathsf{sm}} m\tilde{t''} = s$.* By *i.h.*, there exists $\tilde{u} \in \Lambda_{\mathsf{vsc}}$ such that $\tilde{t'} \to_{\mathsf{sm}} \tilde{u} \;{}_{\mathsf{se}}\!\leftarrow \tilde{t''}$. The analysis of the sub-cases, depending on the open/strong/rigid type contexts involved in $u \;{}_{\mathsf{se}}\!\leftarrow t$ and $t \to_{\mathsf{sm}} s$ follows the same schema as for the previous item, all showing that

$$u = m\tilde{t'} \to_{\mathsf{sm}} m\tilde{u} \;{}_{\mathsf{se}}\!\leftarrow m\tilde{t''} = s$$

- *Application left for both $u \;{}_{\mathsf{se}}\!\leftarrow t$ and $t \to_{\mathsf{sm}} s$; i.e., $u = m'\tilde{t} \;{}_{\mathsf{se}}\!\leftarrow m\tilde{t} = t$ and $t = m\tilde{t} \to_{\mathsf{sm}} m''\tilde{t} = s$.* By *i.h.*, there exists $\tilde{u} \in \Lambda_{\mathsf{vsc}}$ such that $m' \to_{\mathsf{sm}} \tilde{u} \;{}_{\mathsf{se}}\!\leftarrow m''$. The analysis of the sub-cases, depending on the open/strong/rigid type contexts involved in $u \;{}_{\mathsf{se}}\!\leftarrow t$ and $t \to_{\mathsf{sm}} s$ follows the same schema as for the previous item, all showing that

$$u = m'\tilde{t} \to_{\mathsf{sm}} \tilde{u}\tilde{t} \;{}_{\mathsf{se}}\!\leftarrow m''\tilde{t} = s$$

- ES *right for $u \;{}_{\mathsf{se}}\!\leftarrow t$ and* ES *left for $t \to_{\mathsf{sm}} s$; i.e., $u = m[x\leftarrow\tilde{t'}] \;{}_{\mathsf{se}}\!\leftarrow m[x\leftarrow\tilde{t}] = t$ and $t = m[x\leftarrow\tilde{t}] \to_{\mathsf{sm}} m'[x\leftarrow\tilde{t}] = s$.* There are several sub-cases to this:
  – Let $u = m[x\leftarrow O\langle\tilde{\tilde{t}'}\rangle] \;{}_{\mathsf{se}}\!\leftarrow m[x\leftarrow O\langle\tilde{\tilde{t}}\rangle] = t$, with $\tilde{\tilde{t}'} \;{}_{\mathsf{e}}\!\leftarrow\!\!\mapsto \tilde{\tilde{t}}$, and $t = O\langle\tilde{m}\rangle[x\leftarrow\tilde{t}] \to_{\mathsf{sm}} O\langle\tilde{m'}\rangle[x\leftarrow\tilde{t}] = s$, with $\tilde{m} \mapsto_{\mathsf{m}} \tilde{m'}$. Then the statement holds by Lemma 13.7.11.6 (Properties of the VSC).
  – Let $u = m[x\leftarrow O\langle\tilde{\tilde{t}'}\rangle] \;{}_{\mathsf{se}}\!\leftarrow m[x\leftarrow O\langle\tilde{\tilde{t}}\rangle] = t$, with $\tilde{\tilde{t}'} \;{}_{\mathsf{e}}\!\leftarrow\!\!\mapsto \tilde{\tilde{t}}$, and $t = S\langle\tilde{m}\rangle[x\leftarrow\tilde{t}] \to_{\mathsf{sm}} S\langle\tilde{m'}\rangle[x\leftarrow\tilde{t}] = s$, with $\tilde{m} \to_{\mathsf{wm}} \tilde{m'}$ and $\tilde{t}$ is a rigid term. Let $t' \coloneqq S\langle\tilde{m'}\rangle[x\leftarrow O\langle\tilde{\tilde{t}'}\rangle]$, having that

$$u = S\langle\tilde{m}\rangle[x\leftarrow O\langle\tilde{\tilde{t}'}\rangle] \to_{\mathsf{sm}} t' \to_{\mathsf{sm}} S\langle\tilde{m'}\rangle[x\leftarrow O\langle\tilde{\tilde{t}}\rangle] = s$$

Note that $u \to_{\mathsf{sm}} t'$ holds because $O\langle\tilde{\tilde{t}'}\rangle$ is a rigid term —by Lemma 13.7.10.2 (Properties of rigid terms).

  – Let $u = m[x\leftarrow O\langle\tilde{\tilde{t}'}\rangle] \;{}_{\mathsf{se}}\!\leftarrow m[x\leftarrow O\langle\tilde{\tilde{t}}\rangle] = t$, with $\tilde{\tilde{t}'} \;{}_{\mathsf{e}}\!\leftarrow\!\!\mapsto \tilde{\tilde{t}}$, and $t = R\langle\tilde{m}\rangle[x\leftarrow\tilde{t}] \to_{\mathsf{sm}} R\langle\tilde{m'}\rangle[x\leftarrow\tilde{t}] = s$, with $\tilde{m} \to_{\mathsf{wm}} \tilde{m'}$ and $\tilde{t}$ is a rigid term. Let $t' \coloneqq R\langle\tilde{m'}\rangle[x\leftarrow O\langle\tilde{\tilde{t}'}\rangle]$, having that

$$u = R\langle\tilde{m}\rangle[x\leftarrow O\langle\tilde{\tilde{t}'}\rangle] \to_{\mathsf{sm}} t' \to_{\mathsf{sm}} R\langle\tilde{m'}\rangle[x\leftarrow O\langle\tilde{\tilde{t}}\rangle] = s$$

Note that $u \to_{\mathsf{sm}} t'$ holds because $O\langle\tilde{\tilde{t}'}\rangle$ is a rigid term —by Lemma 13.7.10.2 (Properties of rigid terms).

  – Let $u = m[x\leftarrow R\langle\tilde{\tilde{t}'}\rangle] \;{}_{\mathsf{se}}\!\leftarrow m[x\leftarrow R\langle\tilde{\tilde{t}}\rangle] = t$, with $\tilde{t'} \;{}_{\mathsf{we}}\!\leftarrow \tilde{t}$, and $t = O\langle\tilde{m}\rangle[x\leftarrow\tilde{t}] \to_{\mathsf{sm}} O\langle\tilde{m'}\rangle[x\leftarrow\tilde{t}] = s$, with $\tilde{\tilde{t}} \to_{\mathsf{sm}} \tilde{\tilde{t}'}$. Let $t' \coloneqq O\langle\tilde{m'}\rangle[x\leftarrow R\langle\tilde{t'}\rangle]$, having that

$$u = O\langle\tilde{m}\rangle[x\leftarrow R\langle\tilde{\tilde{t}'}\rangle] \to_{\mathsf{sm}} t' \;{}_{\mathsf{se}}\!\leftarrow O\langle\tilde{m'}\rangle[x\leftarrow R\langle\tilde{\tilde{t}}\rangle] = s$$

– Let $u = m[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle]\ {}_{\mathsf{se}}{\leftarrow}\ m[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] = t$, with $\tilde{\tilde{t}}'\ {}_{\mathsf{we}}{\leftarrow}\ \tilde{\tilde{t}}$, and $t = S\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \rightarrow_{\mathsf{sm}} S\langle\tilde{m'}\rangle[x{\leftarrow}\tilde{t}] = s$, with $\tilde{m} \rightarrow_{\mathsf{wm}} \tilde{m'}$ and $\tilde{t}$ is a rigid term. Let $t' \coloneqq S\langle\tilde{m'}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle]$, having that

$$u = S\langle\tilde{m}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] \rightarrow_{\mathsf{sm}} t'\ {}_{\mathsf{se}}{\leftarrow}\ S\langle\tilde{m'}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] = s$$

Note that $u \rightarrow_{\mathsf{sm}} t'$ holds because $R\langle\tilde{\tilde{t}}\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms).

– Let $u = m[x{\leftarrow}R_1\langle\tilde{\tilde{t}}\rangle]\ {}_{\mathsf{se}}{\leftarrow}\ m[x{\leftarrow}R_1\langle\tilde{\tilde{t}}\rangle] = t$, with $\tilde{\tilde{t}}'\ {}_{\mathsf{we}}{\leftarrow}\ \tilde{\tilde{t}}$, and $t = R_2\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \rightarrow_{\mathsf{sm}} R_2\langle\tilde{m'}\rangle[x{\leftarrow}\tilde{t}]$, with $\tilde{m} \rightarrow_{\mathsf{wm}} \tilde{m'}$ and $\tilde{t}$ is a rigid term. Let

$$t' \coloneqq R_2\langle\tilde{m'}\rangle[x{\leftarrow}R_1\langle\tilde{\tilde{t}}\rangle]$$

having that

$$\begin{array}{rcl} u & = & R_2\langle\tilde{m}\rangle[x{\leftarrow}R_1\langle\tilde{\tilde{t}}\rangle] \\ & \rightarrow_{\mathsf{sm}} & t'\ {}_{\mathsf{se}}{\leftarrow}\ R_2\langle\tilde{m'}\rangle[x{\leftarrow}R_1\langle\tilde{\tilde{t}}\rangle] \end{array}$$

Note that $u \rightarrow_{\mathsf{sm}} t'$ holds because $R_1\langle\tilde{\tilde{t}}\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms).

– Let $u = m[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle]\ {}_{\mathsf{se}}{\leftarrow}\ m[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] = t$, with $\tilde{\tilde{t}}'\ {}_{\mathsf{we}}{\leftarrow}\ \tilde{\tilde{t}}$ and m is a rigid term, and $t = O\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \rightarrow_{\mathsf{sm}} O\langle\tilde{m'}\rangle[x{\leftarrow}\tilde{t}] = s$, with $\tilde{m} \mapsto_{\mathsf{m}} \tilde{m'}$. Let $t' \coloneqq O\langle\tilde{m'}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle]$, having that

$$u = O\langle\tilde{m}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] \rightarrow_{\mathsf{sm}} t'\ {}_{\mathsf{se}}{\leftarrow}\ O\langle\tilde{m'}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] = s$$

Note that $t'\ {}_{\mathsf{se}}{\leftarrow}\ s$ holds because $O\langle\tilde{m'}\rangle$ is a rigid term —by Lemma 13.7.10.2 (Properties of rigid terms).

– Let $t = m[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle]\ {}_{\mathsf{se}}{\leftarrow}\ m[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] = t$, with $\tilde{\tilde{t}}'\ {}_{\mathsf{we}}{\leftarrow}\ \tilde{\tilde{t}}$ and m is a rigid term, and $t = S\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \rightarrow_{\mathsf{sm}} S\langle\tilde{m'}\rangle[x{\leftarrow}\tilde{t}]$, with $\tilde{m} \rightarrow_{\mathsf{wm}} \tilde{m'}$ and $\tilde{t}$ is a rigid term. Let

$$t' \coloneqq S\langle\tilde{m'}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle]$$

having that

$$u = S\langle\tilde{m}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] \rightarrow_{\mathsf{sm}} t'\ {}_{\mathsf{se}}{\leftarrow}\ S\langle\tilde{m'}\rangle[x{\leftarrow}R\langle\tilde{\tilde{t}}\rangle] = s$$

Note that $u \rightarrow_{\mathsf{sm}} t'$ holds because $R\langle\tilde{\tilde{t}}\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms)—, and that $t\ {}_{\mathsf{se}}{\leftarrow}\ s$ holds because $S\langle\tilde{m'}\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms).

– Let $u = m[x{\leftarrow}R_1\langle\tilde{\tilde{t}}\rangle]\ {}_{\mathsf{se}}{\leftarrow}\ m[x{\leftarrow}R_1\langle\tilde{\tilde{t}}\rangle] = t$, with $\tilde{\tilde{t}}'\ {}_{\mathsf{we}}{\leftarrow}\ \tilde{\tilde{t}}$ and m is a rigid term, and $t = R_2\langle\tilde{m}\rangle[x{\leftarrow}\tilde{t}] \rightarrow_{\mathsf{sm}} R_2\langle\tilde{m'}\rangle[x{\leftarrow}\tilde{t}] = s$, with $\tilde{m} \rightarrow_{\mathsf{wm}} \tilde{m'}$ and $\tilde{t}$ is a rigid term. Let $t' \coloneqq R_2\langle\tilde{m'}\rangle[x{\leftarrow}R_1\langle\tilde{\tilde{t}}\rangle]$, having that

$$uR_2\langle\tilde{m}\rangle[x{\leftarrow}R_1\langle\tilde{\tilde{t}}\rangle] \rightarrow_{\mathsf{sm}} t'\ {}_{\mathsf{se}}{\leftarrow}\ R_2\langle\tilde{m'}\rangle[x{\leftarrow}R_1\langle\tilde{\tilde{t}}\rangle] = s$$

Note that $u \rightarrow_{\mathsf{sm}} t'$ holds because $R_1\langle\tilde{\tilde{t}}\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms)—, and that $t'\ {}_{\mathsf{se}}{\leftarrow}\ s$ because $R_2\langle\tilde{m'}\rangle$ is a rigid term —by Lemma 13.7.10.3 (Properties of rigid terms).

- ES *left for* $u \ {}_{\mathsf{se}}\!\leftarrow t$ *and* ES *right for* $t \to_{\mathsf{sm}} s$; *i.e.*, $u = m'[x\!\leftarrow\!\tilde{t}] \ {}_{\mathsf{se}}\!\leftarrow m[x\!\leftarrow\!\tilde{t}] = t$ and $t = m[x\!\leftarrow\!\tilde{t}] \to_{\mathsf{sm}} m[x\!\leftarrow\!\tilde{t'}] = s$. There are several sub-cases to this, all of which follow the same kind of reasoning as for the case ES *right for* $u \ {}_{\mathsf{se}}\!\leftarrow t$ *and* ES *left for* $t \to_{\mathsf{sm}} s$. Therefore, we leave this case for the reader.
- ES *right for both* $u \ {}_{\mathsf{se}}\!\leftarrow t$ *and* $t \to_{\mathsf{sm}} s$; *i.e.*,

  $$u = m[x\!\leftarrow\!\tilde{t'}] \ {}_{\mathsf{se}}\!\leftarrow m[x\!\leftarrow\!\tilde{t}] = t$$

  and

  $$t = m[x\!\leftarrow\!\tilde{t}] \to_{\mathsf{sm}} m[x\!\leftarrow\!\tilde{t''}] = s$$

  By *i.h.* there exists $\tilde{u} \in \Lambda_{\mathsf{vsc}}$ such that $\tilde{t'} \to_{\mathsf{sm}} \tilde{t} \ {}_{\mathsf{se}}\!\leftarrow \tilde{t''}$. The analysis of the sub-cases, depending on the open/strong/rigid type contexts involved in $t \to_{\mathsf{sm}} u$ and $t \to_{\mathsf{sm}} s$, follows the same schema as for the previous item, all showing that

  $$u = m[x\!\leftarrow\!\tilde{t'}] \to_{\mathsf{sm}} m[x\!\leftarrow\!\tilde{u}] \ {}_{\mathsf{se}}\!\leftarrow m[x\!\leftarrow\!\tilde{t''}] = s$$

- ES *left for both* $u \ {}_{\mathsf{se}}\!\leftarrow t$ *and* $t \to_{\mathsf{sm}} s$; *i.e.*,

  $$u = m'[x\!\leftarrow\!\tilde{t}] \ {}_{\mathsf{se}}\!\leftarrow m[x\!\leftarrow\!\tilde{t}] = t$$

  and

  $$t = m[x\!\leftarrow\!\tilde{t}] \to_{\mathsf{sm}} m''[x\!\leftarrow\!\tilde{t}] = s$$

  By *i.h.* there exists $\tilde{u} \in \Lambda_{\mathsf{vsc}}$ such that $\tilde{t'} \to_{\mathsf{sm}} \tilde{t} \ {}_{\mathsf{se}}\!\leftarrow \tilde{t''}$. The analysis of the sub-cases, depending on the open / strong / rigid type contexts involved in $t \to_{\mathsf{sm}} u$ and $t \to_{\mathsf{sm}} s$, follows the same schema as for the previous item, all showing that

  $$u = m'[x\!\leftarrow\!\tilde{t}] \to_{\mathsf{sm}} \tilde{u}[x\!\leftarrow\!\tilde{t}] \ {}_{\mathsf{se}}\!\leftarrow m''[x\!\leftarrow\!\tilde{t}] = s$$

7. It follows immediately from Lemma 13.7.11.5 (Properties of the VSC - diamond property for $\to_{\mathsf{sm}}$ and $\to_{\mathsf{se}}$), from Lemma 13.7.11.5 (Properties of the VSC - strong commutation of $\to_{\mathsf{sm}}$ and $\to_{\mathsf{se}}$) and Hindley-Rosen ([Bar84, p. 3.3.5]).

□

Proposition 13.7.12 (Diamond property for Strong CbV)**.**
$\to_{\mathsf{s}}$ *is diamond.*

Proof. *(Click here to go back to main chapter.)*
By Lemma 13.7.11.7 (Properties of the VSC).

□

Proof. *(Click here to go back to main chapter.)*
We first prove the statement concerning $\to_{\mathsf{s}_\lambda}$. To have the right *i.h.*, we prove simultaneously, by induction on t, the following stronger statements (we recall that all strong inert terms are strong fireballs):

1. *Fireball property*: If t is $\to_{\mathsf{s}_\lambda}$-normal, then t is a strong fireball.
2. *Non-value property*: If t is $\to_{\mathsf{s}_\lambda}$-normal and not a value up to ES, then t is a strong inert term.

Cases:

- *Variable*, *i.e.*, $t = x$: both properties trivially hold, since t is a strong inert term and so a strong fireball.
- *Abstraction*, *i.e.*, $t = \lambda x.u$:
  1. *Non-value property*: vacuously true, as t is an abstraction.
  2. *Fireball property*: Since t is $\to_{\mathsf{s}_\lambda}$-normal, so is u. By *i.h.* applied to u (fireball property), u is a strong fireball and hence so is t (as a strong value).
- *Application*; *i.e.*, $t = t_1 t_2$ (which is not a value up to ES):
  1. *Non-value property*: Since t is $\to_{\mathsf{s}_\lambda}$-normal, so are t_1 and t_2. Moreover, t_1 is not a value up to ES (otherwise t would be a $\to_{\mathsf{sm}}$-redex). By *i.h.* applied to t_1 (non-value property) and to t_2 (fireball property), t_1 is a strong inert term and t_2 is a strong fireball. Thus, t is a strong inert term.
  2. *Fireball property*: We have just proved that t is a strong inert term, which implies that it is a strong fireball.
- *Explicit substitutions*, *i.e.*, $t = t_1[x \leftarrow t_2]$:
  1. *Fireball property*: Since t is $\to_{\mathsf{s}_\lambda}$-normal, so are t_1 and t_2. Moreover, t_2 is not a value up to ES (otherwise t would be a $\to_{\mathsf{se}_\lambda}$-redex). By *i.h.* applied to t_1 (fireball property) and to t_2 (non-value property), t_1 is a strong fireball and t_2 is a strong inert term. Thus, t is a strong fireball.
  2. *Non-value property*: We have just proved that t is a strong fireball. If moreover t is not a value up to ES, then t_1 is not a value up to ES and hence, by *i.h.* applied to t_1 (non-value property), t_1 is a strong inert term. Therefore, t is a strong inert term.

Concerning $\to_{\mathsf{s}}$, to have the right *i.h.*, we prove simultaneously, by induction on t, the following stronger statements (we recall that all super strong inert terms are super strong fireballs):

1. *Fireball property*: If t is $\to_{\mathsf{s}}$-normal, then t is a super strong fireball.
2. *Non-value property*: If t is $\to_{\mathsf{s}}$-normal and neither a value up to ES nor of the form $S\langle x\rangle$, then t is a compound super strong inert term.

The proof is analogous to the proof concerning $\to_{\mathsf{s}_\lambda}$, except for the following cases:

- *Variable*, *i.e.*, $t = x$:
  1. *Fireball property*: it trivially holds since x is a super strong fireball.
  2. *Non-value property*: vacuously true, because t is of the form $S\langle x\rangle$.
- *Explicit substitutions*, *i.e.*, $t = t_1[x \leftarrow t_2]$:
  1. *Fireball property*: Since t is $\to_{\mathsf{s}_\lambda}$-normal, so are t_1 and t_2. Moreover, t_2 is neither a value up to ES (otherwise t would be a $\to_{\mathsf{se}_\lambda}$-redex) nor of the form $S\langle x\rangle$ (otherwise t would be a $\to_{\mathsf{se}_{\mathsf{var}}}$-redex). By *i.h.* applied to t_1 (fireball property) and to t_2 (non-value property), t_1 is a super strong fireball and t_2 is a compound super strong inert term. Thus, t is a super strong fireball.
  2. *Non-value property*: We have just proved that t is a super strong fireball. If moreover t is neither a value up to ES nor of the form $S\langle x\rangle$, then t_1 is neither a value up to ES nor of the form $S\langle x\rangle$ and hence, by *i.h.* applied to t_1 (non-value property), t_1 is a compound super strong inert term. Therefore, t is a compound super strong inert term. □

(Click here to go back to main chapter.)

13.8 Proofs of Chapter 11 (Multi types for Strong CbV)

13.8.1 Strong CbV correctness

Lemma 13.8.1 (Relevance of the Strong CbV type system)**.**
Let $t \in \Lambda_{\mathsf{L}}$ and $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ be a type derivation. If $x \notin \mathsf{fv}(t)$ then $x \notin \mathsf{dom}(\Gamma)$.

Proof. (Click here to go back to main chapter.)
Trivial, by induction on the number of typing rules applied in Φ.

□

(Click here to go back to main chapter.)

Typing properties of Strong CbV-normal forms. Providing the right typing properties that Strong CbV-normal forms satisfy requires, first, proving that type derivations for Strong CbV-normal forms satisfy a kind of "*spreading of co-shrinkingness*" from type contexts into the derived (right-hand side) multi type.

This is achieved by proving that a *rigid* terms satisfy this property, noting that strong inert terms are also rigid. The reason why we have to express this property in terms of rigid terms—while a presentation of it in terms of strong inert terms is also feasible—become clearer when dealing with the Subject Reduction and Subject Expansion properties; in particular, it helps us ensure that the conditions under which the *i.h.* can be applied indeed hold.

Remark 5 (Spreading properties in some call-by-need variants).
Before we continue, let us recall some spreading properties studied for previous cases:

- In Chapter 7 (Multi types for Open CbNeed), we saw that the assuming that the *needed* variables of an expression are assigned to some Inert-type was enough to obtain that the derived (right-hand side) multi type was in Inert too—see Lemma 7.1.2.2 (Typing properties of normal terms).
- Similarly, in Chapter 9 (Multi types for Useful Open CbNeed), we saw that the assuming that the *applied* variables of an expression are assigned to some Tight-type was enough to obtain that the derived (right-hand side) multi type was in Inert—see Lemma 9.1.3 (Typing properties of useful inert terms) and Lemma 9.1.6 (Typing properties of useful inert programs).

Lemma 13.8.2 (Spreading of co-shrinkingness)**.**
Let r be a rigid term and $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} r : M$ be a type derivation for it. If Γ is co-shrinking (resp. unitary co-shrinking), then M is a co-shrinking (resp. unitary co-shrinking) multi type.

Proof.
By induction on the definition of the s term r. Cases:

- *Variable*: Let $r := x$. Then necessarily, for some $n \in \mathbb{N}$, Φ is of the following form

$$\frac{\left(x : [L_i] \vdash^{(0,1)} x : L_i\right)_{i=1}^{n}}{x : [L_i]_{i=1}^{n} \vdash^{(0,n)} x : [L_i]_{i=1}^{n}}\ \mathsf{many}_{\mathsf{VAR}}$$

  where $M = [L_i]_{i=1}^{n}$ and $\Gamma = x : M$. Note that since Γ is a co-shrinking (resp. unitary co-shrinking) context, then M is a co-shrinking (resp. unitary co-shrinking) multi type.

