Investigating the expressivity of linear logic subsystems characterizing polynomial time

Matthieu Perrinel

École doctorale 512 : Infomaths

THÈSE

pour obtenir le grade de docteur de l'Université de Lyon délivré par

École Normale Supérieure de Lyon

Discipline: Informatique

présentée et soutenue publiquement par

Matthieu Perrinel

le 02 juillet 2015

Investigating the expressivity of linear logic subsystems characterizing polynomial time

Directeur de thèse : **Patrick Baillot**

Jury

M. Jean-yves Marion,	Professeur, LORIA, Université de Lorraine	Président
M. Patrick Baillot,	Directeur de recherche, LIP, ENS Lyon	Directeur de thèse
M. Simone Martini,	Professeur, Università di Bologna	Rapporteur
M. Ian Mackie,	Chargé de recherche, LIX, Polytechnique	Rapporteur
M. Damiano Mazza,	Chargé de recherche, LIPN, Université Paris 13	Examinateur

Laboratoire de l'Informatique du Parallélisme (LIP)
UMR CNRS 5668, 69007 Lyon, France

THESE

Contents

Abstract

Implicit computational complexity is the characterization of complexity classes by syntactic restrictions on computation models. Several subsystems of linear logic characterizing polynomial time have been defined: these systems are sound (terms normalize in polynomial time) and complete (it is possible to simulate a Turing machine during a polynomial number of steps).

One of the long term goals is to statically prove complexity bounds. This is why we are looking for the most expressive characterizations possible. Our main tool is context semantics: tokens travel across proof-nets (programs of linear logic) according to some rules. The paths defined by these tokens represent the reduction of the proof-net.

Contrary to previous works, we do not directly define subsystems of linear logic. We first define relations $\rightarrow$ on subterms of proof-nets such that: $B \rightarrow C$ means "the number of copies of B depends on the number of copies of C". The acyclicity of $\rightarrow$ allows us to bound the number of copies of any subterm, this bounds the complexity of the term. Then, we define subsystems of linear logic guaranteeing the acyclicity of $\rightarrow$.

We also study characterizations of elementary time and primitive recursive time. In order to adapt our linear logic subsystems to richer languages, we adapt the context semantics to interaction nets, used as a target language for small programming languages. We use this context semantics to define a denotational semantics on interaction nets.

Résumé

La complexité implicite est la caractérisation de classes de complexité par des restrictions syntaxiques sur des modèles de calcul. Plusieurs sous-systèmes de la logique linéaire (*LLL*, *SLL*, L^4,...) caractérisant le temps polynomial ont été définis: ces systèmes sont corrects (l'élimination des coupures normalise en temps polynomial) et complets: pour toute fonction f réalisable en temps polynomial (par ex. le tri de liste) il est possible de simuler une machine de Turing calculant f. Cependant les réseaux de preuves représentant des algorithmes usuels calculant f (par ex. le tri par insertion) ne sont pas tous typables dans ces systèmes.

Un des buts sur le long terme est de donner statiquement des bornes de complexité. C'est pourquoi dans cette thèse nous cherchons à obtenir les caractérisations du temps polynomial les plus expressives possible. Notre principal outil est la sémantique des contextes: des jetons voyagent à travers le réseau selon certaines règles. Les chemins définis par ces jetons représentent la réduction du réseau.

Contrairement aux travaux précédents, nous ne définissons pas directement des sous-systèmes de la logique linéaire. Nous définissons d'abord des relations $\rightarrow$ sur les sous-termes des réseaux de preuves tel que: $B \rightarrow C$ ssi "le nombre de copies de B dépend du nombre de copies de C". L'acyclicité de $\rightarrow$ permet de borner le nombre de copies de chaque sous-terme, donc la complexité du terme. Ensuite nous définissons des sous-systèmes de la logique linéaire assurant l'acyclicité de $\rightarrow$.

Nous étudions aussi des caractérisations du temps élémentaire et des fonction primitives récursives. Fi-

nalement, dans le but d'adapter nos sous-systèmes de la logique linéaire à des langages plus riches, nous adaptons la sémantique des contextes aux réseaux d'interaction, utilisés comme langage cible pour de petits langage de programmation. Nous utilisons cette sémantique des contexte pour définir une sémantique dénotationnelle sur les réseaux d'interactions.

Remerciements

Le premier remerciement revient sans aucun doute à Patrick, toujours disponible au cours de ma thèse. Il m'a lancé sur un sujet sur lequel j'ai pris beaucoup de plaisir, a aiguillé mes recherches, m'a permis de prendre plus de recul sur mes travaux, et m'a aidé les nombreuses fois où j'ai rencontré des difficultés scientifiques ou administratives.

Je voudrais remercier les membres du jury pour avoir accepté d'examiner mes travaux, en particulier les rapporteurs Simone Martini et Ian Mackie pour leur lecture détaillée de ce document.

De nombreux chercheurs ont contribué par leurs remarques à cette thèse et il serait trop long de tous les citer. J'aimerais tout de même spécialement remercier Ugo Dal Lago pour de nombreuses discussions sur la sémantique des contextes, la complexité implicite en général, et pour notre collaboration sur les fonctions primitives récursives. Ainsi que Damiano Mazza: une discussion que j'ai eu avec lui au tout début de cette thèse sur l'expressivité ont réorienté mon sujet de thèse, et les discussions que nous avons eu sur les réseaux d'intéractions ont considérablement amélioré le dernier chapitre de cette document. Enfin, je remercie toute l'équipe de PLUME pour leur aide que ce soit au travers de discussions scientifiques, sur la manière de présenter mes résultats, ou sur le monde de la recherche en général. L'équipe et le laboratoire ne pourraient fonctionner si il n'y avait que des chercheurs. Merci aux secrétaires, notamment Catherine, pour leur réactivité et leur patience tout au long de la thèse.

Pour finir, parce qu'il faut se changer les idées, merci à ceux qui m'ont accompagné pendant ces années: ma famille, mes amis, le foyer et mes colocs qui m'ont supporté pendant ces longs mois de rédaction. Et enfin, merci à ceux qui ont pris le temps de venir à ma soutenance, ou de lire une partie de cette thèse.

Chapter 1

Introduction

1.0.1 Motivation for a Linear Logic subsystem characterizing polynomial time

Motivations for type-systems capturing polynomial time Programming is a notoriously error-prone process. The behaviours of the programs written by programmers on their first attempt often differ from their expected behaviours. This may be because of a typo (as in Figure 1.1a) or because the behaviour of the program is complex. Type systems can detect some of those mistakes so that programmers can correct them more easily. For instance, in Figure 1.1a, OCaml type system notices that the type of `pie` does not match its use in line 5. Thus, the programmer notices that they wrote `pie` instead of `pi` in line 5, and can fix their mistake.

In this thesis, the property we are interested in is time complexity: the time of the execution of a program as a function of the size of its input. A type system S giving a bound on the time complexity of a program would be useful in several ways:

- In some applications, it is very important to have a certified bound on the time of execution. In hard real-time system, programs can never miss a deadline, otherwise the whole system is a failure. For instance, if a pacemaker, a car engine control system, or a missile detection system takes too much time to react to an event, it can cost a life. In these cases, it is not enough to verify that the system reacted fast enough during tests. We want an absolute certainty.

- In complexity theory, the main method to prove that a problem is NP-complete, is to define a polynomial time reduction from another NP-complete problem. If S is well-trusted, it could be used as a specialized proof assistant: the fact that the reduction is typable in S would increase the trust in the proof. Polynomial time reductions are also used in cryptography to prove that a protocol is secure [59]. Such type systems have also been proposed to reason about computational indistinguishability [65]. More generally, S could be used in any proof relying on a complexity bound for a program.