- *Application*: Let $r \coloneqq r't$ where r' is a rigid term. Then Φ must be of the following form:

$$\frac{\Psi \rhd_{\mathrm{S}} \Pi \vdash^{(m',s')} r' : [Q \multimap P] \qquad \Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m'',s'')} t : Q}{\Pi \uplus \Delta \vdash^{(m'+m''+1,s'+s''+1)} r't : P}\ \mathsf{app}$$

  where $\Gamma = \Pi \uplus \Delta$, with Π and Δ co-shrinking (resp. unitary co-shrinking) type contexts. By *i.h.*, $[Q \multimap P]$ is a co-shrinking (resp. unitary co-shrinking) multi type, in turn implying that P is a co-shrinking (resp. unitary co-shrinking) multi type.
- *Explicit substitution*: Let $r \coloneqq r'[x \leftarrow r'']$ where r' and r'' are rigid terms. Then Φ must be of the following form:

$$\frac{\Psi \rhd_{\mathrm{S}} \Pi; x : Q \vdash^{(m',s')} r' : P \qquad \Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m'',s'')} r'' : Q}{\Pi \uplus \Delta \vdash^{(m'+m'',s'+s''+1)} r'r'' : P}\ \mathrm{ES}$$

  where $\Gamma = \Pi \uplus \Delta$, with Π and Δ co-shrinking (resp. unitary co-shrinking) type contexts. By application of the *i.h.* on Θ—since r'' is a rigid term—Q is a co-shrinking (resp. unitary co-shrinking) multi type, and hence $\Pi, x : Q$ is a co-shrinking (resp. unitary co-shrinking) context. By *i.h.* applied to Ψ (as r' is a rigid term), P is a co-shrinking (resp. unitary co-shrinking) multi type.

□

In addition, the following are used repeatedly:

Lemma 13.8.3 (Typing properties of theoretical values)**.**
*Let $t \in (\mathsf{Var} \cup \mathsf{Val})$—*i.e.*, t is a theoretical value—and let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ be a type derivation for it. If $M = \mathbf{0}$, then $\Gamma = \emptyset$ and $(m, s) = (0, 0)$.*

Proof. Trivial.

□

Finally, we use Lemma 13.8.2 (Spreading of co-shrinkingness) to prove the following:

Proposition 13.8.4 (Typing properties of Strong CbV-normal forms)**.**
Let f_s be a strong fireball and let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} f_s : M$ be a type derivation for it, with Γ a co-shrinking (resp. unitary co-shrinking) type context and such that if f_s is an answer then M is shrinking (resp. unitary shrinking). Then $|f_s|_{\mathrm{S}} \leq m$ (resp. $|f_s|_{\mathrm{S}} = m$).

Proof. (Click here to go back to main chapter.)
By structural induction on f_s. Cases:

- *Variable*: Let $f_s \coloneqq x$. Then Φ must be of the following form:

$$\frac{\left(x : [L_i] \vdash^{(0,1)} x : L_i\right)_{i=1}^{n}}{x : [L_i]_{i=1}^{n} \vdash^{(0,n)} x : [L_i]_{i=1}^{n}}\ \mathsf{many}_{\mathsf{VAR}}$$

  where $M = [L_i]_{i=1}^{n}$ and $\Gamma = x : M$. Then, $|f_s|_{\mathrm{S}} = 0 = m$.
- *Application*: Let $f_s \coloneqq i_s g_s$. Then Φ must be of the following form

$$\frac{\Psi \rhd_{\mathrm{S}} \Pi \vdash^{(m',s')} i_s : [N \multimap M] \qquad \Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m'',s'')} g_s : N}{\Pi \uplus \Delta \vdash^{(m'+m''+1,s'+s''+1)} i_s g_s : M}\ \mathsf{app}$$

where $\Gamma = \Pi \uplus \Delta$ and $(m, s) = (m' + m'' + 1, s' + s'' + 1)$. By hypothesis, Γ is co-shrinking (resp. unitary co-shrinking), and then so are Π and Δ.
Now, since i_s is a rigid term, we can apply Lemma 13.8.2 on Ψ and so $[Q \multimap P]$ is co-shrinking (resp. unitary co-shrinking), which entails that Q is shrinking (resp. unitary shrinking).
Note that i_s is not a value up to ESs, so we can apply *i.h.* on both premises, obtaining that $m' \leq |i_s|_{\mathrm{S}}$ and $m'' \leq |g_s|_{\mathrm{S}}$ (resp. $m' = |i_s|_{\mathrm{S}}$ and $m'' = |g_s|_{\mathrm{S}}$). Hence, $|f_s|_{\mathrm{S}} = |i_s|_{\mathrm{S}} + |g_s|_{\mathrm{S}} + 1 \leq m' + m'' + 1 = m$ (resp. $|f_s|_{\mathrm{S}} = |i_s|_{\mathrm{S}} + |g_s|_{\mathrm{S}} + 1 = m' + m'' + 1 = m$).

- *Explicit substitution on inert*: Let $f_s \coloneqq i_s[x \leftarrow j_s]$. Then Φ must be of the following form:

$$\frac{\Psi \rhd_{\mathrm{S}} \Pi; x : N \vdash^{(m',s')} i_s : M \quad \Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m'',s'')} j_s : N}{\Pi \uplus \Delta \vdash^{(m'+m'',s'+s''+1)} i_s[x \leftarrow j_s] : M}\ \mathrm{ES}$$

where $\Gamma = \Pi \uplus \Delta$ and $(m, s) = (m' + m'', s' + s'' + 1)$. Note that since Γ is co-shrinking (resp. unitary co-shrinking), and then so are Π and Δ.
Now, since j_s is a rigid term, we can apply Lemma 13.8.2 on Θ, obtaining that Q is co-shrinking (resp. unitary co-shrinking), which in turn entails that $\Pi; x : Q$ is co-shrinking (resp. unitary co-shrinking).
Next, note that neither i_s nor j_s are abstractions up to ES. We can then apply *i.h.* on both premises, obtaining that $m' \leq |i_s|_{\mathrm{S}}$ and $m'' \leq |j_s|_{\mathrm{S}}$ (resp. $m' = |i_s|_{\mathrm{S}}$ and $m'' = |j_s|_{\mathrm{S}}$). Hence, $|f_s|_{\mathrm{S}} = |i_s|_{\mathrm{S}} + |j_s|_{\mathrm{S}} \leq m' + m'' = m$ (resp. $|f_s|_{\mathrm{S}} = |i_s|_{\mathrm{S}} + |j_s|_{\mathrm{S}} = m' + m'' = m$).

- *Explicit substitution on fireball*: Let $f_s \coloneqq g_s[x \leftarrow i_s]$. Then Φ must be of the following form:

$$\frac{\Psi \rhd_{\mathrm{S}} \Pi; x : N \vdash^{(m',s')} g_s : M \qquad \Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m'',s'')} i_s : N}{\Pi \uplus \Delta \vdash^{(m'+m'',s'+s''+1)} g_s[x \leftarrow i_s] : M}\ \mathrm{ES}$$

where where $\Gamma = \Pi \uplus \Delta$ and $(m, s) = (m' + m'', s' + s'' + 1)$. Note that since Γ is co-shrinking (resp. unitary co-shrinking), and then so are Π and Δ.
Now, since i_s is a rigid term, we can apply Lemma 13.8.2 on Θ to obtain that Q is co-shrinking (resp. unitary co-shrinking), which in turn entails that $\Pi; x : Q$ is co-shrinking (resp. unitary co-shrinking).
Next, note that i_s is not a value up to ESs, while g_s is a value up to ESs if and only if so is $f_s = g_s[x \leftarrow i]$, hence if g_s is a value up to ESs then P is shrinking (resp. unitary shrinking). We can then apply *i.h.* to both premises: $m' \leq |g_s|_{\mathrm{S}}$ and $m'' \leq |i_s|_{\mathrm{S}}$ (resp. $m' = |g_s|_{\mathrm{S}}$ and $m'' = |i_s|_{\mathrm{S}}$). Therefore, $|f_s|_{\mathrm{S}} = |g_s|_{\mathrm{S}} + |i_s|_{\mathrm{S}} \leq m' + m'' = m$ (resp. $|f_s|_{\mathrm{S}} = |g_s|_{\mathrm{S}} + |i_s|_{\mathrm{S}} = m' + m'' = m$).

- *Abstraction*: Let $f_s \coloneqq \lambda x.g_s$. Then, Φ must be of the following form:

$$\frac{\left(\dfrac{\Psi_i \rhd_{\mathrm{S}} \Gamma_i; x : O_i \vdash^{(m_i,s_i)} g_s : N_i}{\Gamma_i \vdash^{(m_i+1,s_i+1)} \lambda x.g_s : O_i \multimap N_i}\ \mathsf{fun}\right)_{i=1}^{n}}{\biguplus_{i=1}^{n} \Gamma_i \vdash^{\left((\sum_{i=1}^{n} m_i)+n,(\sum_{i=1}^{n} s_i)+n\right)} \lambda x.g_s : [O_i \multimap N_i]_{i=1}^{n}}\ \mathsf{many}_{\lambda}$$

where $\Gamma = \biguplus_{i=1}^{n} \Gamma_i$, $(m, s) = ((\sum_{i=1}^{n} m_i) + n, (\sum_{i=1}^{n} s_i) + n)$ and $M = [O_i \multimap N_i]_{i=1}^{n}$.
First, note that since M is shrinking (resp. unitary shrinking) by hypothesis, then it must be that $n \geq 1$ (resp. $n = 1$), that O is co-shrinking (resp. unitary co-shrinking) and that N is shrinking (resp. unitary shrinking). In turn, the latter entails that $\Gamma; x : R$ is a co-shrinking (resp. unitary co-shrinking) type context. We can then apply the *i.h.* on Ψ_i for all $1 \leq i \leq n$,

obtaining that $|g_s|_{\mathrm{S}} \leq m_i$ (resp. $|g_s|_{\mathrm{S}} = m_i$). Therefore, $|f_s|_{\mathrm{S}} = |g_s|_{\mathrm{S}} + 1 \leq n(|g_s|_{\mathrm{S}} + 1) \leq (\sum_{i=1}^{n} m_i) + n = m$, where the first inequality holds because $n \geq 1$ (resp. $|f_s|_{\mathrm{S}} = |g_s|_{\mathrm{S}} + 1 = m_1 + 1 = m$, all of which holds because $n = 1$).

□

(Click here to go back to main chapter.)

Substitution for Strong CbV. The Strong CbV not performing substitutions *one at a time*—like all the other cases in this work—the version of the Substitution property that we present here for Strong CbV is of a considerably simpler nature than, for example, the Open CbNeed and Useful Open CbNeed cases:

Lemma 13.8.5 (Substitution for Strong CbV)**.**
Let v_{T} be a theoretical value and let

$$\Phi_t \rhd_{\mathrm{S}} \Gamma; x : N \vdash^{(m,s)} t : M$$
$$\Psi \rhd_{\mathrm{S}} \Pi \vdash^{(m',s')} v_{\mathrm{T}} : N$$

Then there exists type derivation $\Phi_{t\{x\leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} \Gamma \biguplus \Pi \vdash^{(m'',s'')} t\{x\leftarrow v_{\mathrm{T}}\} : M$ such that $m'' = m + m'$ and $s'' \leq s + s'$.

Proof. (Click here to go back to main chapter.)
By structural induction on the t. Cases:

- *Variable.* Let $t \in \mathsf{Var}$. There are two sub-cases:
  1. Let $t \coloneqq x$. That is, $t\{x\leftarrow v\} = x\{x\leftarrow v\} = v$. Hence, Φ_t must be of the following form:

$$\dfrac{\left(\dfrac{}{x : L_i \vdash^{(0,1)} x : L_i}\ \mathsf{ax}\right)_{i=1}^{n}}{x : [L_i]_{i=1}^{n} \vdash^{(0,n)} x : [L_i]_{i=1}^{n}}\ \mathsf{many_{VAR}}$$

  where $\Gamma = \emptyset$ and $M = [L_1, \ldots, L_n] = N$. The statement holds by taking $\Phi_{t\{x\leftarrow v\}} \coloneqq \Psi$, noting that $m'' = m' = m + m'$ and $s'' = s' \leq n + s' = s + s'$.
  2. Let $t \coloneqq z \neq x$. That is, $t\{x\leftarrow v_{\mathrm{T}}\} = z\{x\leftarrow v_{\mathrm{T}}\} = z$. Then Φ_t must be of the following form:

$$\dfrac{\left(\dfrac{}{z : L_i \vdash^{(0,1)} z : L_i}\ \mathsf{ax}\right)_{i=1}^{n}}{z : [L_i]_{i=1}^{n} \vdash^{(0,n)} z : [L_i]_{i=1}^{n}}\ \mathsf{many_{VAR}}$$

  where $M = [L_i]_{i=1}^{n}$, $\Gamma = \{z : [L_i]_{i=1}^{n}\}$. This means that $N = \mathbf{0}$, and so Ψ must be of the following form:

$$\dfrac{}{\emptyset \vdash^{(0,0)} v_{\mathrm{T}} : \mathbf{0}}\ \mathsf{many_{\lambda}}$$

  if $v_{\mathrm{T}} \in \mathsf{Val}$, or of the following form:

$$\dfrac{}{\emptyset \vdash^{(0,0)} v_{\mathrm{T}} : \mathbf{0}}\ \mathsf{many_{VAR}}$$

  if $v_{\mathrm{T}} \in \mathsf{Var}$. In either case, we have that $\Gamma \biguplus \Pi = \Gamma$ and $(m', s') = (0, 0)$.
  The statement then follows by taking $\Theta_{t\{x\leftarrow v_{\mathrm{T}}\}} \coloneqq \Phi_t$, noting that $m'' = m = m + m'$ and $s'' = s \leq s + 0 = s + s'$.

- *Application*: Let $t \coloneqq sm$. That is, $t\{x \leftarrow v_{\mathrm{T}}\} = s\{x \leftarrow v_{\mathrm{T}}\}m\{x \leftarrow v_{\mathrm{T}}\}$. Then Φ_t must be of the following form

$$\frac{\Phi_s \rhd_{\mathrm{S}} \Gamma_s; x : N_s \vdash^{(m_s,s_s)} s : [O \multimap M] \quad \Phi_m \rhd_{\mathrm{S}} \Gamma_m; x : N_m \vdash^{(m_m,s_m)} m : O}{\Gamma_s \biguplus \Gamma_m; x : N_s \uplus N_m \vdash^{(m_s+m_m+1,s_s+s_m+1)} sm : M}\ \mathsf{app}$$

where $\Gamma = \Gamma_s \biguplus \Gamma_m$, $N = N_s \uplus N_m$ and $(m, s) = (m_s + m_m + 1, s_s + s_m + 1)$.
By application of Lemma 13.8.6 (Splitting multi types of Strong CbV type derivations) with respect to Ψ and the splitting $N = N_s \uplus N_m$, there exist type contexts Π_1 and Π_2 and type derivations

$$\Psi_s \rhd_{\mathrm{S}} \Pi_1 \vdash^{(m'_1,s'_1)} v_{\mathrm{T}} : N_s$$
$$\Psi_m \rhd_{\mathrm{S}} \Pi_2 \vdash^{(m'_2,s'_2)} v_{\mathrm{T}} : N_m$$

where $\Pi = \Pi_1 \biguplus \Pi_2$, $(m', s') = (m'_1 + m'_2, s'_1 + s'_2)$.
Then, we may apply the *i.h.* on Φ_s and Ψ_s to obtain type derivation

$$\Phi_{s\{x \leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} \Gamma_s \biguplus \Pi_1 \vdash^{(m''_1,s''_1)} s\{x \leftarrow v_{\mathrm{T}}\} : [O \multimap M]$$

such that $m''_1 = m_s + m'_1$ and $s''_1 \leq s_s + s'_1$. Moreover, we may apply the *i.h.* on Φ_m and Φ_m to obtain type derivation

$$\Phi_{m\{x \leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} \Gamma_m \biguplus \Pi_2 \vdash^{(m''_2,s''_2)} m\{x \leftarrow v_{\mathrm{T}}\} : O$$

such that $m''_2 = m_m + m'_2$ and $s''_2 \leq s_m + s'_2$.
Finally, and since

$$\Gamma \biguplus \Pi = (\Gamma_s \biguplus \Gamma_m) \biguplus (\Pi_1 \biguplus \Pi_2) = (\Gamma_s \biguplus \Pi_1) \biguplus (\Gamma_m \biguplus \Pi_2)$$

we may derive $\Phi_{t\{x \leftarrow v_{\mathrm{T}}\}}$ as follows

$$\frac{\Phi_{s\{x \leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} \Gamma_s \biguplus \Pi_1 \vdash^{(m''_1,s''_1)} s\{x \leftarrow v_{\mathrm{T}}\} : [O \multimap M] \quad \Phi_{m\{x \leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} \Gamma_m \biguplus \Pi_2 \vdash^{(m''_2,s''_2)} m\{x \leftarrow v_{\mathrm{T}}\} : O}{\Gamma_s \biguplus \Pi_1 \biguplus \Gamma_m \biguplus \Pi_2 \vdash^{(m''_1+m''_2+1,s''_1+s''_2+1)} s\{x \leftarrow v_{\mathrm{T}}\} : M}\ \mathsf{app}$$

noting in particular that

$$m''_1 + m''_2 + 1 = m_s + m'_1 + m_m + m'_2 + 1 = m + m'$$
$$s''_1 + s''_2 \leq s_s + s'_1 + s_m + s'_2 = s + s'$$

- *Abstraction*: Let $t \coloneqq \lambda y.s$. Note that we may safely assume that $y \notin (\mathsf{fv}(v_{\mathrm{T}}) \cup \{x\})$—by α-equivalence. Hence, $t\{x \leftarrow v_{\mathrm{T}}\} = \lambda y.s\{x \leftarrow v_{\mathrm{T}}\}$ and Φ must be of the following form:

$$\frac{\left(\dfrac{\Phi_i \rhd_{\mathrm{S}} \Gamma_i; x : N_i; y : O_i \vdash^{(m_i,s_i)} s : P_i}{\Gamma_i; x : \uplus_{i=1}^n N_i \vdash^{(m_i+1,s_i+1)} \lambda y.s : O_i \multimap P_i}\ \mathsf{fun}\right)_{i=1}^n}{(\biguplus_{i=1}^n \Gamma_i)\,; x : (\uplus_{i=1}^n N_i) \vdash^{((\sum_{i=1}^n m_i)+n,(\sum_{i=1}^n s_i)+n)} \lambda y.s : [O_i \multimap P_i]_{i=1}^n}\ \mathsf{many}_\lambda$$

where $\Gamma = \biguplus_{i=1}^n \Gamma_i$, $N = \uplus_{i=1}^n N_i$, $M = [O_i \multimap P_i]_{i=1}^n$ and $(m, s) = ((\sum_{i=1}^n m_i) + n, (\sum_{i=1}^n s_i) + n)$.
By Lemma 11.2.1 (Relevance of the Strong CbV type system), we have that $y \notin \mathsf{dom}(\Pi)$, and so $\Psi \rhd_{\mathrm{S}} \Pi; y : \mathbf{0} \vdash^{(m',s')} v_{\mathrm{T}} : \uplus_{i=1}^n N_i$. Case analysis on whether $n > 0$:

– Let $n \coloneqq 0$. Then, if $v_{\mathrm{T}} \in \mathsf{Val}$ then Ψ must be of the following form:

$$\frac{}{\emptyset \vdash^{(0,0)} v_{\mathrm{T}} : \mathbf{0}}\ \mathsf{many}_{\lambda}$$

and if $v_{\mathrm{T}} \in \mathsf{Var}$ instead, then Ψ must be of the following form:

$$\frac{}{\emptyset \vdash^{(0,0)} v_{\mathrm{T}} : \mathbf{0}}\ \mathsf{many}_{\mathsf{VAR}}$$

In either case, it is easy to see that the statement holds by taking $\Phi_{t\{x \leftarrow v_{\mathrm{T}}\}}$ as follows:

$$\frac{}{\emptyset \vdash^{(0,0)} \lambda y.z\{x \leftarrow v_{\mathrm{T}}\} : \mathbf{0}}\ \mathsf{many}_{\lambda}$$

– Let $n > 0$. We may repeatedly apply Lemma 13.8.6 (Splitting multi types of type derivations) to obtain the following type derivation for all $1 \leq i \leq n$:

$$\Phi_i \rhd_{\mathrm{S}} \Pi_i ; y : \mathbf{0} \vdash^{(m'_i, s'_i)} v_{\mathrm{T}} : N_i$$

such that $\Pi = \biguplus_{i=1}^{n} \Pi_i$ and $(m', s') = (\sum_{i=1}^{n} m'_i, \sum_{i=1}^{n} s'_i)$.
Next, for all $1 \leq i \leq n$, we may apply the *i.h.* with respect to Φ_i and Ψ_i to obtain type derivation

$$\Phi'_i \rhd_{\mathrm{S}} \left(\Gamma_i \biguplus \Pi_i\right) ; y : O_i \vdash^{(m''_i, s''_i)} s\{x \leftarrow v_{\mathrm{T}}\} : P_i$$

such that $m''_i = m_i + m'_i$ and $s''_i \leq s_i + s'_i$. We may finally derive $\Phi_{t\{x \leftarrow v_{\mathrm{T}}\}}$ as follows:

$$\frac{\left(\dfrac{\Phi'_i \rhd_{\mathrm{S}} (\Gamma_i \biguplus \Pi_i) ; y : O_i \vdash^{(m''_i, s''_i)} s\{x \leftarrow v_{\mathrm{T}}\} : P_i}{(\Gamma_i \biguplus \Pi_i) \vdash^{(m''_i+1, s''_i+1)} \lambda y.(s\{x \leftarrow v_{\mathrm{T}}\}) : O_i \multimap P_i}\ \mathsf{fun}\right)_{i=1}^{n}}{\biguplus_{i=1}^{n} (\Gamma_i \biguplus \Pi_i) \vdash^{(\left(\sum_{i=1}^{n} m''_i\right)+n, \left(\sum_{i=1}^{n} s''_i\right)+n)} \lambda y.(s\{x \leftarrow v_{\mathrm{T}}\}) : [O_i \multimap P_i]_{i=1}^{n}}\ \mathsf{many}_{\lambda}$$

- *Explicit substitution*: Let $t \coloneqq s[y \leftarrow m]$. We may safely assume that $y \notin (\mathsf{fv}(v_{\mathrm{T}}) \cup \{x\})$—by α-equivalence—which entails that $t\{x \leftarrow v_{\mathrm{T}}\} = (s\{x \leftarrow v_{\mathrm{T}}\})[y \leftarrow m\{x \leftarrow v_{\mathrm{T}}\}]$. Then Φ must be of the following form:

$$\frac{\Phi_s \rhd_{\mathrm{S}} \Gamma_s ; x : N_s ; y : O \vdash^{(m_s, s_s)} s : M \quad \Phi_m \rhd_{\mathrm{S}} \Gamma_m ; x : N_m ; y : \mathbf{0} \vdash^{(m_m, s_m)} m : O}{(\Gamma_s \biguplus \Gamma_m) ; x : (N_s \uplus N_m) \vdash^{(m_s + m_m + 1, s_s + s_m + 1)} s[y \leftarrow m] : M}\ \mathrm{ES}$$

where $\Gamma = \Gamma_s \biguplus \Gamma_m$, $N = N_s \uplus N_m$ and $(m, s) = (m_s + m_m + 1, s_s + s_m + 1)$.
By Lemma 13.8.6 (Splitting multi types of Strong CbV type derivations), from the splitting $N = N_s \uplus N_m$ we may infer the existence of the following type derivations:

$$\Psi_s \rhd_{\mathrm{S}} \Pi_s \vdash^{(m'_s, s_s)} v_{\mathrm{T}} : N_s$$
$$\Psi_m \rhd_{\mathrm{S}} \Pi_m \vdash^{(m'_m, s_m)} v_{\mathrm{T}} : N_m$$

Thus, we may apply the *i.h.* on Φ_s and Ψ_s to get the following type derivation:

$$\Phi_{s\{x \leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} \left(\Gamma_s \biguplus \Pi_s\right) ; y : O \vdash^{(m''_s, s''_s)} s\{x \leftarrow v_{\mathrm{T}}\} : M$$

where $m''_s = m_s + m'_s$ and $s''_s \leq s_s + s_s$.

Similary, an application of the *i.h.* on Φ_m and Ψ_m gives the following type derivation:

$$\Phi_{m\{x\leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} \Gamma_m \biguplus \Pi_m \vdash^{(m''_m, s''_m)} m\{x\leftarrow v_{\mathrm{T}}\} : O$$

where $m''_m = m_m + m'_m$ and $s''_m \leq s_m + s_m$.
Therefore, and since

$$\Gamma \biguplus \Pi = (\Gamma_s \biguplus \Gamma_m) \biguplus (\Pi_s \biguplus \Pi_m) = (\Gamma_s \biguplus \Pi_s) \biguplus (\Gamma_m \biguplus \Pi_m)$$

the statement holds by deriving $\Phi_{t\{x\leftarrow v_{\mathrm{T}}\}}$ as follows

$$\frac{\Phi_{s\{x\leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} (\Gamma_s \biguplus \Pi_s)\,;\, y : O \vdash^{(m''_s, s''_s)} s\{x\leftarrow v_{\mathrm{T}}\} : M \quad \Phi_{m\{x\leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} \Gamma_m \biguplus \Pi_m \vdash^{(m''_m, s''_m)} m\{x\leftarrow v_{\mathrm{T}}\} : O}{\Gamma_s \biguplus \Pi_s \biguplus \Gamma_m \biguplus \Pi_m \vdash^{(m''_s + m''_m + 1, s''_s + s''_m + 1)} s\{x\leftarrow v_{\mathrm{T}}\}[x\leftarrow m\{x\leftarrow v_{\mathrm{T}}\}] : M}\ \mathrm{ES}$$

□

(Click here to go back to main chapter.)

Subject Reduction for Strong CbV. This is obtained by first studying the weak case—*i.e.*, the Open evaluation strategy $\rightarrow_{\mathsf{w}}$—and then building on it to obtain the result for the strong case.

Fistly, the following is required to apply Lemma 11.2.3 (Substitution for Strong CbV) in the proofs of Lemma 11.2.4.2 (Open Quantitative Subject Reduction for Strong CbV - exponential case) and Proposition 11.2.5.2 (Shrinking Quantitative Subject Reduction for Strong CbV - exponential case) to obtain the right indices.

Lemma 13.8.6 (Splitting multi types of Strong CbV type derivations)**.**
*Let $t \in (\mathsf{Var} \cup \mathsf{Val})$—*i.e.*, t is a theoretical value—and let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : N \biguplus O$ be a type derivation. Then there exist type derivations*

$$\Psi \rhd_{\mathrm{S}} \Pi \vdash^{(m',s')} t : N$$
$$\Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m'',s'')} t : O$$

such that $\Gamma = \Pi \biguplus \Delta$ and $(m, s) = (m' + m'', s' + s'')$.

Proof.
Trivial, considering how the only typing rules deriving a multi type for theoretical values—*i.e.*, for variables and values—are $\mathsf{many}_{\mathsf{VAR}}$ and many_{λ}, whose premises can be split at will.

□

Lemma 13.8.7 (Open Quantitative Subject Reduction for Strong CbV)**.**
Let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ be a type derivation.

1. Multiplicative*: If $t \rightarrow_{\mathsf{wm}} u$, then there exists a derivation $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m-2, s-1)} u : M$.*
2. Exponential*: If $t \rightarrow_{\mathsf{we}} u$, then there exists a derivation $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} u : M$ such that $m' = m$ and $s' < s$.*

Proof. (Click here to go back to main chapter.)

1. *Multiplicative steps*: By induction on the open evaluation context O such that $t = O\langle s\rangle \rightarrow_{\mathsf{wm}} O\langle s'\rangle = u$, with $s \mapsto_{\mathsf{wm}} s'$:

- *Context hole*: Let $O \coloneqq \langle\cdot\rangle$. Then $t = s = S\langle\lambda x.m\rangle\tilde{t}$ and $u = S\langle m[x{\leftarrow}\tilde{t}]\rangle$. We proceed by induction on the length of S:
  - *Base case*: Let $S \coloneqq \langle\cdot\rangle$. Then Φ is of the following form:

$$\dfrac{\dfrac{\dfrac{\Phi_m \rhd_{\mathrm{S}} \Gamma_m; x : N \vdash^{(m_m,s_m)} m : M}{\Gamma_m \vdash^{(m_m+1,s_m+1)} \lambda x.m : N \multimap M}\,\mathsf{fun}}{\Gamma_m \vdash^{(m_m+1,s_m+1)} \lambda x.m : [N \multimap M]}\,\mathsf{many} \qquad \Phi_{\tilde{t}} \rhd_{\mathrm{S}} \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}},s_{\tilde{t}})} \tilde{t} : N}{\Gamma_m \uplus \Gamma_{\tilde{t}} \vdash^{(m_m+m_{\tilde{t}}+2,s_m+s_{\tilde{t}}+2)} (\lambda x.m)\tilde{t} : M}\,\mathsf{app}$$

    The statement holds by deriving Ψ as follows

$$\dfrac{\Phi_m \rhd_{\mathrm{S}} \Gamma_m; x : N \vdash^{(m_m,s_m)} m : M \quad \Phi_{\tilde{t}} \rhd_{\mathrm{S}} \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}},s_{\tilde{t}})} \tilde{t} : N}{\Gamma_s \uplus \Gamma_m \vdash^{(m_m+m_{\tilde{t}},s_m+s_{\tilde{t}}+1)} : M}\,\mathrm{ES}$$

    noting in particular that $(m_m + m_{\tilde{t}}, s_m + s_{\tilde{t}} + 1) = (m - 2, s - 1)$.
  - *Inductive case*: Let $S \coloneqq S'[y{\leftarrow}\tilde{u}]$. Then Φ is of the following form

$$\dfrac{\Phi_{S\langle\lambda x.m\rangle} \rhd\ \Gamma_1 \vdash^{(m',s')} S\langle\lambda x.m\rangle : [N \multimap M] \quad \Phi_{\tilde{t}} \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m'',s'')} \tilde{t} : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m'+m''+1,s'+s''+1)} (S\langle\lambda x.m\rangle)\tilde{t} : M}\,\mathsf{app}$$

    where, moreover, $\Phi_{S\langle\lambda x.m\rangle}$ is of the following form:

$$\dfrac{\Phi_1 \rhd_{\mathrm{S}} \Pi_1; y : O \vdash^{(m'_1,s'_1)} S'\langle\lambda x.m\rangle : [N \multimap M] \quad \Phi_{\tilde{u}} \rhd_{\mathrm{S}} \Pi_2 \vdash^{(m'_2,s'_2)} \tilde{u} : O}{\Pi_1 \uplus \Pi_2 \vdash^{(m'_1+m'_2+1,s'_1+s'_2+1)} S\langle\lambda x.m\rangle : [N \multimap M]}\,\mathrm{ES}$$

    with $\Gamma_1 = \Pi_1 \uplus \Pi_2$, $(m', s') = (m'_1 + m'_2 + 1, s'_1 + s'_2 + 1)$.
    In order to apply the *i.h.*, let us derive an auxiliary type derivation Φ_{ind} as follows:

$$\dfrac{\Phi_1 \rhd_{\mathrm{S}} \Pi_1; y : O \vdash^{(m'_1,s'_1)} S'\langle\lambda x.m\rangle : [N \multimap M] \quad \Phi_{\tilde{t}} \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m'',s'')} \tilde{t} : N}{(\Pi_1 \uplus \Gamma_2); y : O \vdash^{(m'_1+m''+1,s'_1+s''+1)} (S'\langle\lambda x.m\rangle)\tilde{t} : M}\,\mathsf{app}$$

    Note that we can now apply the *i.h.* on Φ_{ind} and obtain type derivation

$$\Psi_{\mathrm{ind}} \rhd_{\mathrm{S}} \left(\Pi_1 \uplus \Gamma_2\right); y : O \vdash^{(m'_1+m''-1,s'_1+s'')} S'\langle m[x{\leftarrow}\tilde{t}]\rangle : M$$

    We can finally derive $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m-2,s-1)} u : M$ as follows

$$\dfrac{\Psi_{\mathrm{ind}} \rhd_{\mathrm{S}} (\Pi_1 \uplus \Gamma_2); y : O \vdash^{(m'_1+m''-1,s'_1+s'')} S'\langle m[x{\leftarrow}\tilde{t}]\rangle : M \quad \Phi_{\tilde{u}} \rhd_{\mathrm{S}} \Pi_2 \vdash^{(m'_2,s'_2)} \tilde{u} : O}{\Pi_1 \uplus \Gamma_2 \uplus \Pi_2 \vdash^{(m'_1+m''+m'_2,s'_1+s''+s'_2+1)} S\langle m[x{\leftarrow}\tilde{t}]\rangle : M}\,\mathsf{app}$$

    Note that $\Gamma = \Gamma_1 \uplus \Gamma_2 = (\Pi_1 \uplus \Pi_2) \uplus \Gamma_2$ and that

$$(m'_1 + m'' + m'_2, s'_1 + s'' + s'_2 + 1) = ((m' - 1) + m'', s' + s'') = (m - 2, s - 1)$$

- *Left of an application*: Let $O \coloneqq O'm$. Then Φ is of the following form:

$$\dfrac{\Phi_{O'\langle s\rangle} \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_1,s_1)} O'\langle s\rangle : [N \multimap M] \quad \Phi_m \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} m : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2+1,s_1+s_2+1)} O'\langle s\rangle m : M}\,\mathsf{app}$$

  The statement follows easily by application of the *i.h.* on $\Phi_{O'\langle s\rangle}$ and then deriving Ψ by combining the result of the *i.h.* with Φ_m.

- *Right of an application*: Analogous to the previous case.
- *Left of an ES*: Let $O \coloneqq O'[x{\leftarrow}m]$. Then Φ is of the following form:

$$\frac{\Phi_{O'\langle s\rangle} \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} O'\langle s\rangle : M \quad \Phi_m \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} m : N}{\Gamma_1 \biguplus \Gamma_2 \vdash^{(m_1+m_2,s_1+s_2+1)} O'\langle s\rangle[x{\leftarrow}m] : M}\ \mathrm{ES}$$

  The statement follows easily by application of the *i.h.* on $\Phi_{O'\langle s\rangle}$ and then deriving Ψ by combining the result of the *i.h.* with Φ_m.
- *Inside of an ES*: Analogous to the previous case.

2. *Exponential steps*: By induction on the open evaluation context O such that $t = O\langle u\rangle \to_{\mathsf{we}} O\langle u'\rangle = s$, with $u = u''[x{\leftarrow}S\langle v_{\mathrm{T}}\rangle] \mapsto_{\mathsf{we}} S\langle u''\{x{\leftarrow}v_{\mathrm{T}}\}\rangle = u'$:
   - *Context hole*: Let $O \coloneqq \langle\cdot\rangle$. Then $t = u''[x{\leftarrow}S\langle v_{\mathrm{T}}\rangle] \mapsto_{\mathsf{we}} S\langle u''\{x{\leftarrow}v_{\mathrm{T}}\}\rangle = s$. We proceed by induction on the length of S:
     - *Base case*: Let $S \coloneqq \langle\cdot\rangle$. Then Φ is of the following form

$$\frac{\Phi_{u''} \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} u'' : M \quad \Phi_{v_{\mathrm{T}}} \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} v_{\mathrm{T}} : N}{\Gamma_1 \biguplus \Gamma_2 \vdash^{(m_1+m_2,s_1+s_2+1)} u''[x{\leftarrow}v_{\mathrm{T}}] : M}\ \mathrm{ES}$$

       where $\Gamma = \Gamma_1 \biguplus \Gamma_2$ and $(m, s) = (m_1 + m_2, s_1 + s_2 + 1)$. We can apply Lemma 11.2.3 (Substitution for Strong CbV) on $\Phi_{u''}$ and $\Phi_{v_{\mathrm{T}}}$ to obtain type derivation

$$\Phi_{u''\{x\leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} \Gamma_1 \biguplus \Gamma_2 \vdash^{(m'',s'')} u''\{x{\leftarrow}v_{\mathrm{T}}\} : M$$

       where $m'' = m_1 + m_2$ and $s'' \leq s_1 + s_2$. The statement holds by taking $\Psi \coloneqq \Phi_{u''\{x\leftarrow v_{\mathrm{T}}\}}$, noting in particular that $m' = m_1+m_2 = m$ and $s' \leq s_1+s_2 < s_1+s_2+1$.
     - *Inductive case*: Let $S \coloneqq S'[y{\leftarrow}m]$. Then Φ must be of the following form

$$\frac{\Phi_{u''} \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_{u''},s_{u''})} u'' : M \quad \Phi_{S\langle v_{\mathrm{T}}\rangle} \rhd_{\mathrm{S}} \Delta \vdash^{(m'',s'')} S\langle v_{\mathrm{T}}\rangle : N}{\Gamma_1 \biguplus \Gamma_2 \vdash^{(m,s)} u''[x{\leftarrow}S\langle v_{\mathrm{T}}\rangle] : M}\ \mathrm{ES}$$

       where, moreover, $\Phi_{S\langle v_{\mathrm{T}}\rangle} \rhd_{\mathrm{S}} \Delta \vdash^{(m'',s'')} S\langle v_{\mathrm{T}}\rangle : N$ takes the following form

$$\frac{\Phi \rhd_{\mathrm{S}} \Pi_1; y : O \vdash^{(m_1,s_1)} S'\langle v_{\mathrm{T}}\rangle : N \quad \Phi \rhd_{\mathrm{S}} \Pi_2 \vdash^{(m_2,s_2)} m : O}{\Pi_1 \biguplus \Pi_2 \vdash^{(m_1+m_2,s_1+s_2+1)} S\langle v_{\mathrm{T}}\rangle : N}\ \mathrm{ES}$$

       with $\Delta = \Pi_1 \biguplus \Pi_2$ and $(m'', s'') = (m_1 + m_2, s_1 + s_2 + 1)$.
       In order to apply the *i.h.*, let us derive an auxiliary type derivation Φ_{ind} as follows:

$$\frac{\Phi_{u''} \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_{u''},s_{u''})} u'' : M \quad \Phi \rhd_{\mathrm{S}} \Pi_1; y : O \vdash^{(m_1,s_1)} S'\langle v_{\mathrm{T}}\rangle : N}{(\Gamma_1 \biguplus \Pi_1)\,; y : O \vdash^{(m_{u''}+m_1,s_{u''}+s_1+1)} u''[x{\leftarrow}S'\langle v_{\mathrm{T}}\rangle] : M}\ \mathrm{ES}$$

       Note that we can then apply the *i.h.* on $\Phi_{ind} \rhd_{\mathrm{S}} (\Gamma_1 \biguplus \Pi_1)\,; y : O \vdash^{(m_{u''}+m_1,s_{u''}+s_1+1)} u''[x{\leftarrow}S'\langle v_{\mathrm{T}}\rangle] : M$ to obtain type derivation

$$\Psi_{ind} \rhd_{\mathrm{S}} \left(\Gamma_1 \biguplus \Pi_1\right)\,; y : O \vdash^{(m_{u''}+m_1,s_{u''}+s_1)} S'\langle u''\{x{\leftarrow}v_{\mathrm{T}}\}\rangle : M$$

       Therefore, we may derive Ψ as follows

$$\frac{\Psi_{ind} \rhd_{\mathrm{S}} (\Gamma_1 \biguplus \Pi_1)\,; y : O \vdash^{(m_{u''}+m_1,s_{u''}+s_1)} S'\langle u''\{x{\leftarrow}v_{\mathrm{T}}\}\rangle : M \quad \Phi \rhd_{\mathrm{S}} \Pi_2 \vdash^{(m_2,s_2)} m : O}{\Gamma_1 \biguplus \Pi_1 \biguplus \Pi_2 \vdash^{(m_{u''}+m_1+m_2,s_{u''}+s_1+s_2+1)} S\langle u''\{x{\leftarrow}v_{\mathrm{T}}\}\rangle : M}\ \mathrm{ES}$$

- *Left of an application*: Let $O \coloneqq O'm$. Then Φ is of the following form:

$$\dfrac{\Phi_{O'\langle s\rangle} \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_1,s_1)} O'\langle s\rangle : [N \multimap M] \quad \Phi_m \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} m : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2+1,s_1+s_2+1)} O'\langle s\rangle m : M}\ \mathsf{app}$$

  The statement follows easily by application of the *i.h.* on $\Phi_{O'\langle s\rangle}$ and then deriving Ψ by combining the result of the *i.h.* with Φ_m.
- *Right of an application*: Analogous to the previous case.
- *Left of an ES*: Let $O \coloneqq O'[x\leftarrow u]$. Then Φ is of the following form:

$$\dfrac{\Phi_{O'\langle s\rangle} \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} O'\langle s\rangle : M \quad \Phi_m \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} m : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,s_1+s_2+1)} O'\langle s\rangle[x\leftarrow m] : M}\ \mathrm{ES}$$

  The statement follows easily by application of the *i.h.* on $\Phi_{O'\langle s\rangle}$ and then deriving Ψ by combining the result of the *i.h.* with Φ_m.
- *Inside of an ES*: Analogous to the previous case.

□

(Click here to go back to main chapter.)

We may now prove the following

Proposition 13.8.8 (Shrinking Quantitative Subject Reduction for Strong CbV)**.**
Let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ be a type derivation, with Γ a co-shrinking (resp. unitary co-shrinking) type context. Moreover, suppose that if t is an answer then M is shrinking (resp. unitary shrinking).

1. Multiplicative*: If $t \to_{\mathsf{sm}} t'$, then $m \geq 2$, $s \geq 1$, and there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} t' : M$ such that $m' \leq m-2$ and $s' < s$ (resp. $m' = m-2$ and $s' = s-1$).*
2. Exponential*: If $t \to_{\mathsf{se}} t'$, then $s \geq 1$ and there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} t' : M$ such that $m' = m$ and $s' < s$.*

Proof. (Click here to go back to main chapter.)
First, we prove the unitary shrinking version of the statement, under the hypothesis that Γ is unitary co-shrinking and if t is a value up to ESs then M is unitary shrinking. The proof is by induction on the evaluation strong context S such that $t = S\langle u\rangle \to_{\mathsf{s}} S\langle u'\rangle = t'$ with $u \to_{\mathsf{wm}} u'$ or $u' \to_{\mathsf{we}} u'$. Cases for S:

- *Hole*: Let $S \coloneqq \langle\cdot\rangle$ and $t \to_{\mathsf{w}_a} t'$ with $a \in \{\mathsf{m}, \mathsf{e}\}$.
  By Lemma 11.2.4 (Open Quantitative Subject Reduction for Strong CbV), we have that
  - if $t \to_{\mathsf{wm}} t'$ then there exists type derivation $\Phi' \rhd_{\mathrm{S}} \Gamma \vdash^{(m-2,s-1)} t' : M$;
  - if $t \to_{\mathsf{we}} t'$ then there is type derivation $\Phi' \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} t' : M$ such that $m' = m$ and $s' < s$.

  Note that, in this case, we do not use the hypothesis that Γ is a (unitary) co-shrinking context and that M is a (unitary) shrinking multi type if t is an abstraction up to ES.
- *Abstraction*: Let $S \coloneqq \lambda x.S'$. Then $t = S\langle u\rangle = \lambda x.S'\langle u\rangle \to_{\mathsf{s}_a} \lambda x.S'\langle u'\rangle = S\langle u'\rangle = t'$ with $u \to_{\mathsf{w}_a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Since t is an abstraction, M is a unitary shrinking multi type by hypothesis and hence it has the form $M = [O \multimap N]$ where O is unitary co-shrinking and N is unitary shrinking. Thus, the derivation Φ is necessarily

$$\dfrac{\dfrac{\Phi_1 \rhd_{\mathrm{S}} \Gamma; x : O \vdash^{(m_1,s_1)} S'\langle u\rangle : N}{\Gamma \vdash^{(m_1+1,s_1+1)} \lambda x.S'\langle u\rangle : O \multimap N}\ \mathsf{fun}}{\Gamma \vdash^{(m_1+1,s_1+1)} \lambda x.S'\langle u\rangle : [O \multimap N]}\ \mathsf{many}_{\lambda}$$

Since $\Gamma; x : O$ is a unitary co-shrinking type context and N_2 is a unitary shrinking multi type, application of the *i.h.* on Φ_1 yields type derivation $\Psi_1 \rhd_{\mathrm{S}} \Gamma; x : O \vdash^{(m_1', s_1')} S'\langle u \rangle : N$ such that

1. if $u \to_{\mathsf{wm}} u'$, then $m_1' = m_1 - 2$ and $s_1' = s_1 - 1$,
2. if $u \to_{\mathsf{we}} u'$, then $m_1' = m_1$ and $s_1' < s_1$.

We can then derive Ψ as follows

$$\dfrac{\dfrac{\Psi_1 \rhd_{\mathrm{S}} \Gamma; x : O \vdash^{(m_1', s_1')} S'\langle u \rangle : N}{\Gamma \vdash^{(m_1'+1, s_1'+1)} \lambda x.S'\langle u \rangle : O \multimap N}\ \mathsf{fun}}{\Gamma \vdash^{(m_1'+1, s_1'+1)} \lambda x.S'\langle u \rangle : [O \multimap N]}\ \mathsf{many}_{\lambda}$$

- *Explicit substitution of rigid context*: Let $S \coloneqq s[x \leftarrow R]$. Then, $t = S\langle u \rangle = s[x \leftarrow R\langle u \rangle] \to_{\mathsf{s}_a} s[x \leftarrow R\langle u' \rangle] = S\langle u' \rangle = t'$ with $u \to_{\mathsf{w}_a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Then Φ must be of the following form

$$\dfrac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1, s_1)} s : M \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2, s_2)} R\langle u \rangle : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2, s_1+s_2+1)} s[x \leftarrow R\langle u \rangle] : M}\ \mathrm{ES}$$

where $\Gamma = \Gamma_1 \uplus \Gamma_2$ is unitary co-shrinking—implying that so are Γ_1 and Γ_2—and $(m, s) = (m_1 + m_2, s_1 + s_2 + 1)$.
By *i.h.* on Φ_2, there exists $\Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2', s_2')} R\langle u' \rangle : N$ such that

1. if $u \to_{\mathsf{wm}} u'$, then $m_2' = m_2 - 2$ and $s_2' = s_2 - 1$.
2. if $u \to_{\mathsf{we}} u'$, then $m_2' = m_2$ and $s_2' < s_2$.

We may then derive Ψ as follows

$$\dfrac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1, s_1)} s : M \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2', s_2')} R\langle u \rangle : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2', s_1+s_2'+1)} s[x \leftarrow R\langle u' \rangle] : M}\ \mathrm{ES}$$

- *Strong context with explicit substitution of rigid term*: Let $S \coloneqq S'[x \leftarrow r]$. That is, $t = S\langle u \rangle = S'\langle u \rangle[x \leftarrow r] \to_{\mathsf{s}_a} S'\langle u' \rangle[x \leftarrow r] = S\langle u' \rangle = t'$ with $u \to_{\mathsf{w}_a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Then Φ must be of the following form

$$\dfrac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1, s_1)} S'\langle u \rangle : M \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2, s_2)} r : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2, s_1+s_2+1)} S'\langle u \rangle[x \leftarrow r] : M}\ \mathrm{ES}$$

where $\Gamma = \Gamma_1 \uplus \Gamma_2$ is unitary co-shrinking—implying that so are Γ_1 and Γ_2—and $(m, s) = (m_1 + m_2, s_1 + s_2 + 1)$.
First, note that since Γ_2 is unitary co-shrinking and r is a rigid term, then application of Lemma 13.8.2 (Spreading of co-shrinkingness) on Φ_2 gives that N is unitary co-shrinking. Hence, $\Gamma_1; x : N$ is unitary co-shrinking too. Moreover, note that $S'\langle u \rangle[x \leftarrow r]$ is an answer if and only if $S'\langle u \rangle$ is an answer.
We may then apply the *i.h.* on Φ_1, obtaining type derivation $\Psi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1', s_1')} S'\langle u' \rangle : M$ such that

1. if $u \to_{\mathsf{wm}} u'$, then $m_1' = m_1 - 2$ and $s_1' = s_1 - 1$.
2. if $u \to_{\mathsf{we}} u'$, then $m_1' = m_1$ and $s_1' < s_1$.

We may then derive Ψ as follows

$$\dfrac{\Psi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1', s_1')} S'\langle u' \rangle : M \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2, s_2)} r : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1'+m_2, s_1'+s_2+1)} S'\langle u' \rangle[x \leftarrow r] : M}\ \mathrm{ES}$$

- *Rigid term applied to strong context*: Let $S \coloneqq rS'$. Then, $t = S\langle u\rangle = rS'\langle u\rangle \to_{\mathsf{s}_a} rS'\langle u'\rangle = S\langle u'\rangle = t'$ with $u \to_{\mathsf{w}_a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Moreover, Φ must be of the following form

$$\frac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_1,s_1)} r : [N \multimap M] \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} S'\langle u\rangle : N}{\Gamma_1 \biguplus \Gamma_2 \vdash^{(m_1+m_2+1,s_1+s_2+1)} rS'\langle u\rangle : M}\ \mathsf{app}$$

  where $\Gamma = \Gamma_1 \biguplus \Gamma 2$and $(m, s) = (m_1 + m_2 + 1, s_1 + s_2 + 1)$.
  First, by Lemma 13.8.2 (Spreading of co-shrinkingness)—given that Γ_1 is a unitary co-shrinking type context and that r is a rigid term—we have that $[N \multimap M]$ is unitary co-shrinking. Hence, N is unitary shrinking.
  Next, since Γ_1 being unitary co-shrinking implies that both Γ_1 and Γ_2 are unitary co-shrinking, then we may apply the *i.h.* on Φ_2 to obtain type derivation $\Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2',s_2')} S'\langle u'\rangle : N$ such that
  1. if $u \to_{\mathsf{wm}} u'$, then $m_2' = m_2 - 2$ and $s_2' = s_2 - 1$.
  2. if $u \to_{\mathsf{we}} u'$, then $m_2' = m_2$ and $s_2' < s_2$.