- For some softwares, it seems enough to get an empirical estimate of the complexity by running tests. For example, for a console video game, if players do not have slowness problems during the tests one may suppose that such problems are rare enough to be unimportant. In this case, S could be useful to find the origin of the slowness observed during tests (this requires the type inferrer to give useful information when it fails to type a term). For instance, in the program of Figure 1.1b, one can imagine that the type inferrer would answer "Failure to infer a polynomial-time bound: `hanoi l` makes recursive calls to `hanoi q` and `hanoi q` (line 5 and 5). Inequality $|q| + |q| \leq |l|$ could not be

```
let returnPi () =
  let pie = "Pies are delicious"
  and pi =3.14159
  in pie
in 2. *. returnPi() ;;
```

(a) A (non-compiling) OCaml program

```
let rec hanoi l=
 match l with
 | [] -> 0
 | t:: q ->
   (hanoi q)+1+(hanoi q)
in hanoi [5;4;3;2;1];;
```

(b) This naive program computes the number of moves required to move the stack of disk l to another rod in "Tower of Hanoi" puzzle

```
let rec divide a l=
  match l with
  |[] -> ([],[])
  |h::q ->
    let (smaller,bigger)= divide a q in
    if h < a then
      (h::smaller,bigger )
    else
      (smaller, h::bigger)
in
let rec quicksort l =
  match l with
  | [] -> []
  | h::q ->
     let (q1,q2)= divide h q in
     (quicksort q1)@[h]@(quicksort q)
in
quicksort [0;2;1;5;4]
```

(c) An exponential time OCaml program

Figure 1.1: Those programs do not compile because of a type error.

inferred". Thus, the programmer can easily fix this by replacing line 5 by `let hanoiQ= hanoi q in hanoiQ+1+hanoiQ`

- Even when time complexity is not an issue, if the type system fails to infer a bound on execution time corresponding to what the programmer is expecting, it may suggest an error in the program (in the same way the type system of Ocaml notice the error in the program of Figure 1.1a). For instance, the program of Figure 1.1c is an attempt to program the sorting algorithm Quicksort. The program should normalize in polynomial time, but because the programmer made a mistake (in line 16 it should be `quicksort q2` instead of `quicksort q`), this program normalizes in exponential time. As for the program of Figure 1.1b, one can imagine that the type inferrer would answer "Failure to infer a polynomial-time bound: `quicksort l` makes recursive calls to `quicksort q1` and `quicksort q` (line 16 and 16). Inequality $|q1| + |q| \leq |l|$ could not be inferred". Thus, the programmer would quickly notice their mistake and correct line 16.

In this thesis, we define type systems of λ-calculus (those type systems are refinements of System F [35, 62]), such that every typed term normalizes in polynomial time (resp. elementary time and primitive recursive). This property is called polynomial time *soundness*. And, for every function f computable in polynomial time (resp. elementary time and primitive recursive time) there exists a typed term t_f which computes f. This property is called polynomial time *extensional completeness*. We write that a type system S characterizes a complexity class C if and only if S is both sound and extensionally complete for C.

Expressiveness and decidability Determining if λ-term t normalizes in polynomial time is undecidable, even if we are given a System F type derivation for t. Thus, every type system S characterizing polynomial time is in one of the two following situations:

```
let identite n =
  if n == n then
    n
  else
    while (true) {};
    n
in
identite 5
```

Figure 1.2: This program terminates in polynomial time but not blindly.

- Either determining whether a term t is typable in S is undecidable.
- Or S is not *intensional complete*: it is to say that there exist λ-terms of System F which normalize in polynomial time and are not typed by S.

A system can be interesting even if it is undecidable. Dal Lago and Gaboardi have defined the type system *dlPCF* [22] which characterizes exactly the execution time of *PCF* programs. Type-checking in *dlPCF* is undecidable. However, one can imagine defining a heuristics for type inference, asking the programmer to add annotations to help the type inferrer, or restricting *dlPCF* to a decidable fragment. Their framework can be seen as a top-down approach.

Here we follow instead a bottom-up line of work: we take inspiration from previous decidable type systems characterizing *Ptime* and try to relax conditions without losing neither soundess nor decidability. The type systems S characterizing polynomial time we define are decidable and intensionally incomplete. Let us consider list sorting, because it is computable in polynomial time there exists a turing machine M and a polynomial P such that, when M is applied to the list l, M sorts l in at most $P(|l|)$ steps. We define t_{sort} as a term which, on input l, simulates M during $P(|l|)$ steps. Because t_{sort} is proved to be typable in S, this proves the extensional completeness of S. However, people never program by simulating Turing machines (because it is quite tedious) and λ-terms representing usual sorting algorithms (e.g. insertion sort and quicksort) may be untypable in S. The more intensionally expressive S is (i.e. the more terms are typable by S), the more useful S is. Indeed, the four motivations for systems characterizing polynomial time we described earlier require S to type programs written by non-specialists: people who may not have a thorough understanding of S. Thus, we want S to type as many "natural" programs as possible.

Restriction to λ-calculus This quest for expressivity is quite complex. This is why, rather than defining a type system on C++ or OCaml characterizing polynomial time, we first define a subsystem of System F on λ-calculus. We only focus on one aspect of computation: how, in a higher-order setting, does the application of a function to another cause the complexity to be non-polynomial? and how can we prevent it? Moreover, we do not look into the data, only at their size. For instance, typing the λ-terms corresponding to the program of Figure 1.2 is not in the scope of this thesis. Indeed, the fact that this program normalizes in polynomial time comes from the fact that line 5 is never executed because `n` is always equal to itself. If, in line 2, we replaced the second occurrence of `n` by another object of same type and size, the polynomial time bound would no longer hold.

On the contrary, in the programs of Figure 1.1 (after the respective corrections proposed earlier), the polynomial time bounds can be proved without examining the values of the data. One only needs to consider

their sizes. We thus say that they are blindly polynomial time [6]. In this thesis, we are only interested (for the sake of simplicity) in the λ-terms which are typable in System F and blindly polytime.

Thus we try to define a type system S for λ-calculus which characterizes polynomial time and is as expressive as possible. We consider that, in future works, S may be modified to take into account other features of modern programming languages. Such modifications have already been defined for some type systems on pure λ-calculus. For instance, Baillot and Terui defined a type system *DLAL* characterizing polynomial type in pure λ-calculus [12]. Taking inspiration from this work, Baillot, Gaboardi and Mogbil defined a type system characterizing polynomial time in λ-calculus enriched with constructors for algebraic data-types, pattern matching and recursive definitions [7]. Similarly, Antoine Madet defined a type system characterizing polynomial time in λ-calculus enriched with multithreading and side effects [54]. One could also imagine capturing non-blindly polynomial programs by extending S with (limited) dependent types.

Linear logic and proof-nets Linear logic (*LL*) [36] can be considered as a refinement of System F where we focus especially on how the duplication of formulae is managed. In linear logic, the structural rules (contraction and weakening) are carefully controlled. In System F, $A_1, \cdots, A_n \vdash B$ means that "one can prove B using the hypotheses A_1,...,A_n as many times as needed". In linear logic, $A_1, \cdots, A_n \vdash B$ means that "one can prove B using exactly once every hypothesis in A_1,...,A_n". To retrieve the whole expressivity of linear logic, a modality ! is introduced: $!A$ intuitively means "as many instances of A as needed". Contraction and weakening are only allowed for formulae of the shape $!A$:

$$\frac{\Gamma, !A, !A \vdash B}{\Gamma, !A \vdash B}\ ?C \qquad\qquad \frac{\Gamma \vdash B}{\Gamma, !A \vdash B}\ ?W$$

With the three following additional rules (promotion, dereliction and digging), linear logic is as expressive as System F, so the elimination of the *cut* rule (corresponding to the β-reduction of λ-calculus) is not even primitive recursive because the Ackermann function can be expressed in linear logic (Figure 3.12).

$$\frac{A_1, \cdots, A_n \vdash B}{!A_1, \cdots, !A_k \vdash !B}\ !P \qquad \frac{\Gamma, A \vdash B}{\Gamma, !A \vdash B}\ ?D \qquad \frac{\Gamma, !!A \vdash B}{\Gamma, !A \vdash B}\ ?N$$

However, because the structural rules are handled by 5 distinct rules, one can enforce a subtle control on the utilisation of ressources by modifying one of them. If we restrict some of those rules, it restricts the duplication of formulae (so the duplication of subterms through the proofs-as-programs paradigm). For instance, Danos and Joinet proved in [24] that, in the absence of $?D$ and $?N$ rules, the *cut*-elimination normalizes in elementary time. The set of such proofs is defined as Elementary Linear Logic (*ELL*). Through the proof-as-programs correspondence, subsystems of linear logic enjoying a bound on *cut*-elimination can be transformed into type systems for λ-calculus enforcing a bound on β-reduction [12].