  We may then derive Ψ as follows:

$$\frac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_1,s_1)} r : [N \multimap M] \quad \Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2',s_2')} S'\langle u'\rangle : N}{\Gamma_1 \biguplus \Gamma_2 \vdash^{(m_1+m_2+1,s_1+s_2+1)} rS'\langle u\rangle : M}\ \mathsf{app}$$

- *Rigid context applied to term*: Let $S \coloneqq Rs$. Then, $t = S\langle u\rangle = R\langle u\rangle s \to_{\mathsf{s}_a} R\langle u'\rangle s = S\langle u'\rangle = t'$ with $u \to_{\mathsf{w}_a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Moreover, Φ must be of the following form

$$\frac{\Phi_1 \rhd_{\mathrm{S}} [N \multimap M] \vdash^{(m_1,s_1)} R\langle u\rangle : \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} s : N}{\Gamma_1 \biguplus \Gamma_2 \vdash^{(m_1+m_2+1,s_1+s_2+1)} R\langle u\rangle s : M}\ \mathsf{app}$$

  where $\Gamma = \Gamma_1 \biguplus \Gamma_2$ and $(m, s) = (m_1 + m_2 + 1, s_1 + s_2 + 1)$. Note that since Γ is unitary co-shrinking, then so are Γ_1 and Γ_2.
  Since $R\langle u\rangle$ is not an answer, we may directly apply the *i.h.* on Φ_2 and obtain type derivation $\Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2',s_2')} s : N$ such that
  1. if $u \to_{\mathsf{wm}} u'$, then $m_2' = m_2 - 2$ and $s_2' = s_2 - 1$.
  2. if $u \to_{\mathsf{we}} u'$, then $m_2' = m_2$ and $s_2' < s_2$.

  We may then derive Ψ as follows:

$$\frac{\Phi_1 \rhd_{\mathrm{S}} [N \multimap M] \vdash^{(m_1,s_1)} R\langle u\rangle : \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2',s_2')} s : N}{\Gamma_1 \biguplus \Gamma_2 \vdash^{(m_1+m_2'+1,s_1+s_2'+1)} R\langle u\rangle s : M}\ \mathsf{app}$$

- *Rigid context with explicit substitution of rigid term*: Let $S \coloneqq R[x{\leftarrow}r]$. That is, $t = S\langle u\rangle = R\langle u\rangle[x{\leftarrow}r] \to_{\mathsf{s}_a} R\langle u'\rangle[x{\leftarrow}r] = S\langle u'\rangle = t'$ with $u \to_{\mathsf{w}_a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Then Φ must be of the following form

$$\frac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} R\langle u\rangle : M \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} r : N}{\Gamma_1 \biguplus \Gamma_2 \vdash^{(m_1+m_2,s_1+s_2+1)} R\langle u\rangle[x{\leftarrow}r] : M}\ \mathrm{ES}$$

  where $\Gamma = \Gamma_1 \biguplus \Gamma_2$ and $(m, s) = (m_1 + m_2, s_1 + s_2 + 1)$. Note that since Γ is unitary co-shrinking, then so are Γ_1 and Γ_2.
  First, note that since Γ_2 is unitary co-shrinking and r is a rigid term, then application of Lemma 13.8.2 (Spreading of co-shrinkingness) on Φ_2 gives that N is unitary co-shrinking. Hence, $\Gamma_1; x : N$ is unitary co-shrinking too.
  By *i.h.* on Φ_1, there exists $\Psi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1',s_1')} R\langle u'\rangle : M$ such that

1. if $u \to_{\mathsf{wm}} u'$, then $m'_2 = m_2 - 2$ and $s'_2 = s_2 - 1$.
2. if $u \to_{\mathsf{we}} u'$, then $m'_2 = m_2$ and $s'_2 < s_2$.

We may then derive Ψ as follows

$$\frac{\Psi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m'_1,s'_1)} R\langle u' \rangle : M \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} r : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m'_1+m_2,s'_1+s_2+1)} R\langle u' \rangle[x \leftarrow r] : M}\ \mathrm{ES}$$

- *Rigid term with explicit substitution of rigid context*: Let $S = r[x \leftarrow R]$. That is, $t = S\langle u \rangle = r[x \leftarrow R\langle u \rangle] \to_{\mathsf{s}_a} r[x \leftarrow R\langle u' \rangle] = S\langle u' \rangle = t'$ with $u \to_{\mathsf{w}_a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Then Φ must be of the following form

$$\frac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} r : M \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} R\langle u \rangle : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,s_1+s_2+1)} r[x \leftarrow R\langle u \rangle] : M}\ \mathrm{ES}$$

where $\Gamma = \Gamma_1 \uplus \Gamma_2$ and $(m, s) = (m_1 + m_2, s_1 + s_2 + 1)$. Note that Γ is unitary co-shrinking, and then so are Γ_1 and Γ_2.
Then, by *i.h.* on Φ_2—noting that $R\langle u \rangle$ is not an answer—there exists type derivation $\Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m'_2,s'_2)} R\langle u' \rangle : N$ such that

1. if $u \to_{\mathsf{wm}} u'$, then $m'_2 = m_2 - 2$ and $s'_2 = s_2 - 1$.
2. if $u \to_{\mathsf{we}} u'$, then $m'_2 = m_2$ and $s'_2 < s_2$.

We may then derive Ψ as follows

$$\frac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} r : M \quad \Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m'_2,s'_2)} R\langle u' \rangle : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m'_2,s_1+s'_2+1)} r[x \leftarrow R\langle u' \rangle] : M}\ \mathrm{ES}$$

This completes the proof for the unitary (co-)shrinking case. In the (non-unitary) shrinking case—under the weaker hypothesis that Γ is a co-shrinking context and if t is an abstraction up to ES then M is shrinking—the proof is analogous to the one for unitary shrinking, except for the *Abstraction* case. Indeed, while in the base case—*i.e.*, when the context is empty—the unitary (co-)shrinkingness property does not play any role, in the other (inductive) cases cases the statement follows by *i.h.* in a way analogous to when we assumed the unitary shrinkingness property. Let us see the only substantially different case:

- *Abstraction*: Let $S \coloneqq \lambda x.S'$. That is, $t = S\langle u \rangle = \lambda x.S'\langle u \rangle \to_{\mathsf{s}a} \lambda x.S'\langle u' \rangle = S\langle u' \rangle = t'$ with $u \to_{\mathsf{w}a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Since t is an answer, M is a shrinking multi type by hypothesis, and hence it has the form $M = [O_1 \multimap N_1, \ldots, O_n \multimap N_n]$ for some $n > 0$, where O_i is co-shrinking and N_i is shrinking for all $1 \leq i \leq n$. Therefore, Φ is of the following form:

$$\frac{\left(\dfrac{\Phi_i \rhd_{\mathrm{S}} \Gamma_i; x : O_i \vdash^{(m_i,s_i)} S\langle u \rangle : N_i}{\Gamma_i \vdash^{(m_i+1,s_i+1)} \lambda x.S\langle u \rangle : O_i \multimap N_i}\ \mathsf{fun}\right)_{i=1}^{n}}{\biguplus_{i=1}^{n} \Gamma_i \vdash^{(\left(\sum_{i=1}^{n} m_i\right)+n,\left(\sum_{i=1}^{n} s_i\right)+n)} \lambda x.S\langle u \rangle : [O_i \multimap N_i]_{i=1}^{n}}\ \mathsf{many}_{\lambda}$$

where $\Gamma = \biguplus_{i=1}^{n} \Gamma_i$, $(m, s) = ((\sum_{i=1}^{n} m_i) + n, (\sum_{i=1}^{n} s_i) + n)$ and $M = [O_i \multimap N_i]_{i=1}^{n}$.
Now, for all $1 \leq i \leq n$, we have that $\Gamma_i; x : O_i$ is a co-shrinking type context, by hypothesis. Moreover, N_i is shrinking, by hypothesis—because $M = [O_i \multimap N_i]_{i=1}^{n}$ is shrinking by hypothesis, since t is an answer.
Hence, for all $1 \leq i \leq n$, application of the *i.h.* on Φ_i gives the existence of type derivation $\Psi_i \rhd_{\mathrm{S}} \Gamma_i; x : O_i \vdash^{(m'_i,s'_i)} S\langle u' \rangle : N_i$ such that

1. if $u \to_{\mathsf{wm}} u$, then $m'_i = m_i - 2$ and $s'_i = s_i - 1$.
2. if $u \to_{\mathsf{we}} u'$, then $m'_i = m_i$ and $s'_i < s_i$.

We may then derive Ψ as follows:

$$\dfrac{\left(\dfrac{\Psi_i \rhd_{\mathrm{S}} \Gamma_i; x : O_i \vdash^{(m'_i,s'_i)} S\langle u' \rangle : N_i}{\Gamma_i \vdash^{(m'_i+1,s'_i+1)} \lambda x.S\langle u' \rangle : O_i \multimap N_i}\ \mathsf{fun}\right)_{i=1}^{n}}{\biguplus_{i=1}^{n} \Gamma_i \vdash^{(\left(\sum_{i=1}^{n} m'_i\right)+n,\left(\sum_{i=1}^{n} s'_i\right)+n)} \lambda x.S\langle u \rangle : [O_i \multimap N_i]_{i=1}^{n}}\ \mathsf{many}_{\lambda}$$

□

(Click here to go back to main chapter.)

Correctness for Strong CbV. Finally,

Theorem 13.8.9 (Shrinking Correctness for Strong CbV)**.**
Let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ be a shrinking (resp. unitary shrinking) type derivation. Then there exists $u \in \Lambda_{\mathsf{L}}$ such that

1. *u is in $\to_{\mathsf{s}}$-normal form,*
2. *there exists a reduction sequence $d : t \to_{\mathsf{s}}^{*} u$, and*
3. *$m \geq 2|d|_{\mathsf{m}} + |u|_{\mathrm{S}}$ (resp. $m = 2|d|_{\mathsf{m}} + |u|_{\mathrm{S}}$).*

Proof. (Click here to go back to main chapter.)
By induction on index s, and proceeding by case analysis on whether t $\to_{\mathsf{s}}$-reduces or not.

First, if t is in $\to_{\mathsf{s}}$-normal form, then $t = u$ is in $\to_{\mathsf{s}}$-normal form. By Proposition 10.3.2 (Fullness of Strong CbV), u is a strong super fireball. By Proposition 11.2.2 (Typing properties of Strong CbV-normal forms), we have that $m \geq |t|_{\mathrm{S}} = |u|_{\mathrm{S}} + |d|_{\mathsf{m}}$. Moreover, if Φ is *unitary* shrinking, then Proposition 11.2.2 gives that $m = |t|_{\mathrm{S}} = |u|_{\mathrm{S}} + |d|_{\mathsf{m}}$.

Otherwise, if $t \to_{\mathsf{s}} s$ for some $s \in \Lambda_{\mathsf{L}}$, then there are two sub-cases to consider:

1. *Multiplicative steps*: Let $t \to_{\mathsf{sm}} s$. By Proposition 11.2.5.1 (Shrinking Quantitative Subject Reduction for Strong CbV - multiplicative steps), there exists $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} s : M$ such that
   - If Φ is shrinking, then $m' \leq m - 2$ (entailing that $m \geq m' + 2$) and $s' < s$.
   - If Φ is unitary shrinking, then $m' = m - 2$ (entailing that $m = m' + 2$) and $s' = s - 1$.

   noting that Ψ is shrinking (resp. unitary shrinking) if and only if Φ is shrinking (resp. unitary shrinking).
   Then, by *i.h.* on Ψ, there exist $u \in \Lambda_{\mathsf{L}}$ such that
   (a) u is in $\to_{\mathsf{s}}$-normal form,
   (b) there exists a reduction sequence $d' : s \to_{\mathsf{s}}^{*} u$,
   (c) index m' is such that
      - if Ψ is shrinking, then $m' \geq 2|d'|_{\mathsf{m}} + |u|_{\mathrm{S}}$.
      - if Ψ is unitary shrinking, then $m' = 2|d'|_{\mathsf{m}} + |u|_{\mathrm{S}}$.

   Finally, we have that
   - if Φ is shrinking, then
   $$m \geq m' + 2 \geq (2|d'|_{\mathsf{m}} + |u|_{\mathrm{S}}) + 2 = 2(|d'|_{\mathsf{m}} + 1) + |u|_{\mathrm{S}} = 2|d|_{\mathsf{m}} + |u|_{\mathrm{S}}$$
   - if Φ is unitary shrinking, then
   $$m = m' + 2 = (2|d'|_{\mathsf{m}} + |u|_{\mathrm{S}}) + 2 = 2(|d'|_{\mathsf{m}} + 1) + |u|_{\mathrm{S}} = 2|d|_{\mathsf{m}} + |u|_{\mathrm{S}}$$

2. *Exponential steps*: Let $t \to_{\mathsf{se}} s$. By Proposition 11.2.5.2 (Shrinking Quantitative Subject Reduction for Strong CbV - exponential steps), there exists $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} s : M$ such that $m' = m$ and $s' < s$, noting that Ψ is shrinking (resp. unitary shrinking) if and only if Φ is shrinking (resp. unitary shrinking). Then, by *i.h.* on Ψ, there exist $u \in \Lambda_{\mathsf{L}}$ such that
   (a) u is in $\to_{\mathsf{s}}$-normal form,
   (b) there exists a reduction sequence $d' : s \to_{\mathsf{s}}^{*} u$,
   (c) index m' is such that
      - if Ψ is shrinking, then $m' \geq 2|d'|_{\mathsf{m}} + |u|_{\mathrm{S}}$.
      - if Ψ is unitary shrinking, then $m' = 2|d'|_{\mathsf{m}} + |u|_{\mathrm{S}}$.

   Finally, we have that
   - if Φ is shrinking, then
   $$m = m' \geq 2|d'|_{\mathsf{m}} + |u|_{\mathrm{S}} = 2|d|_{\mathsf{m}} + |u|_{\mathrm{S}}$$
   - if Φ is unitary shrinking, then
   $$m = m' = 2|d'|_{\mathsf{m}} + |u|_{\mathrm{S}} = 2|d|_{\mathsf{m}} + |u|_{\mathrm{S}}$$

□

(Click here to go back to main chapter.)

13.8.2 Strong CbV completeness

Typability of Strong CbV-normal forms.

Lemma 13.8.10 (Minimal typability of theoretical values)**.**
Let v_{T} be a theoretical value. There exists type derivation $\Phi \rhd_{\mathrm{S}} \emptyset \vdash^{(0,0)} t : \mathbf{0}$.

Proof. Trivial.

□

Proposition 13.8.11 (Shrinking typability of Strong CbV-normal forms)**.**
1. Inert*: For every strong inert term i_s and co-shrinking (resp. unitary co-shrinking) multi type M, there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} i_s : M$ for some $s \geq 1$, such that Γ is co-shrinking (resp. unitary co-shrinking) and $m \geq |i_s|_{\mathrm{S}}$ (resp. $m = |i_s|_{\mathrm{S}}$).*
2. Fireball*: For every strong fireball f_s, there exists a unitary shrinking type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} f_s : M$ such that $m = |f_s|_{\mathrm{S}}$ and $s \geq 1$.*

Proof. (Click here to go back to main chapter.)
Both points are proved by mutual induction on the definition of strong inert terms i_s and strong fireballs f_s. Cases:

- *Variable*: Let $i_s := x =: f_s$. Let $M = [L_1, \dots, L_n]$ be a co-shrinking (resp. unitary co-shrinking) multi type for some $n \in \mathbb{N}$. We can build Φ as follows
$$\dfrac{\left(\dfrac{}{x : [L_i] \vdash^{(0,1)} x : L_i}\ \mathsf{ax}\right)_{i=1}^{n}}{x : [L_i]_{i=1}^{n} \vdash^{(0,n)} x : [L_i]_{i=1}^{n}}\ \mathsf{many}_{\mathsf{VAR}}$$
  where $\Gamma = \{x : M\}$ is a co-shrinking (resp. unitary co-shrinking) context, with $M = [L_i]_{i=1}^{n}$. Note that if $M = [X]$ then M is both a unitary shrinking and a unitary co-shrinking multi type, and so Γ is a unitary co-shrinking context.

- *Inert application*: Let $i_s \coloneqq j_s f_s$. Let M be a co-shrinking (resp. unitary co-shrinking) multi type. By *i.h.*, there is a derivation $\Psi \rhd_{\mathrm{S}} \Pi \vdash^{(m_1,s_1)} f_s : N$ where Π is unitary co-shrinking and N is unitary shrinking. Hence, $[N \multimap M]$ is co-shrinking (resp. unitary co-shrinking). By *i.h.* (inert), there exists $\Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m_2,s_2)} i_s : [N \multimap M]$ where Δ is co-shrinking (resp. unitary co-shrinking). We can then derive

$$\frac{\Psi \rhd_{\mathrm{S}} \Pi \vdash^{(m_1,s_1)} f_s : N \quad \Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m_2,s_2)} i_s : [N \multimap M]}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,s_1+s_2+1)} i_s f_s : M}\ \mathsf{app}$$

  where $\Gamma = \Pi \biguplus \Delta$ is a co-shrinking (resp. unitary co-shrinking) context. Moreover, if $M = [X]$ then M is both a unitary shrinking and a unitary co-shrinking multi type, hence Γ is a unitary co-shrinking context (by *i.h.*).
- *Explicit substitution on inert*: Let $i_s \coloneqq j_s[x \leftarrow k_s]$. Let M be a co-shrinking (resp. unitary co-shrinking) multi type.
  By *i.h.* (inert), there exists $\Psi \rhd_{\mathrm{S}} \Pi; x : N \vdash^{(m_1,s_1)} j_s : M$ where $\Pi; x : N$ is a co-shrinking (resp. unitary co-shrinking) context; in particular, N is a co-shrinking (resp. unitary co-shrinking) multi type . By *i.h.* (2), there exists $\Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m_2,s_2)} k_s : N$ where Δ is a co-shrinking (resp. unitary co-shrinking) context and N is a co-shrinking (resp. unitary co-shrinking) multi type.
  We may then derive Φ as follows:

$$\frac{\Psi \rhd_{\mathrm{S}} \Pi; x : N \vdash^{(m_1,s_1)} j_s : M \quad \Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m_2,s_2)} k_s : N}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,s_1+s_2+1)} j_s[x \leftarrow s] : M}\ \mathrm{ES}$$

  where $\Gamma = \Pi \biguplus \Delta$ is a co-shrinking (resp. unitary co-shrinking) context. Moreover, if $M = [X]$ then M is both a unitary shrinking and a unitary co-shrinking multi type, hence Γ is a unitary co-shrinking context (by *i.h.*).
- *Abstraction*: Let $f_s \coloneqq \lambda x.g_s$. By *i.h.*, there exists $\Psi \rhd_{\mathrm{S}} \Pi; x : O \vdash^{(m_1,s_1)} g_s : N$ where $\Gamma; x : O$ is a unitary co-shrinking context and N is a unitary shrinking multi type. Note that then $M \coloneqq [O \multimap N]$ is a unitary co-shrinking multi type. We may then derive Φ as follows

$$\frac{\dfrac{\Psi \rhd_{\mathrm{S}} \Pi; x : O \vdash^{(m_1,s_1)} g_s : N}{\Pi \vdash^{(m_1+1,s_1+1)} \lambda x.g_s : O \multimap N}\ \mathsf{fun}}{\Pi \vdash^{(m_1+1,s_1+1)} \lambda x.g_s : [O \multimap N]}\ \mathsf{many}_\lambda$$

  noting that Π is a unitary co-shrinking context.
- *Explicit substitution on fireball*: Let $f_s \coloneqq g_s[x \leftarrow i_s]$.
  By *i.h.*, there exists $\Psi \rhd_{\mathrm{S}} \Pi; x : N \vdash^{(m_1,s_1)} g_s : M$ where M is a unitary co-shrinking multi type and $\Pi; x : N$ is a unitary co-shrinking context; in particular, N is a unitary co-shrinking multi type.
  By *i.h.*, there exists $\Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m_2,s_2)} i_s : N$ where Δ is a unitary co-shrinking context and N is a unitary co-shrinking multi type.
  We may then derive Φ as follows:

$$\frac{\Psi \rhd_{\mathrm{S}} \Pi; x : N \vdash^{(m_1,s_1)} g_s : M \quad \Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m_2,s_2)} i_s : N}{\Pi \biguplus \Delta \vdash^{(m_1+m_2,s_1+s_2+1)} g_s[x \leftarrow i_s] : M}\ \mathrm{ES}$$

  where $\Gamma = \Pi \biguplus \Delta$ is a unitary co-shrinking context.

w

□

(Click here to go back to main chapter.)

Removal for Strong CbV.

Lemma 13.8.12 (Removal for Strong CbV)**.**
Let $t \in \Lambda_{\mathsf{L}}$, let v_{T} be a theoretical value such that $x \notin \mathsf{fv}(v_{\mathrm{T}})$, and let

$$\Phi_{t\{x\leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t\{x\leftarrow v\} : M$$

Then there exist type derivations

$$\begin{array}{l}\Phi_t \rhd_{\mathrm{S}} \Pi; x : N \vdash^{(m',s')} t : M\\ \Theta \rhd_{\mathrm{S}} \Delta \vdash^{(m'',s'')} v : N\end{array}$$

such that $\Gamma = \Pi \uplus \Delta$, $m = m' + m''$ and $s \leq s' + s''$.