Proof-nets [37] are an alternative syntax for linear logic, where proofs are considered up-to meaningless commutations of rules. Proof-nets are graph-like structures where nodes correspond to logical rules. One of the reasons we use proof-nets instead of proof derivations is that context semantics, the main tool we use in this thesis, is much simpler to define and use in proof-nets (although it was also defined directly on proof derivations [49]).

1.0.2 Specificities of our approach

Previous polynomial time subsystems of Linear logic There already exist several subsystems of linear logic characterizing polynomial time. The first such subsystem is Bounded Linear Logic (*BLL*), defined by Girard, Scedrov and Scott in 1992 [39]. The main mechanism of *BLL* is the labelling of ! modalities by

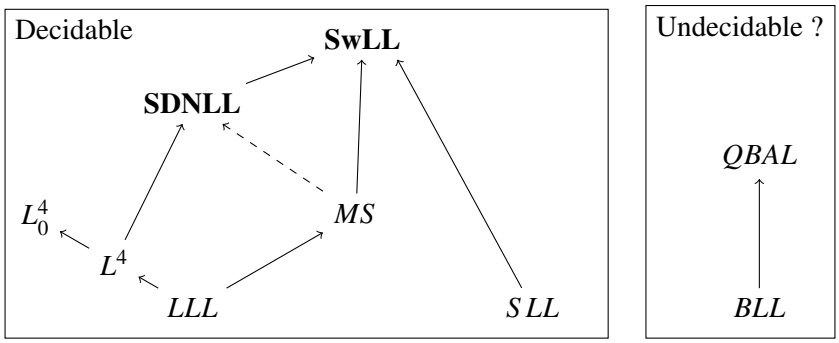


Figure 1.3: State of the art and contributions

polynomials. In 2009, Hofmann and Dal Lago defined Quantified Bounded Affine Logic (*QBAL*) which is a generalization of *BLL*. However, given a proof-net G, determining if G is in *BLL* (resp. *QBAL*) seems undecidable. Moreover, given a λ-term t typed in System F, determining whether there is a proof-net corresponding to t in those systems also seems undecidable. Thus, they do not fit in our approach.

The first decidable system was Light Linear Logic (*LLL*) defined by Girard in 1994 [34, 38]. *LLL* is defined as the proofs of *ELL* such that the contexts have at most one formula in every $!P$ rule[1]. A type system *DLAL* for λ-calculus was inspired by *LLL* [12] and determining whether a λ-term t typed in System F can be typed in *DLAL* is decidable [2]. Type inference for *ELL* has also been the object of various studies [17, 18, 11].

In 2010, Baillot and Mazza generalized *ELL* with a subsystem L^3 of linear logic characterizing elementary time [8]. Then they defined L^4 and L_0^4, characterizing polynomial time, based on L^3 in the same way as *LLL* is based on *ELL*. Those two systems are generalizations of *LLL*. In a separate direction, Roversi and Vercelli also extended *LLL* with *MS*[2] [64] (also in 2010). Those three systems are obtained by decorating formulae with labels and adding local constraints on the labels. One can observe that L^4, L_0^4 and *MS* are trivially decidable on proof-nets: given a proof-net G there exist only a finite number of possible ways to label the formulae of G. One can try every possibility and check whether the labels verify the constraints. And one can conjecture that they can be transformed into decidable type systems for λ-calculus similarly to the transformation between *LLL* and *DLAL*.

Lafont defined another subsystem of linear logic characterizing polynomial time in 2004: Soft Linear Logic [47] (*SLL*). This system does not contain *LLL*, and none of the above generalization of *LLL* contains *SLL*. However, in practice, *SLL* really seems less expressive than *LLL* (and its generalizations). A hint is that (contrary to the *MS* and L_0^4 cases for example) the soundness of *SLL* is "very easy to prove" while the completeness "is more delicate to prove" (according to Lafont [47]). Because of its simplicity, *SLL* inspired many systems to characterize polynomial time in λ-calculus [9, 33], to characterize other classes of complexity [31, 32], and to bound the length of interactions of processes [21, 48].

Figure 1.3 summarizes the state of the art. There is an arrow from the system S to the system T if there is a canonical embedding of S in T. The arrow between *MS* and *SDNLL* is dotted because the canonical embedding is only defined for one of the maximal systems of *MS*. The systems in bold are the systems we define in this thesis.

[1] To keep some expressivity, Girard adds a new modality §.

[2] Which is a set of system rather than a unique system, we give more details in Chapter 5.

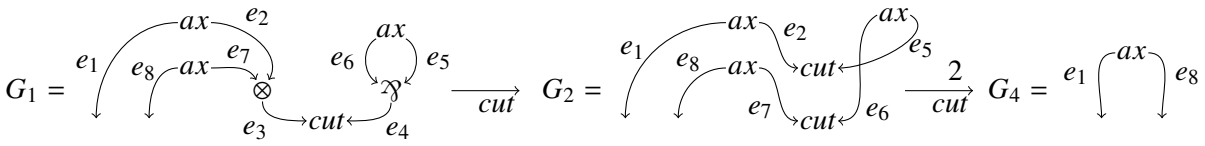


Figure 1.4: Simple example of context semantics:

Context semantics Our main tool is context semantics. Context semantics is a presentation of geometry of interaction [40, 25] defined by tokens traveling across proof-nets according to some rules. The paths defined by those tokens are stable by reduction so they represent the reduction of the proof-net. As a first example, let us consider Figure 1.4. It represents the *cut*-elimination of a proof-net $G_1 \to_{cut} G_2 \to^2_{cut} G_4$. The rules we define in Section 2.2 give us the following path in G_1:

$$((\overline{e_1}, []), T) \mapsto ((e_2, []), T) \mapsto ((e_3, []), T.\otimes_r) \mapsto ((\overline{e_4}, []), T.\otimes_r) \mapsto ((\overline{e_5}, []), T) \mapsto$$
$$((e_6, []), T) \mapsto ((e_4, []), T.\parr_l) \mapsto ((\overline{e_3}, []), T.\parr_l) \mapsto ((\overline{e_7}, []), T) \mapsto ((e_8, []), T)$$

It corresponds to the following path in G_2:

$$((\overline{e_1}, []), T) \mapsto ((e_2, []), T) \mapsto ((\overline{e_5}, []), T) \mapsto ((e_6, []), T) \mapsto ((\overline{e_7}, []), T) \mapsto ((e_8, []), T)$$

And to the following path in G_4:

$$((\overline{e_1}, []), T) \mapsto ((e_8, []), T)$$

Context semantics has first been used to study optimal reduction in λ-calculus [40] and linear logic [41]. It has also been used for the design of interpreters for λ-calculus [52]. Finally, it has been used to prove complexity bounds on subsystems of System T [19] and linear logic [10, 20]. In [20], Dal Lago defines for every proof-net G a weight $W_G \in \mathbb{N} \cup \{\infty\}$ based on the paths of context semantics such that, whenever G reduces to H, $W_G \geq W_H + 1$. Thus W_G is a bound on the length of the longest path of reduction starting from G. Then we can prove theorems of the shape "whenever G satisfies some property (for instance if G belongs to a subsystem such as *LLL*), W_G satisfies some bound (for instance $W_G \leq P(|G|)$ with P a polynomial and $|G|$ the size of G)." From this point of view, context semantics has two major advantages compared to the syntactic study of reduction:

- Its genericity: some common results can be proved for different variants of linear logic, which allows to factor out proofs of complexity results for these various systems.
- It proves strong bounds on reduction: the bounds stand for any strategy of reduction. On the contrary, most bounds proved by syntactic means are only proved for a particular strategy. If the reduction strategy corresponded to strategies frequently used by programming languages (such as left-to-right call-by-value), it would not be a big problem. However, in some cases (L^4 for instance [8]), the strategy is rather farfetched and unlikely to be implemented in a programming language.