Proof. (Click here to go back to main chapter.)
By structural induction on the t. Cases:

- *Variable*: Let $t \in \mathsf{Var}$. There are two sub-cases:
  1. Let $t \coloneqq x$. That is, $t\{x\leftarrow v_{\mathrm{T}}\} = x\{x\leftarrow v_{\mathrm{T}}\} = v_{\mathrm{T}}$. Case analysis on whether $v_{\mathrm{T}} \in \mathsf{Var}$ or $v_{\mathrm{T}} \in \mathsf{Val}$:
     - Let $v_{\mathrm{T}} = y \in \mathsf{Var}$. We may safely assume that $y \neq x$. Then $\Phi_{t\{x\leftarrow v_{\mathrm{T}}\}}$ must be of the following form:
     $$\dfrac{\left(\dfrac{}{y : L_i \vdash^{(0,1)} y : L_i}\ \mathsf{ax}\right)_{i=1}^{n}}{y : [L_i]_{i=1}^{n} \vdash^{(0,n)} y : [L_i]_{i=1}^{n}}\ \mathsf{many}_{\mathsf{VAR}}$$
     where $\Gamma = \emptyset$ and $M = [L_1, \ldots, L_n] = N$.
     The statement holds by deriving Φ_t as
     $$\dfrac{\left(\dfrac{}{x : L_i \vdash^{(0,1)} x : L_i}\ \mathsf{ax}\right)_{i=1}^{n}}{x : [L_i]_{i=1}^{n} \vdash^{(0,n)} x : [L_i]_{i=1}^{n}}\ \mathsf{many}_{\mathsf{VAR}}$$
     and taking $\Theta \coloneqq \Phi_{t\{x\leftarrow v_{\mathrm{T}}\}}$.
     - The case where $v_{\mathrm{T}} \in \mathsf{Val}$ is analogous to the case where $v_{\mathrm{T}} \in \mathsf{Var}$, except that the role played by $\mathsf{many}_{\mathsf{VAR}}$ is now played by many_{λ}.
  2. Let $t \coloneqq z \neq x$. That is, $t\{x\leftarrow v_{\mathrm{T}}\} = z\{x\leftarrow v_{\mathrm{T}}\} = z$. Then $\Phi_{t\{x\leftarrow v_{\mathrm{T}}\}}$ must be of the following form:
  $$\dfrac{\left(\dfrac{}{z : L_i \vdash^{(0,0)} z : L_i}\ \mathsf{ax}\right)_{i=1}^{n}}{z : [L_i]_{i=1}^{n} \vdash^{(0,0)} z : [L_i]_{i=1}^{n}}\ \mathsf{many}_{\mathsf{VAR}}$$
  where $M = [L_i]_{i=1}^{n}$, $\Gamma = \{z : [L_i]_{i=1}^{n}\}$. The statement holds by taking $\Phi_t \coloneqq \Phi_{t\{x\leftarrow v_{\mathrm{T}}\}}$ and deriving Θ depending on whether $v_{\mathrm{T}} \in \mathsf{Var}$ or $v_{\mathrm{T}} \in \mathsf{Val}$:

– If $v_{\mathrm{T}} \in \mathsf{Var}$, then we derive Θ as

$$\frac{}{\emptyset \vdash^{(0,0)} v_{\mathrm{T}} : \mathbf{0}}\ \mathsf{many}_{\mathsf{VAR}}$$

– If $v_{\mathrm{T}} \in \mathsf{Val}$, then we derive Θ as

$$\frac{}{\emptyset \vdash^{(0,0)} v_{\mathrm{T}} : \mathbf{0}}\ \mathsf{many}_{\lambda}$$

- *Application*: Let $t \coloneqq sm$. That is, $t\{x{\leftarrow}v_{\mathrm{T}}\} = s\{x{\leftarrow}v_{\mathrm{T}}\}m\{x{\leftarrow}v_{\mathrm{T}}\}$. Then $\Phi_{t\{x\leftarrow v_{\mathrm{T}}\}}$ must be of the following form

$$\frac{\Phi_s \rhd_{\mathrm{S}} \Gamma_s \vdash^{(m_s,s_s)} s\{x{\leftarrow}v_{\mathrm{T}}\} : [O \multimap M] \quad \Phi_m \rhd_{\mathrm{S}} \Gamma_m \vdash^{(m_m,s_m)} m\{x{\leftarrow}v_{\mathrm{T}}\} : O}{\Gamma_s \uplus \Gamma_m \vdash^{(m_s+m_m+1,s_s+s_m+1)} s\{x{\leftarrow}v_{\mathrm{T}}\}m\{x{\leftarrow}v_{\mathrm{T}}\} : M}\ \mathsf{app}$$

where $\Gamma = \Gamma_s \uplus \Gamma_m$, $N = N_s \uplus N_m$ and $(m, s) = (m_s + m_m + 1, s_s + s_m + 1)$.
By *i.h.* on Φ_s, there exist type derivations

$$\begin{array}{l}\Phi_s \rhd_{\mathrm{S}} \Pi_s; x : N_s \vdash^{(m'_s,s'_s)} s : [O \multimap M] \\ \Theta_s \rhd_{\mathrm{S}} \Delta_s \vdash^{(m''_s,s''_s)} v_{\mathrm{T}} : O\end{array}$$

such that $\Gamma_s = \Pi_s \uplus \Delta_s$, $m_s = m'_s + m''_s$ and $s_s \leq s'_s + s''_s$.
Similarly, an application of the *i.h.* on Φ_m gives the existence of type derivations

$$\begin{array}{l}\Phi_m \rhd_{\mathrm{S}} \Pi_m; x : N_m \vdash^{(m'_m,s'_m)} m : O \\ \Theta_m \rhd_{\mathrm{S}} \Delta_m \vdash^{(m''_m,s''_m)} v_{\mathrm{T}} : O\end{array}$$

such that $\Gamma_m = \Pi_m \uplus \Delta_m$, $m_m = m'_m + m''_m$ and $s_m \leq s'_m + s''_m$.
By application of Lemma 13.8.13 (Merging multi types of Strong CbV type derivations) on Θ_s and Θ_m, we take Φ to be

$$\Theta \rhd_{\mathrm{S}} \Delta_s \biguplus \Delta_m \vdash^{(m''_s+m''_m,s''_s+s''_m)} v_{\mathrm{T}} : N_s \uplus N_m$$

Finally, we may derive Φ_t as follows

$$\frac{\Phi_s \rhd_{\mathrm{S}} \Pi_s; x : N_s \vdash^{(m'_s,s'_s)} s : [O \multimap M] \quad \Phi_m \rhd_{\mathrm{S}} \Pi_m; x : N_m \vdash^{(m'_m,s'_m)} m : O}{\Pi_s \uplus \Pi_m \vdash^{(m'_s+m'_m+1,s'_s+s'_m+1)} sm : M}\ \mathsf{app}$$

- *Abstraction*: Let $t \coloneqq \lambda y.s$, implying that $t\{x{\leftarrow}v_{\mathrm{T}}\} = \lambda y.(s\{x{\leftarrow}v_{\mathrm{T}}\})$. Note that we may safely assume that $y \notin (\mathsf{fv}(v_{\mathrm{T}}) \cup \{x\})$—by α-equivalence. Then $\Phi_{t\{x\leftarrow v_{\mathrm{T}}\}}$ must be of the following form:

$$\frac{\left(\dfrac{\Phi_i \rhd_{\mathrm{S}} \Gamma_i; y : O_i \vdash^{(m_i,s_i)} s\{x{\leftarrow}v_{\mathrm{T}}\} : P_i}{\Gamma_i \vdash^{(m_i+1,s_i+1)} \lambda y.s\{x{\leftarrow}v_{\mathrm{T}}\} : O_i \multimap P_i}\ \mathsf{fun}\right)_{i=1}^{n}}{\biguplus_{i=1}^{n} \Gamma_i \vdash^{(\left(\sum_{i=1}^{n} m_i\right)+n,\left(\sum_{i=1}^{n} s_i\right)+n)} \lambda y.s\{x{\leftarrow}v_{\mathrm{T}}\} : [O_i \multimap P_i]_{i=1}^{n}}\ \mathsf{many}_{\lambda}$$

where $\Gamma = \biguplus_{i=1}^{n} \Gamma_i$, $M = [O_i \multimap P_i]_{i=1}^{n}$ and $(m, s) = ((\sum_{i=1}^{n} m_i) + n, (\sum_{i=1}^{n} s_i) + n)$. Case analysis on whether $n > 0$:

– Let $n \coloneqq 0$. We may then derive Φ_t as

$$\frac{}{\emptyset \vdash^{(0,0)} \lambda y.s : \mathbf{0}}\ \mathsf{many}_\lambda$$

where the type context may be rewritten as $\emptyset; x : \mathbf{0}$. We may then derive Θ as

$$\frac{}{\emptyset \vdash^{(0,0)} v_{\mathrm{T}} : \mathbf{0}}\ \mathsf{many}_{\mathsf{VAR}}$$

if $v_{\mathrm{T}} \in \mathsf{Var}$, or as

$$\frac{}{\emptyset \vdash^{(0,0)} v_{\mathrm{T}} : \mathbf{0}}\ \mathsf{many}_\lambda$$

if $v_{\mathrm{T}} \in \mathsf{Val}$.

– Let $n > 0$. We apply the *i.h.* on each of the premises Φ_i. Thus, for all $1 \leq i \leq n$, there exist type derivations

$$\Phi_i \rhd_{\mathrm{S}} \Pi_i; y : O_i; x : N_i \vdash^{(m'_i, s'_i)} s : P_i$$
$$\Theta_i \rhd_{\mathrm{S}} \Delta_i \vdash^{(m''_i, s''_i)} v_{\mathrm{T}} : N_i$$

where $\Gamma_i = \Pi_i \uplus \Delta_i$, $m_i = m'_i + m''_i$ and $s_i \geq s'_i + s''_i$. Note that $\Delta_i(y) = \mathbf{0}$ for all $1 \leq i \leq n$, by Lemma 11.2.1 (Relevance of the Strong CbV type system) and the fact that $y \notin \mathsf{fv}(v_{\mathrm{T}})$.
We may then obtain Θ by repeated applications of Lemma 13.8.13 (Merging multi types of type derivations), giving us type derivation:

$$\Theta \rhd_{\mathrm{S}} \left(\biguplus_{i=1}^{n} \Delta_i \right) \vdash^{(\sum_{i=1}^n m''_i, \sum_{i=1}^n s''_i)} v_{\mathrm{T}} : (\uplus_{i=1}^n N_i)$$

Finally, we may obtain Φ_t as follows

$$\frac{\left(\dfrac{\Phi_i \rhd_{\mathrm{S}} \Pi_i; y : O_i; x : N_i \vdash^{(m'_i, s'_i)} s : P_i}{\Pi_i; x : N_i \vdash^{(m_i+1, s_i+1)} \lambda y.s : O_i \multimap P_i}\ \mathsf{fun} \right)_{i=1}^{n}}{(\biguplus_{i=1}^n \Pi_i)\,; x : (\uplus_{i=1}^n N_i) \vdash^{((\sum_{i=1}^n m_i)+n, (\sum_{i=1}^n s_i)+n)} \lambda y.s : [O_i \multimap P_i]_{i=1}^n}\ \mathsf{many}_\lambda$$

- *Explicit substitution*: Let $t \coloneqq s[y \leftarrow m]$, implying that $t\{x \leftarrow v_{\mathrm{T}}\} = s\{x \leftarrow v_{\mathrm{T}}\}[y \leftarrow m\{x \leftarrow v_{\mathrm{T}}\}]$. We may safely assume that $y \notin (\mathsf{fv}(v_{\mathrm{T}}) \cup \{x\})$—by α-equivalence. Then $\Phi_{t\{x \leftarrow v_{\mathrm{T}}\}}$ must be of the following form:

$$\frac{\Phi_s \rhd_{\mathrm{S}} \Gamma_s; y : O \vdash^{(m_s, s_s)} s\{x \leftarrow v_{\mathrm{T}}\} : M \quad \Phi_{m\{x \leftarrow v_{\mathrm{T}}\}} \rhd_{\mathrm{S}} \Gamma_m; y : \mathbf{0} \vdash^{(m_m, s_m)} m\{x \leftarrow v_{\mathrm{T}}\} : O}{(\Gamma_s \uplus \Gamma_m) \vdash^{(m_s + m_m, s_s + s_m + 1)} (s\{x \leftarrow v_{\mathrm{T}}\})[y \leftarrow m\{x \leftarrow v_{\mathrm{T}}\}] : M}\ \mathrm{ES}$$

where $\Gamma = \Gamma_s \uplus \Gamma_m$ and $(m, s) = (m_s + m_m + 1, s_s + s_m + 1)$.
By *i.h.* on Φ_s, there exist type derivations

$$\Phi_s \rhd_{\mathrm{S}} (\Pi_s; y : O)\,; x : N_s \vdash^{(m'_s, s'_s)} s : M$$
$$\Theta_s \rhd_{\mathrm{S}} \Delta_s \vdash^{(m''_s, s''_s)} v_{\mathrm{T}} : N_s$$

such that $\Gamma_s = (\Pi_s; y : O) \uplus \Delta_s$, $m_s = m'_s + m''_s$ and $s_s \leq s'_s + s''_s$. Note that $\Delta_s(y) = \mathbf{0}$ by the fact that $y \notin \mathsf{fv}(v_{\mathrm{T}})$ and by Lemma 11.2.1 (Relevance of the Strong CbV type system).

Similarly, an application of the *i.h.* on Φ_m gives the existence of type derivations

$$\Phi_m \rhd_{\mathrm{S}} \Pi_m; x : N_m \vdash^{(m'_m, s'_m)} m : O$$
$$\Theta_m \rhd_{\mathrm{S}} \Delta_m \vdash^{(m''_m, s''_m)} v_{\mathrm{T}} : O$$

such that $\Gamma_m = \Pi_m \uplus \Delta_m$, $m_m = m'_m + m''_m$ and $s_m \leq s'_m + s''_m$.
We may then obtain Θ by repeated applications of Lemma 13.8.13 (Merging multi types of Strong CbV type derivations), yielding type derivation

$$\Theta \rhd_{\mathrm{S}} \Delta_s \biguplus \Delta_m \vdash^{(m''_s + m''_m, s''_s + s''_m)} : N_s \uplus N_m$$

Finally, we may derive Φ_t as follows

$$\frac{\Phi_s \rhd_{\mathrm{S}} \Pi_s; x : N_s \vdash^{(m'_s, s'_s)} s : [O \multimap M] \quad \Phi_m \rhd_{\mathrm{S}} \Pi_m; x : N_m \vdash^{(m'_m, s'_m)} m : O}{\Pi_s \uplus \Pi_m \vdash^{(m'_s + m'_m + 1, s'_s + s'_m + 1)} sm : M}\ \mathrm{ES}$$

□

(Click here to go back to main chapter.)

Subject Expansion for Strong CbV. Analogously to the study of the Subject Reduction property, Subject Expansion for Strong CbV is obtained by first studying the weak case—*i.e.*, the Open evaluation strategy $\rightarrow_{\mathsf{w}}$—and then building on it to obtain the result for the strong case.

Firstly, the following is required to apply Lemma 11.2.8 (Removal for Strong CbV) in the proofs of Lemma 11.2.9.2 (Open Quantitative Subject Expansion for Strong CbV - exponential case) and Proposition 11.2.10.2 (Shrinking Quantitative Subject Expansion for Strong CbV - exponential case) to obtain the right indices.

Lemma 13.8.13 (Merging multi types of Strong CbV type derivations)**.**
Let $t \in (\mathsf{Var} \cup \mathsf{Val})$—i.e.*, t is a theoretical value. For any two type derivations*

$$\Phi_N \rhd_{\mathrm{S}} \Gamma_N \vdash^{(m_N, s_N)} t : N$$
$$\Phi_O \rhd_{\mathrm{S}} \Gamma_O \vdash^{(m_O, s_O)} t : O$$

there exists type derivation

$$\Phi_{N \uplus O} \rhd_{\mathrm{S}} \Gamma_N \vdash^{(m_N + m_O, s_N + s_O)} t : N$$

Proof.
Trivial, considering how the only typing rules deriving a multi type for theoretical values—*i.e.*, for variables and values—are $\mathsf{many}_{\mathsf{VAR}}$ and many_{λ}, whose premises can be joined at will.

□

Lemma 13.8.14 (Open Quantitative Subject Expansion for Strong CbV)**.**
Let $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} u : M$ be a type derivation.

1. Multiplicative*: If $t \rightarrow_{\mathsf{wm}} u$, then there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m+2, s')} t : M$ such that $s' > s$.*
2. Exponential*: If $t \rightarrow_{\mathsf{we}} u$, then there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m, s')} t : M$ such that $s' > s$.*

Proof. (Click here to go back to main chapter.)

1. *Multiplicative steps*: By induction on the open evaluation context O such that $t = O\langle s\rangle \to_{\mathsf{wm}} O\langle s'\rangle = u$, with $s \mapsto_{\mathsf{wm}} s'$. Cases:
   - *Context hole*: Let $O \coloneqq \langle\cdot\rangle$. Then $t = s = S\langle\lambda x.m\rangle\tilde{t}$ and $u = s' = S\langle m[x\leftarrow\tilde{t}]\rangle$. We proceed by induction on the length of S:
     - *Base case*: Let $S \coloneqq \langle\cdot\rangle$. Then Ψ is of the following form:

$$\frac{\Phi_m \rhd_{\mathrm{S}} \Gamma_m; x : N \vdash^{(m_m,s_m)} m : M \quad \Phi_{\tilde{t}} \rhd_{\mathrm{S}} \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}},s_{\tilde{t}})} \tilde{t} : N}{\Gamma_s \uplus \Gamma_m \vdash^{(m_m+m_{\tilde{t}},s_m+s_{\tilde{t}}+1)} \ : M}\ \mathrm{ES}$$

       The statement holds by deriving Ψ as follows

$$\dfrac{\dfrac{\dfrac{\Phi_m \rhd_{\mathrm{S}} \Gamma_m; x : N \vdash^{(m_m,s_m)} m : M}{\Gamma_m \vdash^{(m_m+1,s_m+1)} \lambda x.m : N \multimap M}\ \mathsf{fun}}{\Gamma_m \vdash^{(m_m+1,s_m+1)} \lambda x.m : [N \multimap M]}\ \mathsf{many} \qquad \Phi_{\tilde{t}} \rhd_{\mathrm{S}} \Gamma_{\tilde{t}} \vdash^{(m_{\tilde{t}},s_{\tilde{t}})} \tilde{t} : N}{\Gamma_m \uplus \Gamma_{\tilde{t}} \vdash^{(m_m+m_{\tilde{t}}+2,s_m+s_{\tilde{t}}+2)} (\lambda x.m)\tilde{t} : M}\ \mathsf{app}$$

     - *Inductive case*: Let $S \coloneqq S'[y\leftarrow\tilde{u}]$. Then Φ' is of the following form

$$\frac{\Phi_{S'\langle m[x\leftarrow\tilde{t}]\rangle} \rhd_{\mathrm{S}} \Gamma_1; y : N \vdash^{(m_1,s_1)} S'\langle m[x\leftarrow\tilde{t}]\rangle : M \quad \Phi_{\tilde{u}} \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} \tilde{u} : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,s_1+s_2+1)} S\langle m[x\leftarrow\tilde{t}]\rangle : M}\ \mathrm{ES}$$

       By *i.h.* on $\Phi_{S'\langle m[x\leftarrow\tilde{t}]\rangle}$, we get type derivation

$$\Phi_{\mathrm{ind}} \rhd_{\mathrm{S}} \Gamma_1; y : N \vdash^{(m_1+2,s')} S'\langle\lambda x.m\rangle\tilde{t} : M$$

       such that $s' > s$. Note that Φ_{ind} must must be of the following form

$$\frac{\Phi_1 \rhd_{\mathrm{S}} \Pi_1; y : N \vdash^{(m_1'',s_1'')} S'\langle\lambda x.m\rangle : [O \multimap M] \quad \Phi_2 \rhd_{\mathrm{S}} \Pi_2 \vdash^{(m_2'',s_2'')} \tilde{t} : O}{\Pi_1 \uplus \Pi_2 \vdash^{(m_1''+m_2''+1,s_1''+s_2''+1)} S'\langle\lambda x.m\rangle\tilde{t} : M}\ \mathsf{app}$$

       such that $\Gamma_1; y : N = (\Pi_1; y : N) \uplus \Pi_2$, $m_1 + 2 = m_1'' + m_2'' + 1$ and $s' = s_1'' + s_2'' + 1$. Therefore, the statement holds by deriving Φ as follows

$$\dfrac{\dfrac{\Phi_1 \qquad \Phi_{\tilde{u}}}{\Pi_1 \uplus \Gamma_2 \vdash^{(m_1''+m_2,s_1''+s_2+1)} S\langle\lambda x.m\rangle : [O \multimap M]}\ \mathrm{ES} \qquad \Phi_2 \rhd_{\mathrm{S}} \Pi_2 \vdash^{(m_2'',s_2'')} \tilde{t} : O}{\Pi_1 \uplus \Gamma_2 \uplus \Pi_2 \vdash^{(m_1''+m_2+m_2''+1,s_1''+s_2+1+s_2''+1)} (S\langle\lambda x.m\rangle)\tilde{t} : M}\ \mathsf{app}$$

   - *Left of an application*: Let $O \coloneqq O'm$. Then Ψ is of the following form:

$$\frac{\Phi_{O'\langle s'\rangle} \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_1,s_1)} O'\langle s'\rangle : [N \multimap M] \quad \Phi_m \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} m : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2+1,s_1+s_2+1)} O'\langle s'\rangle m : M}\ \mathsf{app}$$

     The statement follows easily by application of the *i.h.* on $\Phi_{O'\langle s'\rangle}$ and then deriving Φ by combining the result of the *i.h.* with Φ_m.
   - *Right of an application*: Analogous to the previous case.
   - *Left of an ES*: Let $O \coloneqq O'[x\leftarrow m]$. Then Ψ is of the following form:

$$\frac{\Phi_{O'\langle s'\rangle} \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} O'\langle s\rangle : M \quad \Phi_m \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} m : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,s_1+s_2+1)} O'\langle s'\rangle[x\leftarrow m] : M}\ \mathrm{ES}$$

     The statement follows easily by application of the *i.h.* on $\Phi_{O'\langle s\rangle}$ and then deriving Φ by combining the result of the *i.h.* with Φ_m.

- *Inside of an ES*: Analogous to the previous case.

2. *Exponential steps*: By induction on the open evaluation context O such that $t = O\langle s\rangle \to_{\mathsf{we}} O\langle s'\rangle = u$, with $s = s''[x{\leftarrow}S\langle v_{\mathrm{T}}\rangle] \mapsto_{\mathsf{we}} S\langle s''\{x{\leftarrow}v_{\mathrm{T}}\}\rangle = s'$:
   - *Context hole*: Let $O \coloneqq \langle\cdot\rangle$. Then $t = s = s''[x{\leftarrow}S\langle v_{\mathrm{T}}\rangle]$ and $u = s' = S\langle s''\{x{\leftarrow}v_{\mathrm{T}}\}\rangle$. We proceed by induction on the length of S:
     - *Base case*: Let $S \coloneqq \langle\cdot\rangle$. Then $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} s''\{x{\leftarrow}v_{\mathrm{T}}\} : M$. By Lemma 11.2.8 (Removal for Strong CbV), there exist

$$\begin{array}{l}\Phi_{s'} \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m'_1,s'_1)} s'' : M\\ \Phi_{v_{\mathrm{T}}} \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m'_2,s'_2)} v_{\mathrm{T}} : N\end{array}$$

     such that $\Gamma = \Gamma_1 \uplus \Gamma_2$ and $m' = m'_1 + m'_2$ and $s' \le s'_1 + s'_2$.
     The statement holds by deriving Φ as follows:

$$\frac{\Phi_{s'} \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m'_1,s'_1)} s'' : M \quad \Phi_{v_{\mathrm{T}}} \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m'_2,s'_2)} v_{\mathrm{T}} : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m'_1+1,s'_1+s'_2+1)} s''[x{\leftarrow}v_{\mathrm{T}}] : M}\ \mathrm{ES}$$

     - *Inductive case*: Let $S \coloneqq S'[y{\leftarrow}m]$. Then Ψ must be of the following form

$$\frac{\Phi_{S'\langle s''\{x{\leftarrow}v_{\mathrm{T}}\}\rangle} \rhd_{\mathrm{S}} \Gamma_1; y : N \vdash^{(m_1,s_1)} S'\langle s''\{x{\leftarrow}v_{\mathrm{T}}\}\rangle : M \quad \Phi_m \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} m : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,s_1+s_2+1)} (S'[y{\leftarrow}m])\langle s''\{x{\leftarrow}v_{\mathrm{T}}\}\rangle : M}\ \mathrm{ES}$$

     By *i.h.* on $\Phi_{S'\langle s''\{x{\leftarrow}v_{\mathrm{T}}\}\rangle}$, there exist type derivation

$$\Phi_{\mathrm{ind}} \rhd_{\mathrm{S}} \Gamma_1; y : N \vdash^{(m_1,s'_1)} s''[x{\leftarrow}S'\langle v_{\mathrm{T}}\rangle] : M$$

     such that $s'_1 > s'$. Moreover, note that Φ_{ind} must be of the following form

$$\frac{\Psi_1 \rhd_{\mathrm{S}} \Pi_1; x : N \vdash^{(m''_1,s''_1)} s'' : M \quad \Psi_2 \rhd_{\mathrm{S}} \Pi_2 \vdash^{(m''_2,s''_2)} S'\langle v_{\mathrm{T}}\rangle : N}{\Pi_1 \uplus \Pi_2 \vdash^{(m''_1+m''_2,s''_1+s''_2+1)} s''[x{\leftarrow}S'\langle v_{\mathrm{T}}\rangle] :}\ \mathrm{ES}$$

     where $\Gamma_1; y : N = \Pi_1 \uplus \Pi_2$ and $(m_1, s'_1) = (m''_1 + m''_2, s''_1 + s''_2 + 1)$.
     The statement then holds by deriving Φ as follows

$$\frac{\Psi_1 \qquad \dfrac{\Psi_2 \qquad \Phi_m}{\Pi_2 \uplus \Gamma_2 \vdash^{(m''_2+m_2,s''_2+s'+1)} S\langle v_{\mathrm{T}}\rangle : N}\ \mathrm{ES}}{\Pi_1 \uplus \Pi_2 \uplus \Gamma_2 \vdash^{(m''_1+m''_2+m_2,s''_1+s''_2+s'+2)} s''[x{\leftarrow}S\langle v_{\mathrm{T}}\rangle] : M}\ \mathrm{ES}$$

   - *Left of an application*: Let $O \coloneqq O'm$. Then Φ is of the following form:

$$\frac{\Phi_{O'\langle s'\rangle} \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_1,s_1)} O'\langle s\rangle : [N \multimap M] \quad \Phi_m \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} m : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2+1,s_1+s_2+1)} O'\langle s'\rangle m : M}\ \mathsf{app}$$

     The statement follows easily by application of the *i.h.* on $\Phi_{O'\langle s\rangle}$ and then deriving Ψ by combining the result of the *i.h.* with Φ_m.
   - *Right of an application*: Analogous to the previous case.

- *Left of an ES*: Let $O \coloneqq O'[x \leftarrow u]$. Then Φ is of the following form:

$$\frac{\Phi_{O'\langle s'\rangle} \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} O'\langle s'\rangle : M \quad \Phi_m \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} m : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,s_1+s_2+1)} O'\langle s'\rangle[x \leftarrow m] : M}\ \mathrm{ES}$$

  The statement follows easily by application of the *i.h.* on $\Phi_{O'\langle s\rangle}$ and then deriving Ψ by combining the result of the *i.h.* with Φ_m.
- *Inside of an ES*: Analogous to the previous case.

□

(Click here to go back to main chapter.)
We can now prove the following

Proposition 13.8.15 (Shrinking Quantitative Subject Expansion for Strong CbV)**.**
Let $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} t' : M$ be a type derivation, with Γ a unitary co-shrinking type context. Moreover, suppose that if u is an answer, then M is unitary shrinking.