Our context semantics, presented in Section 2.2, is slightly different from Dal Lago's context semantics. In particular, Dal Lago worked in intuitionnistic linear logic, and we work in classical linear logic. So the results of [20] can not be directly applied. However most theorems of [20] have correspondents in our framework, with quite similar proofs.

Semantic criteria based on context semantics The existing subsystems refuse many proof-nets whose polynomiality seems straightforward. Thus, one may fear that their extension to real world programming languages will not type a great number of natural polynomial time programs. In this thesis, we push the limits of expressivity of linear logic subsystems characterizing polynomial time.

Contrary to previous works, we do not directly define linear logic subsystems. First, we define semantic criteria forbidding behaviours which can result in non-polynomial complexity. Typically, we define a relation $\to$ on boxes (special subterms of proof-nets) such that $B \to C$ means that "the number of times B is copied depends on the number of times C is copied". Then, the acyclicity of $\to$, ensures a bound on the number of times every subterm is copied so a bound on the length of normalization sequences.

Let us suppose that there exist two relations $\to_1$ and $\to_2$ whose acyclicity entail a bound on the complexity of *cut*-elimination. If we have $\to_1 \subseteq \to_2$ then the acyclicity of $\to_2$ entails the acyclicity of $\to_1$. So the criterion "$\to_1$ must be acyclic" is a generalization of "$\to_2$ must be acyclic": it is true on at least as many proof-nets. This is why we will endeavour to define the smallest relations possible whose acyclicity entails a polynomial bound.

Then (in Chapter 5), we define subsystems of linear logic such that those relations are acyclic on every proof-net of the subsystem. This gives us a bound on the length of normalization for every proof-net of the subsystem. The relations $\to$ we study are based on the paths of context semantics. The rules defining those paths are local, as the typing constraints. We use the typing constraints to define invariants along the paths, proving that if $B \to C$ then the "types" of B and C are such that we can not have $C \to B$.

Stratification, Dependence control, Nesting More precisely, our criteria entailing polynomial time bounds will be composed of three conditions: a stratification condition (which, alone, implies an elementary time bound), a dependence control condition and a nesting condition.

In Section 3.1, as a toy example, we define a relation $\twoheadrightarrow$ whose acyclicity entails an elementary time bound (here, the stratification condition is the acyclicity of $\twoheadrightarrow$). On proof-nets G which have at most one formula in the contexts of their $!P$ rules[3] (dependence control condition) and without $?N$ nodes (nesting condition), it entails a polynomial bound. On proof-nets of ELL, $\twoheadrightarrow$ is acyclic. So this section proves the elementary bound of ELL. Because the LLL proof-nets are ELL proof-nets without $?N$ nodes and whose boxes have at most one auxiliary door, this section proves the polynomial bound of LLL.

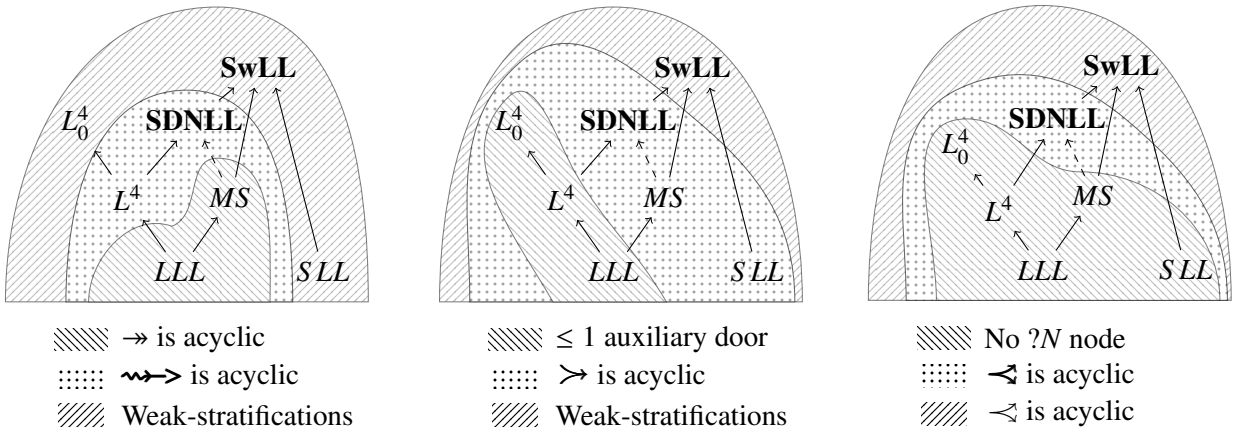
The boxes of the proof-nets of L^4 and L^4_0 also have at most one auxiliary door, so this dependence control condition is general enough for those systems. However, because $\twoheadrightarrow$ is not always acyclic on the proof-nets of those system, this is not enough to prove their polynomial time bounds. Let us notice that L^4, L^4_0, MS and SLL do not have $?N$, so this nesting condition is general enough for every previous decidable subsystem of linear logic characterizing polynomial time.

In Section 3.2, we define a relation $\rightsquigarrow \subseteq \twoheadrightarrow$ whose acyclicity entails an elementary time bound (stratification condition). This relation is acyclic on every proof-net of L^3 (so every proof-net of L^4 and L^4_0), proving the elementary time bound of L^3.

In Section 3.3, we define relations $\succ\!\!\!\to$ and $\prec\!\!\!\vartriangleleft$. The acyclicity of $\succ\!\!\!\to$ is a dependence control condition more general than the condition "at most one auxiliary door by box". In fact, if contexts of $!P$ rules have at most one formula, $\succ\!\!\!\to$ is the empty relation which is trivially acyclic. The relation $\succ\!\!\!\to$ is also acyclic on every proof-net of MS and SLL. The relations $\rightsquigarrow$ and $\succ\!\!\!\to$ were published in [60] where we proved that their acyclicities entail a polynomial time bound in the absence of $?N$ nodes.

The acyclicity of $\prec\!\!\!\vartriangleleft$ is a nesting condition which is a generalization of "no $?N$ nodes": for proof-nets without $\prec\!\!\!\vartriangleleft$ nodes, $\prec\!\!\!\vartriangleleft$ is the empty relation. If $\rightsquigarrow$, $\succ\!\!\!\to$ and $\prec\!\!\!\vartriangleleft$ are acyclic, it entails a polynomial time

[3] It means that boxes have at most one auxiliary door



(a) Stratification on *LL* subsystems (b) Dependence control on *LL* subsystems (c) Nesting on *LL* subsystems

Figure 1.5: Tools needed to prove the polynomial time bounds of *LL* subsystems.

bound. This proves the polynomial bound of L^4 and MS. Because there seems to be lot of room between the condition "⇝, ↣ and ⊰ are acyclic" and the systems LLL, L^4 and MS, we define a new system called *Stratified Dependence controlled Nested Linear Logic* ($SDNLL$). This system is trivially decidable (for any given proof-net, there is a finite number of possible labels). L^4 and a maximal system of MS are trivially embedded in $SDNLL$.

Finally in Section 3.5 we define a condition called "weak stratification" which is a generalization of previous stratification and dependence control conditions. We also define a relation $\prec \subseteq \lessdot$ whose acyclicity is a nesting condition. The stratification condition is general enough for SLL and L_0^4. The situation is summarized in Figure 1.5. Thus, this section proves the polynomial bound of every previous decidable subsystem of linear logic characterizing polynomial time. Because there is a lot of room between the conditions of this section and previously defined systems, we define a new system *Sweetened Linear Logic* ($SwLL$). $SDNLL$ and all the systems of MS are trivially embedded in $SwLL$. It is unclear whether L_0^4 can be embedded in $SwLL$ or not.

Although it is nice to have a single system in which we can embed L^4, MS and SLL, if it was our goal we would be killing a fly with a sledgehammer: $SwLL$ is a lot more expressive than either of those systems. In particular, contrary to previous systems, $SwLL$ contains proof-nets which are not hereditarily polynomial time: they contain subproof-nets which do not normalize in polynomial time. Another interesting feature is that, in some cases, we can iterate a function obtained by iteration. The price we pay is that the definition of $SwLL$ is quite complex. One might argue that such a complex system could not be understood by programmers. But it does not need to be understood, we only need the type inference engine to give meaningful messages when it fails to infer a type for a term. We will argue in Section 5 that, because $SwLL$ is much closer to the set of polynomial time proof-nets than LLL, we can expect the indications of the type inference engine to be much more meaningful.

Iteration of functions obtained by iteration Non-size-increasing (NSI) type systems [1, 43], a line of work prompted by Hofmann [44], were also an important inspiration for this thesis. In previous subsystems of linear logic characterizing polynomial time, one can iterate[4] a function f only if f has a type of the shape $A \multimap A$. And the iteration of f then has type $B \multimap C$ with $B \neq C$. Thus, one can never iterate a function

[4]For instance, to define a function computing $n \mapsto f(\cdots f(f(x))\cdots)$ with n successive applications of f to some term x

obtained by iteration. To understand this restriction, one can consider the function $f : n \mapsto n + 2$. Its iteration is of the shape $g : n \mapsto 2 \cdot n + x$. Thus the iteration of g is of the shape $h : n \mapsto y \cdot \sum_{i=0}^{n}(2^i \cdot x)$. This function h is not computable in polynomial time. However, this restriction forbids many natural algorithms. For instance, insertion sort is usually defined as the iteration of a function obtained by iteration.

In [44] Hofmann noticed that, if the size of $f(n)$ is at most 1 plus the size of n, then one can iterate the iteration of f. For instance, if we consider $f : n \mapsto n+1$, the iteration of f is of the shape $g : n \mapsto n+x$. The iteration of g is of the shape $h : n \mapsto x \cdot n + y$ which is computable in polynomial time. Non-size-increasing type systems discriminate between the two cases by a careful handling of constructors.

In linear logic, whenever we iterate a function obtained by iteration we have $B \twoheadrightarrow B$ and $B \rightsquigarrow B$ with B a box, and $\twoheadrightarrow$ and $\rightsquigarrow$ the relations defined in Sections 3.1 and 3.2. Thus, the iteration of a function obtained by iteration prevents those relations from being acyclic. But the criteria defined in Section 3.5 and the type system *SwLL* of Section 5.3 allow iteration of functions obtained by iteration in some cases.

Characterizing polynomial time in other frameworks Characterizations of polynomial time have also been defined for other frameworks. The first such characterizations were restrictions of recursion [15, 50]. The direction of research defining the most expressive characterizations is the one where programs are term rewriting systems. Polynomial time bounds are enforced by path orderings and quasi-interpretations [16]. They are considered much more expressive than type-systems based on subsystems of linear logic. One of their limitations is that they only deal with first-order functions (although Baillot and Dal Lago recently generalized quasi-interpretations to a framework allowing higher-order functions in a limited way [4, 5]).

With this thesis, we have reduced the gap of expressivity between the interpretation and the linear logic approaches. But we do not view these approaches are competitors, we think that they will be combined. For instance, one can imagine that for every term rewriting system R, with a quasi-interpretation and an ordering of function satisfying the conditions of [16], one could assign *SwLL* types to the constructor and function symbols of R such that *SwLL*, augmented with the constructors and functions of R, still characterizes polynomial time.

1.0.3 Primitive recursion

Dal Lago, Roversi and Vercelli defined a strategy of reduction on proof-nets called "superlazy reduction" [23]. This strategy is not complete: there exist *blocking* proof-nets which are not in normal forms with respect to *cut*-elimination but are normal with respect to superlazy reduction. They prove that this strategy characterizes primitive recursion in the following meaning: superlazy reduction is computable in time bounded by a primitive recursive function, and every primitive recursive function f is representable by a proof-net G_f which does not block (superlazy reduction reduces G_f to a non-blocking proof-net). However, as they wrote: "Unfortunately, we do not even know any criteria allowing to guarantee that certain proof nets can be reduced to normal form (w.r.t. ordinary reduction) by way of superlazy reduction".

After seeing our work on polynomial time, Dal Lago defined a relation $\dot{\twoheadrightarrow}$ and conjectured that: for every proof-net G, if $\dot{\twoheadrightarrow}$ is acyclic, then G does not block. This relation is closely related to the relation $\twoheadrightarrow$ we defined to enforce elementary time: $\twoheadrightarrow$ can be viewed as a projection of $\dot{\twoheadrightarrow}$ (in particular, if $\twoheadrightarrow$ is acyclic then $\dot{\twoheadrightarrow}$ is acyclic). We proved directly (without using the notion of superlazy reduction) that the acyclicity of $\dot{\twoheadrightarrow}$ enforces a primitive recursive bound. In fact, in Section 4, we define a smaller relation $\dot{\rightsquigarrow}$ whose acyclicity entails a primitive recursive bound. In Section 5.2, we define Quantifier Predicative Linear Logic (*QPLL*), a subsystem of linear logic characterizing primitive recursion: $\dot{\rightsquigarrow}$ is acyclic on every *QPLL* proof-net.

1.0.4 Interaction nets

Earlier, we wrote that for the sake of simplicity, we only considered subsystems of linear logic. We think that, with respect to the set of blindly polynomial *LL* proof-nets, $SwLL$ is quite expressive. However, the whole set of *LL* proof-nets (let alone the set of blindly polynomial ones) is not quite expressive, because it is only as expressive as System F. For instance, common algorithms such as merge sort and quicksort are hard to implement in System F/linear logic. Indeed, recursion is usually implemented by the iteration of a function. However, such recursive definitions is very limited. For instance, functions defined by the iteration of a list only make one recursive call. This rules out programs such as the one of Figure 1.1c.

To gain some expressivity, we need to extend linear logic with more primitives: built-in arithmetic, inductive types, pattern matching, less constrained recursion, side-effects, multi-threading, exceptions,... If we extended linear logic with one of the above features, there are risks that we would spend a lot of time just to transpose the basic results of this thesis to this extension. Then, we may need a lot more time to add another feature, or even slightly modify the language, and so on. As the set of features needed is not precisely defined, a general framework of systems would be preferred to a single system. This way, we would need to define the context semantics and prove the general theorems only once, and they will stand for any system of the framework.