1. Multiplicative*: If $t \to_{\mathsf{sm}} t'$, then there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m+2,s+1)} t : M$.*
2. Exponential*: If $t \to_{\mathsf{se}} t'$, then there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} t : M$ such that $m' = m$ and $s' > s$.*

Proof. (Click here to go back to main chapter.)
By induction on the evaluation context S in the step $t = S\langle u\rangle \to_{\mathsf{s}} S\langle u'\rangle = t'$ with $u \to_{\mathsf{wm}} u'$ or $u \to_{\mathsf{we}} u'$. Cases for S:

- *Hole context*: Let $S \coloneqq \langle\cdot\rangle$ and $t \to_{\mathsf{w}a} t'$ with $a \in \{\mathsf{m}, \mathsf{e}\}$. By Lemma 11.2.4 (Open Quantitative Subject Reduction for Strong CbV) on Ψ, we have that
  - if $t \to_{\mathsf{wm}} t'$, then there exists $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m'+2,s'+1)} t : M$.
  - if $t \to_{\mathsf{we}} t'$, then there exists $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ such that $m = m'$ and $s > s'$.

  Note that, in this case, we do not use the hypothesis that Γ is a unitary co-shrinking context and that M is a unitary shrinking multi type if t' is an answer.
- *Abstraction*: Let $S \coloneqq \lambda x.S'$. Then, $t = S\langle u\rangle = \lambda x.S'\langle u\rangle \to_{\mathsf{s}a} \lambda x.S'\langle u'\rangle = S\langle u'\rangle = t'$ with $u \to_{\mathsf{w}a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Since t' is an abstraction, M is a unitary shrinking multi type by hypothesis and hence it has the form $M = [O \multimap N]$ where O is unitary co-shrinking and N is unitary shrinking. Then Ψ must be of the following form

$$\dfrac{\dfrac{\Psi_1 \rhd_{\mathrm{S}} \Gamma; x : O \vdash^{(m_1',s_1')} S'\langle u'\rangle : N}{\Gamma \vdash^{(m_1'+1,s_1'+1)} S'\langle u'\rangle : O \multimap N}\ \mathsf{fun}}{\Gamma \vdash^{(m_1'+1,s_1'+1)} S'\langle u'\rangle : [O \multimap N]}\ \mathsf{many}_\lambda$$

  where $(m', s') = (m_1 + 1, s_1 + 1)$.
  Since $\Gamma; x : O$ is a unitary co-shrinking type context, then we may apply the *i.h.* on Ψ_1 to get type derivation $\Phi_1 \rhd_{\mathrm{S}} \Gamma; x : O \vdash^{(m_1,s_1)} S'\langle u\rangle : N$ such that
  1. if $u \to_{\mathsf{wm}} u'$, then $m_1 = m_1' + 2$ and $s_1 = s_1' + 1$.
  2. if $u \to_{\mathsf{we}} u'$, then $m_1 = m_1'$ and $s_1 < s_1'$.

  We may then derive Φ as follows

$$\dfrac{\dfrac{\Phi_1 \rhd_{\mathrm{S}} \Gamma; x : O \vdash^{(m_1,s_1)} S'\langle u'\rangle : N}{\Gamma \vdash^{(m_1+1,s_1+1)} S'\langle u'\rangle : O \multimap N}\ \mathsf{fun}}{\Gamma \vdash^{(m_1+1,s_1+1)} S'\langle u'\rangle : [O \multimap N]}\ \mathsf{many}_\lambda$$

- *Explicit substitution of rigid context*: Let $S \coloneqq s[x{\leftarrow}R]$. That is, $t = S\langle u\rangle = s[x{\leftarrow}R\langle u\rangle] \to_{\mathsf{s}_a} s[x{\leftarrow}R\langle u'\rangle] = S\langle u'\rangle = t'$ with $u \to_{\mathsf{w}_a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Then Ψ must be of the following form
$$\frac{\Psi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m'_1,s'_1)} s : M \quad \Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m'_2,s'_2)} R\langle u'\rangle : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m'_1+m'_2,s'_1+s'_2+1)} s[x{\leftarrow}R\langle u'\rangle] : M}\ \mathrm{ES}$$
where $\Gamma = \Gamma_1 \uplus \Gamma_2$ and $(m', s') = (m'_1 + m'_2, s'_1 + s'_2 + 1)$. Note that since Γ is unitary co-shrinking, then so are Γ_1 and Γ_2.
By *i.h.* on Ψ_2—noting that $R\langle u'\rangle$ is not an answer—there exists type derivation $\Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} R\langle u\rangle : N$ such that:
  1. if $u \to_{\mathsf{wm}} u'$, then $m_2 = m'_2 + 2$ and $s_2 = s'_2 + 1$.
  2. if $u \to_{\mathsf{we}} u'$, then $m_2 = m'_2$ and $s_2 < s'_2$.

We may then derive Φ as follows
$$\frac{\Psi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m'_1,s'_1)} s : M \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} R\langle u\rangle : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m'_1+m_2,s'_1+s_2+1)} s[x{\leftarrow}R\langle u\rangle] : M}\ \mathrm{ES}$$

- *Explicit substitution on strong*: Let $S \coloneqq S'[x{\leftarrow}r]$. That is, $t = S\langle u\rangle = S'\langle u\rangle[x{\leftarrow}r] \to_{\mathsf{s}_a} S'\langle u'\rangle[x{\leftarrow}r] = S\langle u'\rangle = t'$ with $u \to_{\mathsf{w}_a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Then Ψ must be of the following form:
$$\frac{\Psi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m'_1,s'_1)} S'\langle u'\rangle : M \quad \Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m'_2,s'_2)} r : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m'_1+m'_2,s'_1+s'_2+1)} S'\langle u'\rangle[x{\leftarrow}r] : M}\ \mathrm{ES}$$
where $\Gamma = \Gamma_1 \uplus \Gamma_2$ and $(m', s') = (m'_1 + m'_2, s'_1 + s'_2 + 1)$. Note that since Γ is unitary co-shrinking, then so are Γ_1 and Γ_2.
First, note that since r is a rigid term and Γ_2 is unitary co-shrinking, we may apply Lemma 13.8.2 (Spreading of co-shrinkingness) on Ψ_2 obtaining that N is unitary co-shrinking. Hence, $\Gamma_1; x : O$ is unitary co-shrinking. Moreover, note that $S'\langle u'\rangle[x{\leftarrow}r]$ is an answer if and only if $S'\langle u\rangle$ is an answer. Hence, we may apply the *i.h.* on Ψ_1 to obtain type derivation $\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} S'\langle u\rangle : M$ such that
  1. if $u \to_{\mathsf{wm}} u'$, then $m_1 = m'_1 + 2$ and $s_1 = s'_1 + 1$.
  2. if $u \to_{\mathsf{we}} u'$, then $m_1 = m'_1$ and $s_1 < s'_1$.

We may then derive Φ as follows
$$\frac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} S'\langle u\rangle : M \quad \Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m'_2,s'_2)} r : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m'_2,s_1+s'_2+1)} S'\langle u\rangle[x{\leftarrow}r] : M}\ \mathrm{ES}$$

- *Application to strong*: Let $S \coloneqq rS'$. That is, $t = S\langle u\rangle = rS'\langle u\rangle \to_{\mathsf{s}_a} rS'\langle u'\rangle = S\langle u'\rangle = t'$ with $u \to_{\mathsf{w}_a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Then Ψ must be of the following form
$$\frac{\Psi_1 \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m'_1,s'_1)} r : [N \multimap M] \quad \Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m'_2,s'_2)} S'\langle u'\rangle : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m'_1+m'_2+1,s'_1+s'_2+1)} rS'\langle u'\rangle : M}\ \mathsf{app}$$
where $\Gamma = \Gamma_1 \uplus \Gamma_2$ and $(m', s') = (m'_1 + m'_2 + 1, s'_1 + s'_2 + 1)$. Note that since Γ is unitary co-shrinking, then so are Γ_1 and Γ_2.
By Lemma 13.8.2(Spreading of co-shrinkingness) on Ψ_1—given that Γ_1 is unitary co-shrinking and that r is a rigid term—we have that $[N \multimap M]$ is unitary co-shrinking, implying that N is unitary shrinking.
Hence, we may apply the *i.h.* on Ψ_2 obtaining type derivation $\Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} S'\langle u\rangle : N$ such that

1. if $u \to_{\mathsf{wm}} u'$, then $m_2 = m_2' + 2$ and $s_2 = s_2' + 1$.
2. if $u \to_{\mathsf{we}} u'$, then $m_2 = m_2'$ and $s_2 < s_2'$.

We may then build Φ as follows

$$\frac{\Psi_1 \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_1',s_1')} r : [N \multimap M] \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} S'\langle u\rangle : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1'+m_2+1,s_1'+s_2+1)} rS'\langle u'\rangle : M}\ \mathsf{app}$$

- *Application of inert*: Let $S \coloneqq Rs$. That is, $t = S\langle u\rangle = R\langle u\rangle s \to_{\mathsf{s}_a} R\langle u'\rangle s = S\langle u'\rangle = t'$ with $u \to_{\mathsf{w}_a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Then Ψ must be of the following form

$$\frac{\Psi_1 \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_1',s_1')} R\langle u'\rangle : [N \multimap M] \quad \Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2',s_2')} S'\langle s\rangle : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1'+m_2'+1,s_1'+s_2'+1)} R\langle u'\rangle s : M}\ \mathsf{app}$$

  where $\Gamma = \Gamma_1 \uplus \Gamma_2$ and $(m', s') = (m_1' + m_2' + 1, s_1' + s_2' + 1)$. Note that since Γ is unitary co-shrinking, then so are Γ_1 and Γ_2.
  Since $R\langle u'\rangle$ is not an answer, we may directly apply the *i.h.* on Ψ_1 obtaining type derivation $\Phi_1 \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_1,s_1)} R\langle u\rangle : [N \multimap M]$ such that
  1. if $u \to_{\mathsf{wm}} u'$, then $m_1 = m_1' + 2$ and $s_1 = s_1' + 1$.
  2. if $u \to_{\mathsf{we}} u'$, then $m_1 = m_1'$ and $s_1 < s_1'$.

  We may finally derive Φ as follows

$$\frac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_1,s_1)} R\langle u\rangle : [N \multimap M] \quad \Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2',s_2')} S'\langle s\rangle : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2'+1,s_1+s_2'+1)} R\langle u\rangle s : M}\ \mathsf{app}$$

- *Explicit substitution on inert*: Let $S \coloneqq R[x{\leftarrow}r]$. That is, $t = S\langle u\rangle = R\langle u\rangle[x{\leftarrow}r] \to_{\mathsf{s}_a} R\langle u'\rangle[x{\leftarrow}r] = S\langle u'\rangle = t'$ with $u \to_{\mathsf{w}_a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Then Ψ must be of the following form

$$\frac{\Psi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1',s_1')} R\langle u'\rangle : M \quad \Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2',s_2')} r : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1'+m_2',s_1'+s_2'+1)} R\langle u'\rangle[x{\leftarrow}r] : M}\ \mathrm{ES}$$

  where $\Gamma = \Gamma_1 \uplus \Gamma_2$ and $(m', s') = (m_1' + m_2', s_1' + s_2' + 1)$. Note that since Γ is unitary co-shrinking, then so are Γ_1 and Γ_2.
  First, by Lemma 13.8.2 (Spreading of co-shrinkingness) on Ψ_2—given that r is a rigid term—we have that N is a unitary co-shrinking multi type. Hence, $\Gamma_1; x : N$ is a unitary co-shrinking type context. We may then apply the *i.h.* on Ψ_1, obtaining type derivation $\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} R\langle u\rangle : M$ such that
  1. if $u \to_{\mathsf{wm}} u'$, then $m_1 = m_1' + 2$ and $s_1 = s_1' + 1$.
  2. if $u \to_{\mathsf{we}} u'$, then $m_1 = m_1'$ and $s_1 < s_1'$.

  We may finally derive Φ as follows

$$\frac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} R\langle u\rangle : M \quad \Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2',s_2')} r : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2',s_1+s_2'+1)} R\langle u\rangle[x{\leftarrow}r] : M}\ \mathrm{ES}$$

- *Explicit substitution of inert on rigid*: Let $S \coloneqq r[x{\leftarrow}R]$. That is, $t = S\langle u\rangle = r[x{\leftarrow}R\langle u\rangle] \to_{\mathsf{s}_a} r[x{\leftarrow}R\langle u'\rangle] = S\langle u'\rangle = t'$ with $u \to_{\mathsf{w}_a} u'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Then Ψ must be of the following form

$$\frac{\Psi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1',s_1')} r : M \quad \Psi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2',s_2')} R\langle u'\rangle : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1'+m_2',s_1'+s_2'+1)} R\langle u'\rangle[x{\leftarrow}r] : M}\ \mathrm{ES}$$

where $\Gamma = \Gamma_1 \biguplus \Gamma_2$ and $(m', s') = (m'_1 + m'_2, s'_1 + s'_2 + 1)$. Note that since Γ is unitary co-shrinking, then so are Γ_1 and Γ_2.
By *i.h.* applied to Ψ_2—directly applicable because $R\langle u\rangle$ is not an answer—there exists type derivation $\Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} R\langle u\rangle : N$ such that

1. if $u \to_{\mathsf{wm}} u'$, then $m_2 = m'_2 + 2$ and $s_2 = s'_2 + 1$.
2. if $u \to_{\mathsf{we}} u'$, then $m_2 = m'_2$ and $s_2 < s'_2$.

We may then derive Φ as follows:

$$\frac{\Psi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m'_1,s'_1)} r : M \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} R\langle u\rangle : N}{\Gamma_1 \biguplus \Gamma_2 \vdash^{(m'_1+m_2,s'_1+s_2+1)} r[x \leftarrow R\langle u\rangle] : M}\ \mathrm{ES}$$

□

(Click here to go back to main chapter.)

Completeness for Strong CbV. Finally,

Theorem 13.8.16 (Shrinking Completeness for Strong CbV)**.**
Let $t \in \Lambda_{\mathsf{L}}$. If there exists $d : t \to_{\mathsf{s}}^* u$ such that u in $\to_{\mathsf{s}}$-normal form, then there exists a unitary shrinking type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(2|d|_{\mathsf{m}}+|u|_{\mathrm{S}},s)} t : M$, for some $s \geq 0$.

Proof. *(Click here to go back to main chapter.)*
By induction on the length of $|d|$ of the reduction sequence $d : t \to_{\mathsf{s}}^* u$:

- *Base case*: Let $k \coloneqq 0$. Then t is in $\to_{\mathsf{s}}$-normal form and $t = u$. By Proposition 10.3.2 (Fullness of Strong CbV), t is a strong super fireball. The statement then holds by Proposition 11.2.7 (Shrinking typability of Strong CbV-normal forms).
- *Inductive case*: Let $k > 0$. That is, d is of the following form

$$d : t \to_{\mathsf{s}} \underbrace{s \to_{\mathsf{s}}^{k-1} u}_{d'}$$

  By *i.h.* on $d' : s \to_{\mathsf{s}}^* u$, there exists unitary shrinking type derivation

$$\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(2|d'|_{\mathsf{m}}+|u|_{\mathrm{S}},s')} s : M$$

  for some $s' \geq 0$. We proceed by case analysis on the kind of reduction step in $t \to_{\mathsf{s}} s$:
  - *Multiplicative step*: Let $t \to_{\mathsf{sm}} s$. By Proposition 11.2.10.1 (Shrinking Quantitative Subject Expansion for Strong CbV - multiplicative steps), there exists type derivation

$$\Theta \rhd_{\mathrm{S}} \Gamma \vdash^{(2|d'|_{\mathsf{m}}+|u|_{\mathrm{S}}+2,s'+1)} s : M$$

    noting that

$$2|d'|_{\mathsf{m}} + |u|_{\mathrm{S}} + 2 = 2(|d'|_{\mathsf{m}} + 1) + |u|_{\mathrm{S}} = 2|d|_{\mathsf{m}} + |u|_{\mathrm{S}}$$

    and that $s' \geq 0$. The statement follows by taking $\Phi \coloneqq \Theta$.
  - *Exponential step*: Let $t \to_{\mathsf{se}} s$. By Proposition 11.2.10.2 (Shrinking Quantitative Subject Expansion for Strong CbV - exponential steps), there exists type derivation

$$\Theta \rhd_{\mathrm{S}} \Gamma \vdash^{(2|d'|_{\mathsf{m}}+|u|_{\mathrm{S}},s'')} s : M$$

    for some $s'' > s' \geq 0$. Note that $2|d'|_{\mathsf{m}} + |u|_{\mathrm{S}} = 2|d|_{\mathsf{m}} + |u|_{\mathrm{S}}$. The statement follows by taking $\Phi \coloneqq \Theta$.

□

(Click here to go back to main chapter.)

13.8.3 Size of Strong CbV-normal forms via multi types

Proposition 13.8.17 (Shrinking types bound the size of Strong CbV-normal forms)**.**

Let $t \in \Lambda_{\mathsf{L}}$ be in $\to_{\mathsf{s}}$-normal form and let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ be a type derivation, with Γ a co-shrinking type context.

1. Inert*: If t is a strong inert term, then $|M| + |t|_{\mathrm{S}} \leq |\hat{\Gamma}|$.*
2. Fireball*: If t is a strong fireball and M is shrinking, then $|t|_{\mathrm{S}} \leq |M| + |\hat{\Gamma}|$*

Proof. (Click here to go back to main chapter.)

By mutual induction on the definition of strong inert term and strong fireball. Note that the statement for strong inert terms implies the one for strong fireballs, and so it is enough to prove Point 1 for strong inert terms. Cases:

- *Variable*: Let $t \coloneqq x \in \mathsf{Var}$. Then Φ must be of the following form

$$\frac{\left(\dfrac{}{x : [L_i] \vdash^{(0,1)} x : L_i}\ \mathsf{ax}\right)_{i=1}^{n}}{x : [L_i]_{i=1}^{n} \vdash^{(0,n)} x : [L_i]_{i=1}^{n}}\ \mathsf{many}_{\mathsf{VAR}}$$

  where $\Gamma = \{x : M\}$ and $M = [L_i]_{i=1}^{n}$. Since $|t|_{\mathrm{S}} = 0$, we have that $|M| + |t|_{\mathrm{S}} = |\hat{\Gamma}|$.
- *Application*: Let $t \coloneqq i_s f_s$. Then Φ must be of the following form:

$$\frac{\Psi \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_1,s_1)} i_s : [N \multimap M] \quad \Theta \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} f_s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2+1, s_1+s_2+1)} i_s f_s : M}\ \mathsf{app}$$

  where $\Gamma = \Gamma_1 \uplus \Gamma_2$. Note that since Γ is co-shrinking, then so are Γ_1 and Γ_2.
  First, note that since i_s is a rigid term and Γ_1 is co-shrinking, we have that $[N \multimap M]$ is co-shrinking, by Lemma 13.8.2 (Spreading of co-shrinkingness). This entails that N is shrinking.
  By *i.h.* (1) on Ψ, we have that $|[N \multimap M]| + |i_s|_{\mathrm{S}} \leq |\hat{\Gamma_1}|$.
  By *i.h.* (2) on Θ, we have that $|f_s|_{\mathrm{S}} \leq |N| + |\hat{\Gamma_2}|$.
  Then

$$\begin{array}{rcl} |M| + |i_s f_s|_{\mathrm{S}} & = & |M| + |i_s|_{\mathrm{S}} + |f_s|_{\mathrm{S}} + 1 \\ & \leq & |M| + |i_s|_{\mathrm{S}} + (|N| + |\hat{\Gamma_2}|) + 1 \\ & = & |\hat{\Gamma_2}| + |[N \multimap M]| + |i_s|_{\mathrm{S}} \\ & \leq & |\hat{\Gamma_1}| + |\hat{\Gamma_2}| \\ & = & |\hat{\Gamma}| \end{array}$$

  Since t is a strong inert term, Point 2 follows from Point 1.
- *Explicit substitution on inert*: Let $t \coloneqq i_s[x \leftarrow j_s]$. Then Φ must be of the following form:

$$\frac{\Psi \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} i_s : M \quad \Theta \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} f_s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2, s_1+s_2+1)} i_s[x \leftarrow f_s] : M}\ \mathrm{ES}$$

  Note that since Γ is co-shrinking, then so are Π and Δ.
  First, since j_s is a rigid term, we can apply Lemma 13.8.2 (Spreading of co-shrinkingness) to Θ, getting that N is co-shrinking. This entails that $\Pi; x : N$ is a co-shrinking type context.
  By *i.h.* (1) on Ψ, we get that $|M| + |i_s|_{\mathrm{S}} \leq |(\Gamma_1; \hat{x : N})| = |\hat{\Gamma_1}| + |N|$.
  By *i.h.* (2) on Θ, we get that $|N| + |j_s|_{\mathrm{S}} \leq |\hat{\Gamma_2}|$.

Then

$$|M| + |t|_{\mathrm{S}} \quad = \quad |M| + |i_s|_{\mathrm{S}} + |j_s|_{\mathrm{S}} \leq (|\hat{\Gamma_1}| + |N|) + |j_s|_{\mathrm{S}} \leq |\hat{\Gamma_1}| + |\hat{\Gamma_2}| = |\hat{\Gamma}|$$

Since t is a strong inert term, Point 2 follows from Point 1.

- *Explicit substitution on fireball*: Let $t \coloneqq f_s[x \leftarrow i_s]$. Then Φ must be of the following form

$$\frac{\Psi \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} f_s : M \quad \Theta \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} i_s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2, s_1+s_2+1)} f_s[x \leftarrow i_s] : M}\ \mathrm{ES}$$

Note that since $\Gamma = \Pi \uplus \Delta$ is co-shrinking, then so are Π and Δ.
First, and since i_s is a rigid term, we have that N is co-shrinking, by Lemma 13.8.2 (Spreading of co-shrinkingness) on Θ. This entails entails that $\Pi; x : N$ is co-shrinking too.
By *i.h.* (2) on Ψ, we have that $|f_s|_{\mathrm{S}} \leq |M| + |(\Gamma_1; \hat{x} : N)|$.
By *i.h.* (1) on Θ, we have that $|N| + |i_s|_{\mathrm{S}} \leq |\hat{\Gamma_2}|$.
Thus, we have that

$$\begin{array}{rcl} |t|_{\mathrm{S}} & = & |f_s|_{\mathrm{S}} + |i_s|_{\mathrm{S}} \\ & \leq & \Big(|M| + |(\Gamma_1; \hat{x} : N)|\Big) + |i_s|_{\mathrm{S}} \\ & = & |M| + |\hat{\Gamma_1}| + |\hat{N}| + |i_s|_{\mathrm{S}} \\ & \leq & |M| + |\hat{\Gamma_1}| + |\hat{\Gamma_2}| \\ & = & |M| + |\hat{\Gamma}| \end{array}$$

- *Abstraction*: Let $t \coloneqq \lambda x.f_s$. Then Φ must be of the following form

$$\frac{\left(\dfrac{\Phi_i \rhd_{\mathrm{S}} \Gamma_i; x : N_i \vdash^{(m_i,s_i)} f_s : M_i}{\Gamma_i \vdash^{(m_i+1,s_i+1)} \lambda x.f_s : N_i \multimap M_i}\ \mathsf{fun}\right)_{i=1}^{n}}{\biguplus_{i=1}^{n} \Gamma_i \vdash^{(\left(\sum_{i=1}^{n} m_i\right)+n, \left(\sum_{i=1}^{n} s_i\right)+n)} \lambda x.f_s : [N_i \multimap M_i]_{i=1}^{n}}\ \mathsf{many}_{\lambda}$$

where $\Gamma = \biguplus_{i=1}^{n} \Gamma_i$ and $M = [N_i \multimap M_i]_{i=1}^{n}$. Since t is not a strong inert term, then we only prove Point 2, for which we assume that M is shrinking, in turn implying that $M \neq \mathbf{0}$ and $n \geq 1$.
By *i.h.* (2) on Φ_i, we have that $|f_s|_{\mathrm{S}} \leq |M_i| + |(\Gamma_i; \hat{x} : N_i)| = |M_i| + |\hat{\Gamma_i}| + |N_i|$.
We then have that

$$\begin{array}{rcl} |t| & = & |f_s| + 1 \\ & \leq & n(|f_s| + 1) \\ & \leq & \sum_{i=1}^{n} \Big(|M_i| + |\hat{\Gamma_i}| + |N_i| + 1\Big) \\ & = & \sum_{i=1}^{n} (|N_i \multimap M_i|) + \sum_{i=1}^{n} \Big(|\hat{\Gamma_i}|\Big) \\ & = & |M| + |\hat{\Gamma}| \end{array}$$

□

(Click here to go back to main chapter.)