The framework we chose is interaction nets: a well-behaved class of graph rewriting systems [45]. Interaction nets are a model of asynchronous deterministic computation. A net is a graph-like structure whose nodes are called *cells*. Each cell is labelled by a symbol. A library defines the set of symbols and the rewriting rules for the symbols. Thus, a library corresponds to a programming language. Interaction nets as a whole, correspond to a set of programming languages. Interaction nets present several major advantages:

- The definition of interaction nets was inspired by the proof-nets of linear logic. Because they have many similarities, it may be relatively easy to transpose the methods of the present thesis to interaction nets (compared to other equally expressive frameworks).

- Interaction nets have been used to encode several systems. Proof-nets [53] and λ-calculus [51] but also functional programming languages containing pattern-matching and built-in recursion [30]. A non-deterministic extension is powerful enough to encode the full π-calculus [55]. This could be used to control ressources of processes as in the SHO_π calculus [48].

Because context semantics is our main tool, our first step will be to extend context semantics to every library of interaction nets. As context semantics is a model of geometry of interaction, the most relevant work is the definition of a geometry of interaction for an arbitrary library by De Falco [27]. De Falco defines a notion of paths in nets and a notion of reduction of those paths. Then, he defines a geometry of interaction of a library as a weighing of paths by elements of a semi-group such that the weights are stable along reduction. However, he exhibits such a semi-group only for some particular libraries (based on linear logic). Thus, there is no complete geometry of interaction model of interaction nets yet.

The most difficult part to define a context semantics on interaction net was to define a notion of tokens. Then, defining the rules of the paths, proving that these paths are stable along reduction, defining a weight $W_G \in \mathbb{N} \cup \{\infty\}$, and proving that it decreases along reduction is relatively easy and natural. For instance, the paths of context semantics for interaction nets are governed by only 5 rules, compared to the 29 of context semantics for linear logic. However, it seems harder to use it than the linear logic context semantics. We think we still need to define additional tools.

We study another application for this context semantics: a denotational semantics for a large class of interaction net systems. We define a notion of observational equivalence for each library. Then we define

a denotational semantics which is, on a class of libraries named *crossing libraries*, sound and fully abstract with respect to our equivalence. We previously presented those works in [61].

Related works Concerning our first application, we are not aware of other works aiming at proving complexity bounds on generic interaction nets. There are also few tools to analyze the semantics of generic libraries. Lafont defined an observational equivalence, based on paths, for a special library called interaction combinators [46]. Then, he defines a geometry of interaction for interaction combinators: he assigns a weight to each path in the nets such that two nets are equivalent if and only if their paths have the same weights. Thus, the set of weights of paths is a denotational semantics sound and fully abstract for his equivalence.

In [56], Mazza designed an observational equivalence for every library. This equivalence is similar, but not equal to Lafont's on interaction combinators. Then, he defines a denotational semantics for symmetric combinators, a variant of interaction combinators [56, 57]. Symmetric combinators are Turing-complete and can encode a large class of libraries (called *polarized* libraries). However, as we will detail later, defining the semantics of a net as the semantics of its translation in interaction combinators does not give quite a good semantics. It would differentiate nets that behave similarly. Our definition of observational equivalence is strongly inspired from Mazza's.

Finally, in [29], Fernandez and Mackie define an observational equivalence for every library. This equivalence is stronger than Mazza's semantics on symmetric combinators but, in general, they are not comparable.

1.0.5 Organisation of the thesis

In Chapter 2, we present linear logic and context semantics. The results of this chapter either correspond to similar results proven by Dal Lago in his presentation of context semantics [20] or small technical lemmas. In Chapter 3, we present our semantic criteria entailing elementary and polynomial bounds. Section 3.1 presents simple criteria for polynomial and elementary bound (both on *cut*-elimination and β-reduction). Its goal is mainly pedagogic: the criterion defined for polynomial time is only general enough to prove the soundness of *LLL*. In Section 3.2 we define a more general criterion for elementary time. And in Section 3.3 we use it to define a more general criterion for polynomial time. Those two sections contain the main ideas of this thesis. Those ideas are pushed to their limits in Section 3.5, where we define an even more general characterization of polynomial time. In Chapter 4, we use a criterion similar to the criterion of Section 3.2 to define a criterion entailing a primitive recursive bound.

In Chapter 5, we define subsystems of Linear Logic whose complexity bounds rely on the criteria of Chapters 3 and 4. The system $SDNLL$ of Section 5.1.1 characterizes polynomial time and is based on Section 3.3. The system $QPLL$ of Section 5.2 entails a primitive recursive bound and is based on Chapter 4. The system $SwLL$ of Section 5.3 characterizes polynomial time and is based on Section 3.5.

Finally, Chapter 6 presents a context semantics for interaction nets and use it to define a denotational semantics. This chapter is almost independent from the other chapters.

Chapter 2

Context Semantics and Linear Logic

$$\frac{}{\vdash A^\perp, A}\,ax \qquad \frac{\vdash \Gamma, A \qquad \vdash \Delta, A^\perp}{\vdash \Gamma, \Delta}\,cut \qquad \frac{\vdash \Gamma, A \qquad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B}\otimes \qquad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}\parr \qquad \frac{\vdash \Gamma, A[B/X]}{\vdash \Gamma, \exists X.A}\exists$$

$$\frac{\vdash \Gamma, ?A, ?A}{\vdash \Gamma, ?A}\,?C \qquad \frac{\vdash A_1, \cdots, A_n, B}{\vdash ?A_1, \cdots, ?A_n, !B}\,!P \qquad \frac{\vdash \Gamma}{\vdash \Gamma, ?A}\,?W \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, ?A}\,?D \qquad \frac{\vdash \Gamma, ??A}{\vdash \Gamma, \ ?A}\,?N \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, \forall X.A}\forall$$

Figure 2.1: Sequent calculus presentation of linear logic. In the $\forall$ rule, we suppose that X does not appear free in Γ.

2.1 Linear Logic

Linear logic (*LL*) [36] can be considered as a refinement of System F [35, 62] where we focus especially on how the duplication of formulae is managed. In System F, $A \Rightarrow B$ means "with many proofs of A, I can create a proof of B". Linear logic decomposes it into two connectives: $!A$ means "infinitely many proofs of A", $A \multimap B$ means "using exactly one proof of A, I can create a proof of B". We can notice that we can represent $A \Rightarrow B$ with $(!A) \multimap B$. In fact, $A \multimap B$ is a notation of $A^\perp \parr B$. $(\_)^\perp$ can be considered as a negation and $\parr$ as a disjunction. In fact the disjunctions $\vee$ and conjunction $\wedge$ are separated into two disjunctions ($\parr$ and $\oplus$) and two conjunctions ($\otimes$ and $\&$). In this work, we will only use the "multiplicative" ones: $\parr$ and $\otimes$.

Finally $\forall$ and $\exists$ allow us, as in System F, to quantify over the sets of formulae. As examples, let us notice that $\forall X.X \multimap X$ is provable (for any formula X, using exactly one proof of X, we can create a proof of X). On the contrary, $\forall X.X \multimap (X \otimes X)$ is not provable because, in the general case, we need two proofs of X to prove $X \otimes X$.