Proposition 13.8.18 (Strong CbV-normal forms have a minimal unitary shrinking type)**.**
Let $t \in \Lambda_{\mathsf{L}}$ be in $\rightarrow_{\mathsf{s}}$-normal form.

1. Inert*: If t is a strong inert term, then for every co-shrinking (resp. unitary co-shrinking) multi type M there exists a type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ such that Γ is a co-shrinking (resp. unitary co-shrinking) type context and $|M| + |t|_{\mathrm{S}} = |\hat{\Gamma}|$.*
2. Fireball*: If t is a strong fireball, then there exists a unitary shrinking derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ such that $m = |t|_{\mathrm{S}} = |M| + |\hat{\Gamma}|$. Moreover,*

$$\begin{aligned} m \;=\; \min\{m' \mid \exists\, \Psi \rhd_{\mathrm{S}} \Pi \vdash^{(m',s')} t : N,\ &\textit{with } \Pi \textit{ co-shrinking} \\ &\textit{and if } t \textit{ is an answer then } N \textit{ is shrinking}\} \end{aligned}$$

Proof. (Click here to go back to main chapter.)

First of all, we do not prove that

$$m \;=\; \min\{m' \mid \exists\, \Psi \rhd_{\mathrm{S}} \Pi \vdash^{(m',s')} t : N, \text{ with } \Pi \text{ co-shrinking and if } t \text{ is an answer then } N \text{ is shrinking}\}$$

because it is a direct consequence of all the other facts in the statement, together with Proposition 11.2.2 (Typing properties of Strong CbV-normal forms). Point 1 and Point 2—except for the part on m being the minimum of such set—is proven by mutual induction on the definition of strong inert terms and of strong fireballss. Cases:

- *Variable*: Let $t := x \in \mathsf{Var}$. Let M be co-shrinking, with Φ of the following form

$$\dfrac{\left(\dfrac{}{x : [L_i] \vdash^{(0,1)} x : L_i}\ \mathsf{ax}\right)_{i=1}^{n}}{x : [L_i]_{i=1}^{n} \vdash^{(0,n)} x : [L_i]_{i=1}^{n}}\ \mathsf{many}_{\mathsf{VAR}}$$

  where $\Gamma = \{x : M\}$ and $M = [L_i]_{i=1}^{n}$. Since $|t|_{\mathrm{S}} = 0$, we have that $|M| + |t|_{\mathrm{S}} = |M| = |\hat{\Gamma}|$.
  Now, note that the shrinkingness (resp. unitary shrinkingness) of M implies that of Γ. hence, if we takes $M := [X]$, then Φ is a unitary shrinking type derivation satisfying that

$$|t|_{\mathrm{S}} = 0 = |M| + |\hat{\Gamma}|$$

- *Inert application*: Let $t := i_s f_s$. Note that $|t|_{\mathrm{S}} = |i_s|_{\mathrm{S}} + |f_s|_{\mathrm{S}} + 1$. First, let us define $M := [X]$. By *i.h.* (2) on f_s, there exists unitary shrinking type derivation $\Phi_2 \rhd\ \Gamma_2 \vdash^{(m_2,s_2)} f_s : N$ such that Γ_2 is unitary co-shrinking, N is unitary shrinking, and $m_2 = |f_s|_{\mathrm{S}} = |N| + |\hat{\Gamma}|$.
  Next, note that since N is unitary shrinking and M is unitary co-shrinking, then $[N \multimap M]$ is unitary co-shrinking. By *i.h.* (1) on i_s with respect to $[N \multimap M]$, there exists type derivation $\Phi_1 \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_1,s_1)} i_s : [N \multimap M]$ such that Γ_1 is unitary co-shrinking and $|[N \multimap M]| + |i_s|_{\mathrm{S}} = |\Gamma_1|$.
  We may then derive Φ

$$\dfrac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_1,s_1)} i_s : [N \multimap M] \quad \Phi_2 \rhd\ \Gamma_2 \vdash^{(m_2,s_2)} f_s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,s_1+s_2)} i_s f_s : M}\ \mathsf{app}$$

  noting that
  1.

$$\begin{aligned} |M| + |t|_{\mathrm{S}} &= |M| + |i_s|_{\mathrm{S}} + |f_s|_{\mathrm{S}} + 1 \\ &= |M| + (|\hat{\Gamma_1}| - |[N \multimap M]|) + (|N| + |\hat{\Gamma_2}|) + 1 \\ &= |M| + |\hat{\Gamma_1}| - (|N| + |M| + 1) + |N| + |\hat{\Gamma_2}| + 1 \\ &= |\hat{\Gamma_1}| + |\hat{\Gamma_2}| \\ &= |\hat{\Gamma}| \end{aligned}$$

  and that

2.

$$\begin{array}{rcl} |t|_{\mathrm{S}} & = & |i_s|_{\mathrm{S}} + |f_s|_{\mathrm{S}} + 1 \\ & = & (|\hat{\Gamma_1}| - |[N \multimap M]|) + (|N| + |\hat{\Gamma_2}|) + 1 \\ & = & |\hat{\Gamma_1}| - (|N| + |M| + 1) + |N| + |\hat{\Gamma_2}| + 1 \\ & = & |\hat{\Gamma_1}| + |\hat{\Gamma_2}| + |M| \\ & = & |\hat{\Gamma}| + |M| \end{array}$$

- *Explicit substitution on inert*: Let $t \coloneqq i_s[x \leftarrow j_s]$. Note that $|t|_{\mathrm{S}} = |i_s|_{\mathrm{S}} + |j_s|_{\mathrm{S}}$.
  By *i.h.* (1) on i_s with respect to co-shrinking multi type M, there exists unitary shrinking type derivation $\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} i_s : M$ such that Γ_1 is a co-shrinking type context, and that $|M| + |i_s|_{\mathrm{S}} = |\hat{\Gamma}|$.
  Note that N is co-shrinking. Hence, we can apply *i.h.* (1) on j_s with respect to N, yielding type derivation $\Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} j_s : N$ such that Γ_2 is a co-shrinking type context, and $|N| + |j_s|_{\mathrm{S}} = |\hat{\Gamma_2}|$.
  We may then derive Φ as follows

$$\frac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} i_s : M \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} j_s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,s_1+s_2+1)} i_s[x \leftarrow j_s] : M}\ \mathrm{ES}$$

  noting that
  1.

$$\begin{array}{rcl} |M| + |t|_{\mathrm{S}} & = & |M| + |i_s|_{\mathrm{S}} + |f_s|_{\mathrm{S}} \\ & = & |M| + (|\Gamma_1; \hat{x} : N| - |M|) + (|\hat{\Gamma_2}| - |N|) + 1 \\ & = & |M| + |\hat{\Gamma_1}| + |N| - |N| - |M| - 1 + |N| + |\hat{\Gamma_2}| + 1 \\ & = & |\hat{\Gamma_1}| + |\hat{\Gamma_2}| \\ & = & |\hat{\Gamma}| \end{array}$$

  2. Finally, if we take Φ_1 to be unitary shrinking, then Φ_1 is such that $|i_s|_{\mathrm{S}} = |M| + |(\Gamma_1; \hat{x} : N)|$. Then,

$$\begin{array}{rcl} |t|_{\mathrm{S}} & = & |i_s|_{\mathrm{S}} + |j_s|_{\mathrm{S}} \\ & = & (|M| + |\Gamma_1; \hat{x} : N|) + (|\hat{\Gamma_2}| - |N|) \\ & = & |M| + |\hat{\Gamma_1}| + |N| + |\hat{\Gamma_2}| - |N| \\ & = & |M| + |\hat{\Gamma_1}| + |\hat{\Gamma_2}| \\ & = & |M| + |\hat{\Gamma}| \end{array}$$

- *Abstraction*: Let $f_s \coloneqq \lambda x.g_s$. By *i.h.* (2) on f_s, there exists unitary shrinking type derivation $\Psi \rhd_{\mathrm{S}} \Pi; x : N \vdash^{(m',s')} f_s : O$ such that $m' = |f|_{\mathrm{S}} = |O| + |\Pi; \hat{x} : N|$. We derive $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} \lambda x.f_s : M$ as follows

$$\dfrac{\dfrac{\Psi \rhd_{\mathrm{S}} \Pi; x : N \vdash^{(m',s')} f_s : O}{\Pi \vdash^{(m'+1,s'+1)} f_s : N \multimap O}\ \mathsf{fun}}{\Pi \vdash^{(m'+1,s'+1)} f_s : [N \multimap O]}\ \mathsf{many}_\lambda$$

  noting that

$$\begin{array}{rcl} |t|_{\mathrm{S}} & = & |f_s|_{\mathrm{S}} + 1 \\ & = & |O| + |\Pi; \hat{x} : N| + 1 \\ & = & |O| + |\hat{\Pi}| + |N| + 1 \\ & = & |[N \multimap O]| + |\hat{\Pi}| \end{array}$$

- *Explicit substitution on fireball*: Let $t \coloneqq f_s[x{\leftarrow}i_s]$. Note that $|t| = |f_s| + |i_s|$. Since t is a non-inert strong fireball, we only prove Point 2:
  By *i.h.* (2) on f_s, there exists unitary shrinking type derivation $\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} f_s : M$ such that $m_1 = f_s = |M| + |\Gamma_1; \hat{x} : N| = |M| + |\hat{\Gamma_1}| + |N|$.
  Note that N is unitary co-shrinking. By *i.h.* (1) on i_s with respect to N, there exists $\Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} i_s : N$ such that Γ_2 is unitary co-shrinking and $|N| + |i_s|_{\mathrm{S}} = |\hat{\Gamma}|$.
  We may then derive $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ as follows

$$\frac{\Phi_1 \rhd_{\mathrm{S}} \Gamma_1; x : N \vdash^{(m_1,s_1)} f_s : M \quad \Phi_2 \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} i_s : N}{\Gamma_1 \uplus \Gamma_2 \vdash^{(m_1+m_2,s_1+s_2+1)} f_s[x{\leftarrow}i_s] : M}\ \mathrm{ES}$$

  noting that

$$\begin{array}{rcl} |t|_{\mathrm{S}} & = & |f_s|_{\mathrm{S}} + |i_s|_{\mathrm{S}} \\ & = & (|M| + |\Gamma_1; \hat{x} : N|) + (|\hat{\Gamma_2}| - |N|) \\ & = & |M| + |\hat{\Gamma_1}| + |N| + |\hat{\Gamma_2}| - |N| \\ & = & |M| + |\hat{\Gamma_1}| + |\hat{\Gamma_2}| \\ & = & |M| + |\hat{\Gamma}| \end{array}$$

□

(Click here to go back to main chapter.)

13.8.4 Types and structural equivalence in the VSC

Proposition 13.8.19 (Structural equivalence preserves typability and indices)**.**
Let $t, u \in \Lambda_{\mathrm{L}}$ be such that $t \equiv t'$.
Then there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ if and only if there exists type derivation $\Phi' \rhd \Gamma \vdash^{(m,s)} t' : M$.

Proof. (Click here to go back to main chapter.)
By structural induction on the context C such that $t = C\langle u\rangle \equiv C\langle u'\rangle = t'$. In all cases, showing that the indices of Φ and Ψ are the same is trivial, and is hence not explicitly done. Moreover, note that by symmetry of $\equiv$, it is enough to prove only one implication of the equivalence. Cases of C:

- *Empty context*: Let $C \coloneqq \langle\cdot\rangle$. There are several sub-cases to this:
  - Let $t = u = s[y{\leftarrow}m][z{\leftarrow}\tilde{t}] \equiv_{\mathrm{com}} s[z{\leftarrow}\tilde{t}][y{\leftarrow}m] = u' = t'$, with $y \notin \mathsf{fv}(\tilde{t})$ and $z \notin \mathsf{fv}(m)$. By α-equivalence, we may also assume that $y \notin \mathsf{fv}(m)$ and $z \notin \mathsf{fv}(\tilde{t})$.Then Φ is of the form

$$\dfrac{\dfrac{\Psi \rhd_{\mathrm{S}} \Gamma_1; y : N; z : O \vdash^{(m_1,s_1)} s : M \quad \Theta \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} m : N}{(\Gamma_1 \uplus \Gamma_2); z : O \vdash^{(m_1+m_2,s_1+s_2+1)} s[y{\leftarrow}m] : M}\ \mathrm{ES} \quad \Xi \rhd_{\mathrm{S}} \Gamma_3 \vdash^{(m_3,s_3)} \tilde{t} : O}{\Gamma_1 \uplus \Gamma_2 \uplus \Gamma_3 \vdash^{(m_1+m_2+m_3,s_1+s_2+s_3+2)} s[y{\leftarrow}m][z{\leftarrow}\tilde{t}] : M}\ \mathrm{ES}$$

    Note that $\Gamma_2(y) = \Gamma_2(z) = \mathbf{0}$, by Lemma 11.2.1 (Relevance of the Strong CbV type) and the fact that $y, z \notin \mathsf{fv}(m)$. Similarly, $\Gamma_3(y) = \Gamma_3(z) = \mathbf{0}$.
    We may then derive Φ' as follows

$$\dfrac{\dfrac{\Psi \rhd_{\mathrm{S}} \Gamma_1; y : N; z : O \vdash^{(m_1,s_1)} s : M \quad \Xi \rhd_{\mathrm{S}} \Gamma_3 \vdash^{(m_3,s_3)} \tilde{t} : O}{(\Gamma_1 \uplus \Gamma_3); y : N \vdash^{(m_1+m_3,s_1+s_3+1)} s[z{\leftarrow}\tilde{t}] : M}\ \mathrm{ES} \quad \Theta \rhd_{\mathrm{S}} \Gamma_2 \vdash^{(m_2,s_2)} m : N}{\Gamma_1 \uplus \Gamma_2 \uplus \Gamma_3 \vdash^{(m_1+m_2+m_3,s_1+s_2+s_3+2)} s[z{\leftarrow}\tilde{t}][y{\leftarrow}m] : M}\ \mathrm{ES}$$

– Let $t = u = s(m[y{\leftarrow}\tilde{t}]) \equiv_{@\mathrm{r}} (sm)[y{\leftarrow}\tilde{t}] = u' = t'$, where $y \notin \mathsf{fv}(s)$. By α-equivalence, we may assume that $y \notin \mathsf{fv}(\tilde{t})$. Then Φ must be of the following form

$$\dfrac{\Psi \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_3,s_3)} s : [N \multimap M] \qquad \dfrac{\Theta \rhd_{\mathrm{S}} \Gamma_2; y : O \vdash^{(m_1,s_1)} m : N \quad \Xi \rhd_{\mathrm{S}} \Gamma_3 \vdash^{(m_2,s_2)} \tilde{t} : O}{\Gamma_2 \uplus \Gamma_3 \vdash^{(m_1+m_2,s_1+s_2+1)} m[y{\leftarrow}\tilde{t}] : N}\ \mathsf{ES}}{\Gamma_1 \uplus \Gamma_2 \uplus \Gamma_3 \vdash^{(m_1+m_2+m_3+1,s_1+s_2+s_3+2)} s(m[y{\leftarrow}\tilde{t}]) : M}\ \mathsf{app}$$

noting that $\Gamma_1(y) = \Gamma_3(y) = \mathbf{0}$—by Lemma 11.2.1 (Relevance of the Strong CbV type). We may then derive Φ' as follows:

$$\dfrac{\dfrac{\Psi \rhd_{\mathrm{S}} \Gamma_1 \vdash^{(m_3,s_3)} s : [N \multimap M] \quad \Theta \rhd_{\mathrm{S}} \Gamma_2; y : O \vdash^{(m_1,s_1)} m : N}{(\Gamma_1 \uplus \Gamma_2); y : O \vdash^{(m_1+m_3+1,s_1+s_3+1)} sm : N}\ \mathsf{app} \qquad \Xi \rhd_{\mathrm{S}} \Gamma_3 \vdash^{(m_2,s_2)} \tilde{t} : O}{\Gamma_1 \uplus \Gamma_2 \uplus \Gamma_3 \vdash^{(m_1+m_2+m_3+1,s_1+s_2+s_3+2)} (sm)[y{\leftarrow}\tilde{t}]) : M}\ \mathsf{ES}$$

– The case where $t = u = (s[y{\leftarrow}m])\tilde{t} \equiv_{@\mathrm{l}} (s\tilde{t})[y{\leftarrow}m] = u' = t'$, with $y \notin \mathsf{fv}(\tilde{t})$, follows a similar reasoning to the previous case and is left for the reader.

– Let $t = u = s[y{\leftarrow}m[z{\leftarrow}\tilde{t}]] \equiv_{[\cdot]} s[y{\leftarrow}m][z{\leftarrow}\tilde{t}] = u' = t'$, with $z \notin \mathsf{fv}(s)$. By α-equivalence, we may assume that $y \notin \mathsf{fv}(m)$ and $y, z \notin \mathsf{fv}(\tilde{t})$. Then Φ must be of the following form

$$\dfrac{\Psi \rhd_{\mathrm{S}} \Gamma_1; y : N \vdash^{(m_1,s_1)} s : M \qquad \dfrac{\Theta \rhd_{\mathrm{S}} \Gamma_2; z : O \vdash^{(m_2,s_2)} m : N \quad \Xi \rhd_{\mathrm{S}} \Gamma_3 \vdash^{(m_3,s_3)} \tilde{t} : O}{\Gamma_2 \uplus \Gamma_3 \vdash^{(m_2+m_3,s_2+s_3+1)} m[z{\leftarrow}\tilde{t}] : N}\ \mathsf{ES}}{\Gamma_1 \uplus \Gamma_2 \uplus \Gamma_3 \vdash^{(m_1+m_2+m_3,s_1+s_2+s_3+2)} s[y{\leftarrow}m[z{\leftarrow}\tilde{t}]] : M}\ \mathsf{ES}$$

noting that $\Gamma_1(z) = \Gamma_2(y) = \Gamma_3(y) = \Gamma_3(z) = \mathbf{0}$, by Lemma 11.2.1 (Relevance of the Strong CbV type system).
We may then derive Φ' as follows

$$\dfrac{\dfrac{\Psi \rhd_{\mathrm{S}} \Gamma_1; y : N \vdash^{(m_1,s_1)} s : M \quad \Theta \rhd_{\mathrm{S}} \Gamma_2; z : O \vdash^{(m_2,s_2)} m : N}{(\Gamma_1 \uplus \Gamma_2); z : O \vdash^{(m_1+m_2,s_1+s_2+1)} s[y{\leftarrow}m] : M}\ \mathsf{ES} \qquad \Xi \rhd_{\mathrm{S}} \Gamma_3 \vdash^{(m_3,s_3)} \tilde{t} : O}{\Gamma_1 \uplus \Gamma_2 \uplus \Gamma_3 \vdash^{(m_1+m_2+m_3,s_1+s_2+s_3+2)} s[y{\leftarrow}m][z{\leftarrow}\tilde{t}] : M}\ \mathsf{ES}$$

– The case where $t = u = u[y{\leftarrow}m][z{\leftarrow}\tilde{t}] \equiv_{[\cdot]} s[y{\leftarrow}m[z{\leftarrow}\tilde{t}]] = u' = t'$, with $z \notin \mathsf{fv}(s)$, follows a similar reasoning to the previous case and is left for the reader.

- *Non-empty context*: For every case where $C \neq \langle\cdot\rangle$, the statement follows easily by application of *i.h.*. The details are left for the reader

□

(Click here to go back to main chapter.)

13.8.5 A semantical proof of VSC-normalization via Strong CbV

Proposition 13.8.20 (Qualitative Subject Reduction for VSC)**.**
Let $t, u \in \Lambda_{\mathsf{L}}$ be such that $t \to_{\mathsf{vsc}} u$, and let $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ be a type derivation. Then there exists type derivation $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} u : M$ such that $m' \leq m$ and $s' \leq s$.

Proof. *(Click here to go back to main chapter.)*
By induction on the evaluation context C in the step $t = C\langle s\rangle \to_{\mathsf{vsc}} C\langle s'\rangle = u$, with $s \mapsto_{\mathsf{m}} s'$ or $s \mapsto_{\mathsf{e}} s'$. The proof is analogous to the one for Lemma 11.2.4 (Open Quantitative Subject Reduction for Strong CbV), except that now there is one more case:

- *Abstraction*: Let $C \coloneqq \lambda x.C'$. That is, $t = C\langle s\rangle = \lambda x.C'\langle s\rangle \to_a \lambda x.C'\langle s'\rangle = C\langle s'\rangle = u$ with $s \mapsto_a s'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Then, $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ must be of the following form:

$$\dfrac{\left(\dfrac{\Psi_i \rhd_{\mathrm{S}} \Gamma_i; x : N_i \vdash^{(m_i,s_i)} C'\langle s\rangle : M_i}{\Gamma_i \vdash^{(m_i+1,s_i+1)} \lambda x.C'\langle s\rangle : N_i \multimap M_i}\ \mathsf{fun}\right)_{i=1}^{n}}{\biguplus_{i=1}^{n} \Gamma_i \vdash^{\left(\left(\sum_{i=1}^{n} m_i\right)+n, \left(\sum_{i=1}^{n} s_i\right)+n\right)} \lambda x.C'\langle s\rangle : [N_i \multimap M_i]_{i=1}^{n}}\ \mathsf{many}_\lambda$$

  where $\Gamma = \biguplus_{i=1}^{n} \Gamma_i$ and $M = [N_i \multimap M_i]_{i=1}^{n}$.
  By *i.h.* on Ψ_i for all $1 \leq i \leq n$, there exist $\Theta_i \rhd_{\mathrm{S}} \Gamma_i; x : N_i \vdash^{(m'_i,s'_i)} C'\langle s'\rangle : M_i$ such that $m'_i \leq m_i$ and $s'_i \leq s_i$.
  We may then derive $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} u : M$ as follows

$$\dfrac{\left(\dfrac{\Theta_i \rhd_{\mathrm{S}} \Gamma_i; x : N_i \vdash^{(m'_i,s'_i)} C'\langle s'\rangle : M_i}{\Gamma_i \vdash^{(m'_i+1,s'_i+1)} \lambda x.C'\langle s'\rangle : N_i \multimap M_i}\ \mathsf{fun}\right)_{i=1}^{n}}{\biguplus_{i=1}^{n} \Gamma_i \vdash^{\left(\left(\sum_{i=1}^{n} m'_i\right)+n, \left(\sum_{i=1}^{n} s'_i\right)+n\right)} \lambda x.C'\langle s'\rangle : [N_i \multimap M_i]_{i=1}^{n}}\ \mathsf{many}_\lambda$$

□

(Click here to go back to main chapter.)

Proposition 13.8.21 (Qualitative Subject Expansion for VSC)**.**
Let $t, u \in \Lambda_{\mathsf{L}}$ be such that $t \to_{\mathsf{vsc}} u$, and let $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} u : M$ be a type derivation. Then there exists type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ such that $m' \leq m$ and $s' \leq s$.

Proof. (Click here to go back to main chapter.)
By induction on the evaluation context C in the step $t = C\langle s\rangle \to_{\mathsf{vsc}} C\langle s'\rangle = u$, with $s \mapsto_{\mathsf{m}} s'$ or $s \mapsto_{\mathsf{e}} s'$. The proof is analogous to the one for Lemma 11.2.9 (Open Quantitative Subject Expansion for Strong CbV), except that now there is one more case:

- *Abstraction*: Let $C \coloneqq \lambda x.C'$. That is, $t = C\langle s\rangle = \lambda x.C'\langle s\rangle \to_a \lambda x.C'\langle s'\rangle = C\langle s'\rangle = u$ with $s \mapsto_a s'$ and $a \in \{\mathsf{m}, \mathsf{e}\}$. Then, $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} u : M$ must be of the following form:

$$\dfrac{\left(\dfrac{\Psi_i \rhd_{\mathrm{S}} \Gamma_i; x : N_i \vdash^{(m'_i,s'_i)} C'\langle s'\rangle : M_i}{\Gamma_i \vdash^{(m'_i+1,s'_i+1)} \lambda x.C'\langle s'\rangle : N_i \multimap M_i}\ \mathsf{fun}\right)_{i=1}^{n}}{\biguplus_{i=1}^{n} \Gamma_i \vdash^{\left(\left(\sum_{i=1}^{n} m'_i\right)+n, \left(\sum_{i=1}^{n} s'_i\right)+n\right)} \lambda x.C'\langle s'\rangle : [N_i \multimap M_i]_{i=1}^{n}}\ \mathsf{many}_\lambda$$

  where $\Gamma = \biguplus_{i=1}^{n} \Gamma_i$ and $M = [N_i \multimap M_i]_{i=1}^{n}$.
  By *i.h.* on Ψ_i for all $1 \leq i \leq n$, there exist $\Theta_i \rhd_{\mathrm{S}} \Gamma_i; x : N_i \vdash^{(m_i,s_i)} C'\langle s\rangle : M_i$ such that $m_i \geq m'_i$ and $s_i \geq s'_i$.
  We may then derive $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(m,s)} t : M$ as follows

$$\dfrac{\left(\dfrac{\Theta_i \rhd_{\mathrm{S}} \Gamma_i; x : N_i \vdash^{(m_i,s_i)} C'\langle s\rangle : M_i}{\Gamma_i \vdash^{(m_i+1,s_i+1)} \lambda x.C'\langle s\rangle : N_i \multimap M_i}\ \mathsf{fun}\right)_{i=1}^{n}}{\biguplus_{i=1}^{n} \Gamma_i \vdash^{\left(\left(\sum_{i=1}^{n} m_i\right)+n, \left(\sum_{i=1}^{n} s_i\right)+n\right)} \lambda x.C'\langle s\rangle : [N_i \multimap M_i]_{i=1}^{n}}\ \mathsf{many}_\lambda$$

□

(Click here to go back to main chapter.)