In this work we use neither the additives ($\oplus$ and $\&$) nor the constants. This fragment is usually named *Multiplicative Exponential Linear Logic with Quantifiers* (abbreviated by $MELL_\forall$). To simplify notations, we will abusively refer to it as *Linear Logic* (abreviated by *LL*). The set $\mathcal{F}_{\mathrm{LL}}$, defined as follows, designs the set of formulae of linear logic.

$$\mathcal{F}_{\mathrm{LL}} = X \mid X^\perp \mid \mathcal{F}_{\mathrm{LL}} \otimes \mathcal{F}_{\mathrm{LL}} \mid \mathcal{F}_{\mathrm{LL}} \parr \mathcal{F}_{\mathrm{LL}} \mid \forall X\, \mathcal{F}_{\mathrm{LL}} \mid \exists X\, \mathcal{F}_{\mathrm{LL}} \mid !\mathcal{F}_{\mathrm{LL}} \mid ?\mathcal{F}_{\mathrm{LL}}$$

You can notice that the "negation" $(\_)^\perp$ is only defined on atomic formulae. We define inductively an involution $(\_)^\perp$ on $\mathcal{F}_{\mathrm{LL}}$, which can be considered as a negation: $(X)^\perp = X^\perp$, $(X^\perp)^\perp = X$, $(A \otimes B)^\perp = A^\perp \parr B^\perp$, $(A \parr B)^\perp = A^\perp \otimes B^\perp$, $(\forall X.A)^\perp = \exists X.A^\perp$, $(\exists X.A)^\perp = \forall X.A^\perp$, $(!A)^\perp = ?(A^\perp)$ and $(?A)^\perp = !(A^\perp)$.

Linear logic is usually presented as a sequent calculus (Figure 2.1). In this thesis, we will consider an alternative syntax, proof-nets, which corresponds to the sequent calculus [37]. They are graph-like structures whose nodes correspond to uses of logical rules. Intuitively, proof-nets are λ-terms where applications and abstractions are respectively replaced by $\otimes$ and $\parr$ and with additional information on duplication. Figure 2.2 illustrates this intuitive correspondence. On the right is a graphical representation of the syntactic tree of $\lambda n.\lambda f.\lambda x.(nf)(nf)x$, on the left is a corresponding proof-net

Definition 1 (proof-net)**.** *A LL* proof-net *is a graph-like structure, defined inductively by the graphs of Figure 2.3 (G and H being LL proof-nets). Every edge e is labelled by $\beta_G(e) \in \mathcal{F}_{\mathrm{LL}}$ (written $\beta(e)$ if the proof-net is obvious from context) satisfying the constraints of Figure 2.3. The set of edges is written* E_G.

A proof-net *is a graph-like structure, whose edges are not labelled, defined inductively by the graphs of Figure 2.3 (G and H being proof-nets). The constraints of Figure 2.3 on labels are not taken into account.*

Let us notice that every *LL* proof-net is a proof-net. For the following definitions, we supposed fixed a proof-net G.

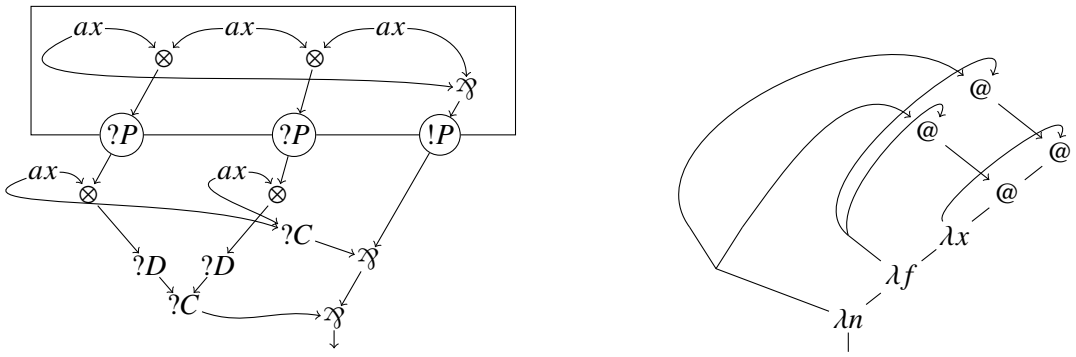


Figure 2.2: We can observe graphically the proofs-as-program correspondence: if we erase the $!P$, $?P$ and $?C$ of the proof net, we obtain the syntactic tree of the corresponding λ-term

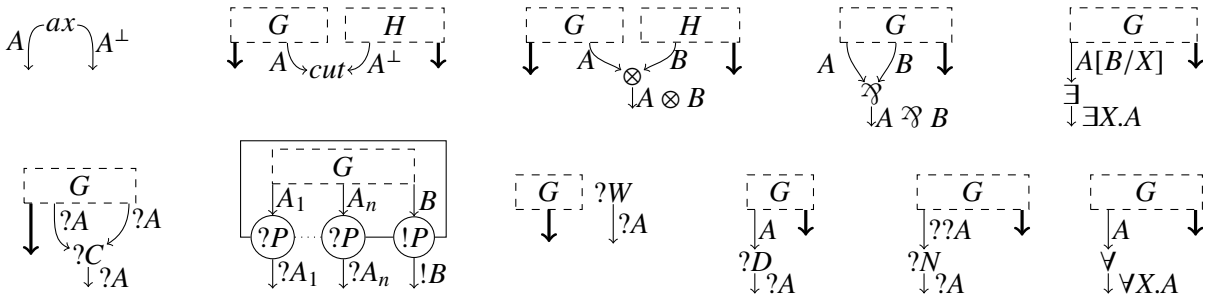


Figure 2.3: Construction of *LL* proof-nets. For the $\forall$ rule, we require X not to be free in the formulae labelling the other conclusions of G

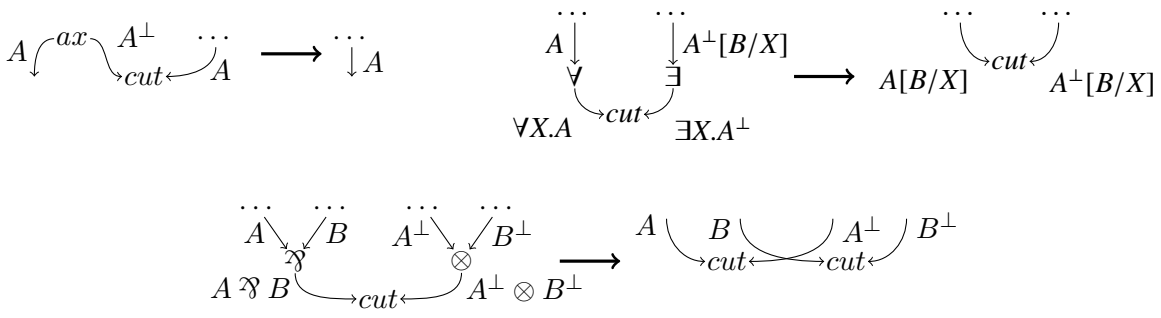


Figure 2.4: Non-exponential *cut*-elimination steps. For the $\forall/\exists$ step, the substitution $[B/X]$ takes place on the whole net.

Nodes Let X be a node label (i.e. $X \in \{ax, cut, \otimes, \parr, \exists, \forall, ?P, !P, ?W, ?D, ?N\}$), then N_G^X refers to the nodes of G whose label is X. The set of nodes of G is written N_G.

Directed edges The edges of proof-nets are directed. For any $(l, m) \in E_G$, we denote its inverted edge (m, l) by $\overline{(l, m)}$. We define $\vec{\mathrm{E}}_G$ as the set of all edges of G as well as their inverted edges ($\vec{\mathrm{E}}_G = E_G \cup \{\overline{e} \mid e \in E_G\}$). We consider that $\overline{e}$ is labelled by the dual of the formula labelling e : for any $e \in E_G$, we define $\beta(\overline{e}) = \beta(e)^\perp$.

Premises and conclusions For any node n, the incoming edges of n are called the *premises* of n. The outgoing edges of n are called the *conclusions* of n. Whenever n has only one premise (resp. conclusion), $prem_n$ (resp. $concl_n$) refers to the premise of n (resp. conclusion of n). The *tail* of the edge (m, n) refers to m, while the *head* of (m, n) refers to n. Some edges have no conclusion. Such edges are called the *pending edges* of G.

Boxes The rectangle of Figure 2.3 with the $?P$ and $!P$ nodes is called a *box*. Formally a box is a subset of the nodes of the proof-net. We say that an edge (m, n) belongs to box B if n is in B.