Theorem 13.8.22 ($\to_{\mathsf{s}}$ is normalizing)**.**
Let $t \in \Lambda_{\mathsf{L}}$. If there exists a reduction sequence $d : t \to^*_{\mathsf{vsc}} u$ for some u in $\to_{\mathsf{vsc}}$-normal form, then there exists $d' : t \to^*_{\mathsf{s}} u$.

Proof. *(Click here to go back to main chapter.)*
Since $\to_{\mathsf{s}}$ is a sub-relation of $\to_{\mathsf{vsc}}$, then every $\to_{\mathsf{vsc}}$-normal form u has a unitary shrinking type derivation $\Phi \rhd_{\mathrm{S}} \Gamma \vdash^{(|u|_{\mathrm{S}},s)} u : M$, by Proposition 11.2.7 (Shrinking typability of Strong CbV-normal forms).

Next, by repeated applications of Proposition 11.3.2 (Qualitative Subject Expansion for VSC) iterated along $t \to^*_{\mathsf{vsc}} u$, we get unitary shrinking type derivation $\Psi \rhd_{\mathrm{S}} \Gamma \vdash^{(m',s')} t : M$ such that $m' \geq |u|_{\mathrm{S}}$ and $s' \geq s$.

Moreover, by Theorem 11.2.6 (Shrinking Correctness for Strong CbV), there exists $d : t \to^*_{\mathsf{s}} s$ with s in $\to_{\mathsf{s}}$-normal form.

Now, s is a super strong fireball—by Proposition 10.3.2 (Fullness of Strong CbV)—and so s is also in $\to_{\mathsf{vsc}}$-normal form—by Proposition 10.2.3 (Syntactic characterization of VSC-normal forms). Theorem 10.2.2 (Confluence of $\to_{\mathsf{vsc}}$) finally gives that $s = u$.

□

(Click here to go back to main chapter.)

Bibliography

[Aba+91] Martín Abadi, Luca Cardelli, Pierre-Louis Curien, and Jean-Jacques Lévy. "Explicit Substitutions". In: *J. Funct. Program.* 1.4 (1991), pp. 375–416.

[Acc11] Beniamino Accattoli. "Jumping around the box: graphical and operational studies on λ-calculus and Linear Logic". PhD thesis. *La Sapienza* University of Rome, Feb. 2011.

[Acc12] Beniamino Accattoli. "An Abstract Factorization Theorem for Explicit Substitutions". In: *23rd International Conference on Rewriting Techniques and Applications (RTA'12)*. Vol. 15. LIPIcs. 2012, pp. 6–21. DOI: 10.4230/LIPIcs.RTA.2012.6.

[Acc15] Beniamino Accattoli. "Proof nets and the call-by-value λ-calculus". In: *Theor. Comput. Sci.* 606 (2015), pp. 2–24. DOI: 10.1016/j.tcs.2015.08.006. URL: https://doi.org/10.1016/j.tcs.2015.08.006.

[Acc16] Beniamino Accattoli. "The Useful MAM, a Reasonable Implementation of the Strong λ-Calculus". In: *WoLLIC 2016*. 2016, pp. 1–21.

[Acc18] Beniamino Accattoli. "Proof Nets and the Linear Substitution Calculus". In: *Theoretical Aspects of Computing (ICTAC 2018), 15th International Colloquium*. Vol. 11187. Lecture Notes in Computer Science. 2018, pp. 37–61. DOI: 10.1007/978-3-030-02508-3_3.

[ABM14] Beniamino Accattoli, Pablo Barenbaum, and Damiano Mazza. "Distilling Abstract Machines". In: *Proceedings of the 19th ACM SIGPLAN international conference on Functional programming (ICFP 2014)*. 2014, pp. 363–376. DOI: 10.1145/2628136.2628154.

[Acc+14] Beniamino Accattoli, Eduardo Bonelli, Delia Kesner, and Carlos Lombardi. "A nonstandard standardization theorem". In: *The 41st Annual Symposium on Principles of Programming Languages (POPL '14)*. ACM, 2014, pp. 659–670. DOI: 10.1145/2535838.2535886.

[AD12] Beniamino Accattoli and Ugo Dal Lago. "On the Invariance of the Unitary Cost Model for Head Reduction". In: *RTA'12*. Ed. by Ashish Tiwari. Vol. 15. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012, pp. 22–37.

[AGK20] Beniamino Accattoli, Stéphane Graham-Lengrand, and Delia Kesner. "Tight typings and split bounds, fully developed". In: *J. Funct. Program.* 30 (2020), e14. DOI: 10.1017/S095679682000012X. URL: https://doi.org/10.1017/S095679682000012X.

[AG16] Beniamino Accattoli and Giulio Guerrieri. "Open Call-by-Value". In: *APLAS 2016*. 2016, pp. 206–226.

[AG17] Beniamino Accattoli and Giulio Guerrieri. "Implementing Open Call-by-Value". In: *FSEN 2017, Tehran, Iran, April 26-28, 2017, Revised Selected Papers*. 2017, pp. 1–19.

[AG18] Beniamino Accattoli and Giulio Guerrieri. "Types of Fireballs". In: *Programming Languages and Systems - 16th Asian Symposium, APLAS 2018, Wellington, New Zealand, December 2-6, 2018, Proceedings*. 2018, pp. 45–66. DOI: 10.1007/978-3-030-02768-1_3. URL: https://doi.org/10.1007/978-3-030-02768-1%5C_3.

[AGL19] Beniamino Accattoli, Giulio Guerrieri, and Maico Leberle. "Types by Need (Extended Version)". In: *CoRR* abs/1902.05945 (2019).

[AK10] Beniamino Accattoli and Delia Kesner. "The Structural λ-Calculus". In: *CSL'10*. 2010, pp. 381–395.

[AL16] Beniamino Accattoli and Ugo Dal Lago. "(Leftmost-outermost) Beta Reduction is Invariant, Indeed". In: *LMCS* 12.1 (2016).

[AP12] Beniamino Accattoli and Luca Paolini. "Call-by-value solvability, revisited". In: *FLOPS*. 2012, pp. 4–16.

[AS14] Beniamino Accattoli and Claudio Sacerdoti Coen. "On the Value of Variables". In: *WoLLIC 2014*. 2014, pp. 36–50.

[AS15] Beniamino Accattoli and Claudio Sacerdoti Coen. "On the Relative Usefulness of Fireballs". In: *30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015*. IEEE Computer Society, 2015, pp. 141–155. DOI: 10.1109/LICS.2015.23.

[AKV19] Sandra Alves, Delia Kesner, and Daniel Ventura. "A Quantitative Understanding of Pattern Matching". In: *25th International Conference on Types for Proofs and Programs, TYPES 2019, June 11-14, 2019, Oslo, Norway*. Ed. by Marc Bezem and Assia Mahboubi. Vol. 175. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019, 3:1–3:36. DOI: 10.4230/LIPIcs.TYPES.2019.3. URL: https://doi.org/10.4230/LIPIcs.TYPES.2019.3.

[AF97] Zena M. Ariola and Matthias Felleisen. "The Call-By-Need lambda Calculus". In: *J. Funct. Program.* 7.3 (1997), pp. 265–301.

[Bak92] Steffen van Bakel. "Complete Restrictions of the Intersection Type Discipline". In: *Theor. Comput. Sci.* 102.1 (1992), pp. 135–163. DOI: 10.1016/0304-3975(92)90297-S. URL: https://doi.org/10.1016/0304-3975(92)90297-S.

[Bal+17] Thibaut Balabonski, Pablo Barenbaum, Eduardo Bonelli, and Delia Kesner. "Foundations of strong call by need". In: *PACMPL* 1.ICFP (2017), 20:1–20:29. DOI: 10.1145/3110264.

[Bar84] Hendrik Pieter Barendregt. *The Lambda Calculus – Its Syntax and Semantics*. Vol. 103. North-Holland, 1984.

[Bar99] Bruno Barras. "Auto-validation d'un système de preuves avec familles inductives". PhD thesis. Université Paris 7, 1999.

[BR13] Erika De Benedetti and Simona Ronchi Della Rocca. "Bounding normalization time through intersection types". In: *6th Workshop on Intersection Types and Related Systems (ITRS)* (2013), pp. 48–57.

[BG13] Alexis Bernadet and Stéphane Graham-Lengrand. "Non-idempotent intersection types and strong normalisation". In: *Logical Methods in Computer Science* 9.4 (2013). DOI: 10.2168/LMCS-9(4:3)2013.

[BG95] Guy E. Blelloch and John Greiner. “Parallelism in Sequential Functional Languages”. In: *FPCA’95*. Ed. by John Williams. ACM, 1995, pp. 226–237.

[BCL99] Gérard Boudol, Pierre-Louis Curien, and Carolina Lavatelli. “A semantics for lambda calculi with resources”. In: *Mathematical Structures in Computer Science, 9* (1999), pp. 437–483.

[Bru72] Nicolaas G. de Bruijn. “Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the church-rosser theorem”. In: (1972).

[Bru78] Nicolaas G. de Bruijn. “A namefree lambda calculus with facilities for internal definition of expressions and segments”. In: (1978).

[Bru87] Nicolaas G. de Bruijn. “Generalizing Automath by Means of a Lambda-Typed Lambda Calculus”. In: (1987).

[Buc+20] Antonio Bucciarelli, Delia Kesner, Alejandro Ríos, and Andrés Viso. “The Bang Calculus Revisited”. In: *Functional and Logic Programming - 15th International Symposium, FLOPS 2020, Akita, Japan, September 14-16, 2020, Proceedings*. Ed. by Keisuke Nakano and Konstantinos Sagonas. Vol. 12073. Lecture Notes in Computer Science. Springer, 2020, pp. 13–32. DOI: 10.1007/978-3-030-59025-3_2. URL: https://doi.org/10.1007/978-3-030-59025-3%5C_2.

[BKV17] Antonio Bucciarelli, Delia Kesner, and Daniel Ventura. “Non-idempotent intersection types for the Lambda-Calculus”. In: *Logic Journal of the IGPL* 25.4 (2017), pp. 431–464. DOI: 10.1093/jigpal/jzx018.

[CG14] Alberto Carraro and Giulio Guerrieri. “A Semantical and Operational Account of Call-by-Value Solvability”. In: *Foundations of Software Science and Computation Structures - 17th International Conference (FoSSaCS 2014)*. Vol. 8412. Lecture Notes in Computer Science. 2014, pp. 103–118. DOI: 10.1007/978-3-642-54830-7_7.

[Car07] Daniel de Carvalho. “Sémantiques de la logique linéaire et temps de calcul”. PhD thesis. Université Aix-Marseille II, 2007.

[Car09] Daniel de Carvalho. “Execution Time of lambda-Terms via Denotational Semantics and Intersection Types”. In: *CoRR* abs/0905.4251 (2009).

[Car18] Daniel de Carvalho. “Execution time of λ-terms via denotational semantics and intersection types”. In: *Mathematical Structures in Computer Science* 28.7 (2018), pp. 1169–1203. DOI: 10.1017/S0960129516000396.

[CPT11] Daniel de Carvalho, Michele Pagani, and Lorenzo Tortora de Falco. “A semantic measure of the execution time in linear logic”. In: *Theoretical Computer Science* 412.20 (2011), pp. 1884–1902. DOI: 10.1016/j.tcs.2010.12.017.

[CH00] Pierre-Louis Curien and Hugo Herbelin. “The duality of computation”. In: *ICFP*. 2000, pp. 233–243.

[DM09] Ugo Dal Lago and Simone Martini. “Derivational Complexity Is an Invariant Cost Model”. In: *FOPARA 2009*. 2009, pp. 100–113.

[DR04] Vincent Danos and Laurent Regnier. *Head Linear Reduction*. Tech. rep. 2004.

[DL07] Roy Dyckhoff and Stéphane Lengrand. “Call-by-Value lambda-calculus and LJQ”. In: *J. Log. Comput.* 17.6 (2007), pp. 1109–1134.

[Ehr12] Thomas Ehrhard. "Collapsing non-idempotent intersection types". In: *Computer Science Logic (CSL'12) - 26th International Workshop/21st Annual Conference of the EACSL*. Vol. 16. LIPIcs. 2012, pp. 259–273. DOI: 10.4230/LIPIcs.CSL.2012.259.

[Gar94] Philippa Gardner. "Discovering Needed Reductions Using Type Theory". In: *Theoretical Aspects of Computer Software (TACS '94)*. Vol. 789. Lecture Notes in Computer Science. 1994, pp. 555–574. DOI: 10.1007/3-540-57887-0_115.

[Gir87] Jean-Yves Girard. "Linear Logic". In: *Theoretical Computer Science* 50 (1987), pp. 1–102. DOI: 10.1016/0304-3975(87)90045-4.

[GL02] Benjamin Grégoire and Xavier Leroy. "A compiled implementation of strong reduction". In: *(ICFP '02)*. 2002, pp. 235–246.

[Gue18] Giulio Guerrieri. *Towards a Semantic Measure of the Execution Time in Call-by-Value lambda-Calculus*. to appear in the proceedings of ITRS 2018. 2018.

[HZ09] Hugo Herbelin and Stéphane Zimmermann. "An Operational Account of Call-by-Value Minimal and Classical lambda-Calculus in "Natural Deduction" Form". In: *TLCA*. 2009, pp. 142–156.

[JGS93] N. D. Jones, C. K. Gomard, and P. Sestoft. *Partial Evaluation and Automatic Program Generation*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1993. ISBN: 0-13-020249-5.

[Kes16] Delia Kesner. "Reasoning about call-by-need by means of types". In: *Foundations of Software Science and Computation Structures - 19th International Conference (FOSSACS 2016)*. Vol. 9634. Lecture Notes in Computer Science. 2016, pp. 424–441. DOI: 10.1007/978-3-662-49630-5_25.

[KV20] Delia Kesner and Pierre Vial. "Consuming and Persistent Types for Classical Logic". In: *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*. 2020, pp. 619–632. DOI: 10.1145/3373718.3394774.

[KV21] Delia Kesner and Andrés Viso. "The Power of Tightness for Call-By-Push-Value". In: *CoRR* abs/2105.00564 (2021). arXiv: 2105.00564. URL: https://arxiv.org/abs/2105.00564.

[Kri93] Jean-Louis Krivine. *Lambda-calculus, types and models*. Ellis Horwood series in computers and their applications. Masson, 1993. ISBN: 978-0-13-062407-9.

[Lan64] Peter John Landin. "The Mechanical Evaluation of Expressions". In: *The Computer Journal* 6.4 (Jan. 1964), pp. 308–320.

[Las05] Søren B. Lassen. "Eager Normal Form Bisimulation". In: *20th IEEE Symposium on Logic in Computer Scienc, LICS 2005*. IEEE Computer Society, 2005, pp. 345–354. DOI: 10.1109/LICS.2005.15.

[Mar+99] John Maraist, Martin Odersky, David N. Turner, and Philip Wadler. "Call-by-name, Call-by-value, Call-by-need and the Linear lambda Calculus". In: *Theor. Comput. Sci.* 228.1-2 (1999), pp. 175–210. DOI: 10.1016/S0304-3975(98)00358-2.

[MOW98] John Maraist, Martin Odersky, and Philip Wadler. "The Call-by-Need Lambda Calculus". In: *J. Funct. Program.* 8.3 (1998), pp. 275–317.

[MP94] Gianfranco Mascari and Marco Pedicini. “Head Linear Reduction and Pure Proof Net Extraction”. In: *Theoretical Computer Science* 135.1 (1994), pp. 111–137.

[MPV18] Damiano Mazza, Luc Pellissier, and Pierre Vial. “Polyadic approximations, fibrations and intersection types”. In: *PACMPL* 2.POPL (2018), 6:1–6:28. DOI: 10.1145/3158094.

[Mil06] Robin Milner. “Local bigraphs and confluence: two conjectures”. In: (2006).

[Mog89] Eugenio Moggi. “Computational λ-Calculus and Monads”. In: *LICS '89*. 1989, pp. 14–23.

[Pao01] Luca Paolini. “Call-by-Value Separability and Computability”. In: *ICTCS*. 2001, pp. 74–89.

[PR99] Luca Paolini and Simona Ronchi Della Rocca. “Call-by-value Solvability”. In: *ITA* 33.6 (1999), pp. 507–534.

[Plo75] Gordon D. Plotkin. “Call-by-Name, Call-by-Value and the lambda-Calculus”. In: *Theor. Comput. Sci.* 1.2 (1975), pp. 125–159.

[RP04] Simona Ronchi Della Rocca and Luca Paolini. *The Parametric Lambda Calculus*. Springer Berlin Heidelberg, 2004.

[SF93] Amr Sabry and Matthias Felleisen. “Reasoning about Programs in Continuation-Passing Style”. In: *Lisp and Symbolic Computation* 6.3-4 (1993), pp. 289–360.

[SW97] Amr Sabry and Philip Wadler. “A Reflection on Call-by-Value”. In: *ACM Trans. Program. Lang. Syst.* 19.6 (1997), pp. 916–941.

[SGM02] David Sands, Jörgen Gustavsson, and Andrew Moran. “Lambda Calculi and Linear Speedups”. In: *The Essence of Computation*. 2002, pp. 60–84.

[SP87] Paula Severi and Erik Poll. “Pure type systems with definitions”. In: (1987).

[SE84] Cees F. Slot and Peter van Emde Boas. “On Tape Versus Core; An Application of Space Efficient Perfect Hash Functions to the Invariance of Space”. In: *STOC 1984*. 1984, pp. 391–400.

[Wad71] Christopher P. Wadsworth. “Semantics and pragmatics of the lambda-calculus”. Chapter 4. PhD Thesis. University of Oxford, 1971.

[Yos93] Nobuko Yoshida. “Optimal Reduction in Weak-λ-calculus with Shared Environments”. In: *Proceedings of the conference on Functional programming languages and computer architecture, FPCA 1993, Copenhagen, Denmark, June 9-11, 1993*. Ed. by John Williams. ACM, 1993, pp. 243–254. DOI: 10.1145/165180.165217. URL: https://doi.org/10.1145/165180.165217.

Résumé

Dans cette thèse, on présente une étude *quantitative* du *lambda calcul en appel-par-nécessité*, particulièrement en reliant certaines quantités de sa sémantique opérationnelle avec des quantités extraites des systèmes de types.

En ce qui concerne la théorie de types, on dérive plusieurs systèmes de *multi types*—un variant non-idempotent des types d'intersection. Chacun de ces systèmes cible une certaine définition de l'appel-par-nécessité, de telle façon que le typage équivaut à la normalisation dans la version correspondante de l'appel-par-nécessité. Dans chaque cas, la caractérisation de la normalisation est tellement raffinée que l'on est capable, à partir d'une dérivation de type donnée, d'extraire d'informations quantitatives concernant plusieurs aspects du processus de normalisation. Typiquement, on extraira le nombre de pas de réduction vers la forme normale dans la version de réduction pour laquelle le système a été conçu, ainsi que la taille de cette forme normale.

En ce qui concerne les différents variants de l'appel-par-nécéssité, on définit la sémantique opérationnelle en suivant une approche incrémentale. Le cas de base est la version faible et fermée de l'appel-par-nécessité—le cadre habituel pour modeler le calcul dans les langages de programmation fonctionnelle—formalisée dans un calcul avec des substitutions explicites—le *Linear Substitution Calculus*—permettant un traitement subtil des substitutions.

En visant le cas le plus général—c'est-à-dire le modèle de calcul requis pour les assistants de preuves—on étend le cas de base d'abord vers le cadre de la réduction faible et ouverte—où les termes sont potentiellement ouverts mais la réduction reste faible. On appelle cette stratégie l'*Open CbNeed*.

Ensuite, on présente un variant *utile* de l'Open CbNeed, que l'on appelle *Useful Open CbNeed*. Cette stratégie est conçue comme un variant de l'Open CbNeed où l'on n'effectue que les substitutions d'arguments qui contribuent à la création de nouveaux pas de réduction—les dites *substitutions utiles*—tandis que l'on évite les autres pas de substitutions. De cette manière, l'Useful Open CbNeed réduit les expressions jusqu'à l'obtention de formes normales dite partagées.

L'intérêt pour ce variant utile de l'appel-par-nécessité provient de la théorie de la complexité du lambda calcul, notamment pour la preuve de l'existence d'un modèle de coût de temps *raisonnable* pour le lambda calcul dans le cadre fort—où la réduction peut aller au-dessous des abstractions lambdas. On sait que l'existence d'un tel modèle de coût de temps raisonnable pour le cadre fort requiert la mise en œuvre des substitutions utiles. Plus important encore, elle implique une relation polynomiale entre le lambda calcul fort et les modèles de calcul raisonnable (comme les machines de Turing), en ce qui concerne le nombre de pas de réduction et d'exécution, respectivement.

Jusqu'ici, la littérature ne contient d'analyses sur les substitutions utiles que pour la stratégie d'évaluation leftmost-outermost—la stratégie d'évaluation standard du lambda calcul, appartenant à la famille de l'appel-par-nom. La présentation de l'Useful Open CbNeed est donc la toute première analyse des substitutions utiles pour le lambda calcul en appel-par-nécessité. Présentées comme un élément clé dans la sémantique opérationnelle de l'Useful Open CbNeed, les substitutions utiles sont ensuite représentées dans le système de multi types de l'Useful Open CbNeed comme une simple modification sur le système de l'Open CbNeed.

Finalement, et dans le but de mieux comprendre les relations quantitatives entre les systèmes de multi types et le cadre fort du lambda calcul en appel-par-nécéssité, on dérive une stratégie d'évaluation pour le lambda calcul fort en *appel-par-valeur*, ainsi qu'un système de multi types qui lui correspond. Ce résultat ne contribue pas qu'à la théorie quantitative de l'appel-par-valeur, il concerne aussi à celle de l'appel-par-nécessité en tant qu'il est un pas nécessaire pour la

mise en œuvre d'un variant fort de l'Useful Open CbNeed—ainsi que d'un système de multi types caractérisant sa normalisation et ayant les propriétés quantitatives désirées.

ECOLE DOCTORALE

Titre: Une dissection de l'appel-par-nécessité par la personnalisation des systèmes de multi types

Mots clés: Système de multi types, lambda calcul, appel-par-nécessité, réduction utile

Résumé:
De nombreuses applications technologiques sont modélisées avec le lambda calcul, allant des langages de programmation fonctionnelle aux assistants de preuves.

Dans cette thèse, on part du variant appel-par-nécessité du lambda calcul mis en œuvre dans les langages de programmation fonctionnelles, et procède à l'étendre de plus en plus jusqu'à atteindre le cas le plus général. Chacun de ces variants étend le prédécesseur avec une nouvelle caractéristique, requise pour la preuve de sa complexité. De cette façon, la dernière version de l'appel-par-nécessité a la complexité espérée.

Simultanément, chacun de ces variants est accompagné d'un système de types, où le typage est égal à la terminaison d'exécution dans ce variant. Cette caractérisation de la terminaison par les systèmes de types est tellement serrée que l'on peut extraire d'informations quantitatives pertinentes sur l'exécution des programmes (typiquement, le nombre de pas d'exécution).

Title: Dissecting call-by-need by customizing multi type systems

Keywords: Multi type systems, λ-calculus, call-by-need, useful reduction

Abstract:
The lambda-calculus is used to model the computational content of several technological applications, ranging from the least general case of functional programming languages to the most general one of proof assistants.

We begin by taking the call-by-need variant of the lambda-calculus implemented in functional programming languages and increasingly extend it, targeting the most general one implemented in proof assistants. Each of these versions extends the previous one by adding a single feature, required for proving its complexity behavior. Thus, we end with a well-behaved version of the call-by-need lambda calculus, complexity-wise.

Simultaneously, we develop type systems for each of these variants, such that typability equals termination in the variant. This characterization of termination via type systems is tightened to such an extent that we are able to extract relevant quantitative information about the execution of programs (typically, the number of execution steps).

Institut Polytechnique de Paris
91120 Palaiseau, France