Let us call B the box in figure 5.1. The node labelled $!P$ is the *principal door* of B, its conclusion is written $\sigma(B)$ and called the *principal edge* of B. The $?P$ nodes are the auxiliary doors of box B. The edge going out of the i-th auxiliary door is written $\sigma_i(B)$ and called the *i-th auxiliary edge* of B. $D_G(B)$ is the set of doors of B and $D_G = \max_{B \in B_G} |D_G(B)|$ (for any set E, $|E|$ refers to the cardinality of E). The doors of box B are considered in box B, they are exactly the nodes which are in B but whose conclusions are not in B.

The number of boxes containing an element (box, node or edge) x is its *depth* written $\partial(x)$. ∂_G is the maximum depth of an edge of G. The set of boxes of G is B_G. $\rho_G(e)$ is the deepest box of G containing e.

Quantifiers We call *eigenvariables* of a proof-net, the variables X quantified by a $\forall$ node. We will always suppose that they are pairwise distinct. Any proof-net which does not respect this convention can be transformed in a proof-net with pairwise distinct eigenvariables by substitutions of variables. This is possible because when we add a $\forall$ node to a proof-net, the eigenvariable can not be free in the other pending edges, so even if the eigenvariables are equal, they can not be related. This allows to refer to "*the* node associated to the eigenvariable X".

***cut*-elimination** *cut*-elimination, is a relation on proof-nets which is related to β-reduction. The rules of *cut*-elimination can be found in figures 2.4 and 2.5.

Lemma 2. *[37] Proof-nets are stable under cut-elimination.*

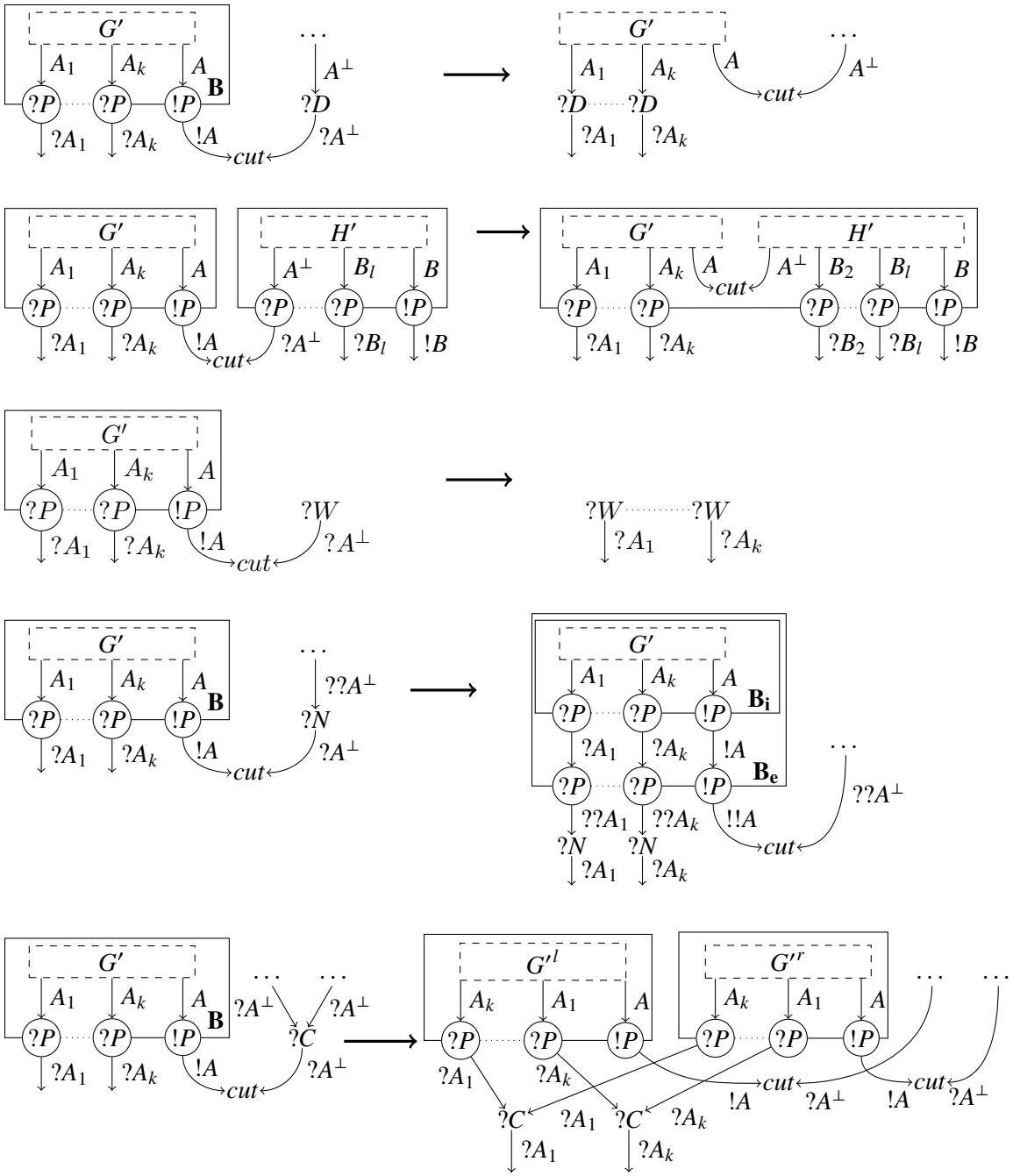


Figure 2.5: Exponential *cut*-elimination steps

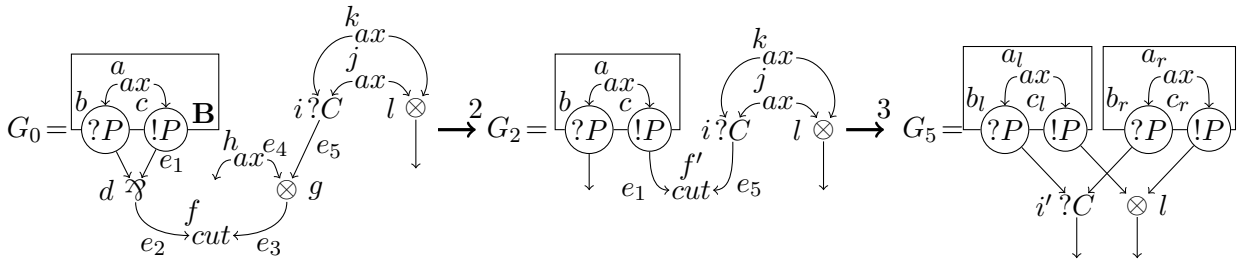


Figure 2.6: Notion of residue. The e_i refers to edges, other lower cases refer to nodes.

2.2 Definition of Context Semantics

A common method to prove strong bounds on a rewriting system is to assign a weight $W_G \in \mathbb{N}$ to each term G such that, if G reduces to H, $W_G > W_H$. In *LL*, the $!P/?C$ step makes the design of such a weight hard: a whole box is duplicated, increasing the number of nodes, edges, cuts,... The idea of context semantics is to define W_G as the number of nodes which can appear during reduction.

Let us suppose that G reduces to H. The reduction concerns only parts of the proof-net, elsewhere there is a canonical one-to-one correspondence between the nodes of G and the nodes of H. We identify corresponding nodes. For instance, in Figure 2.6, the $\otimes$ node l is never affected by *cut*-elimination so we identify the nodes named l in the three nets. There are some nodes of G which are not in H (at least the reduced cuts): in Figure 2.6, the nodes d, f, g and h are deleted in the reduction from G_0 to G_2, the nodes a, b, c, f', i, j and k are deleted in the reduction from G_2 to G_5. Finally, there may be some nodes of H which are not in G: in Figure 2.6, the edge f' is created from G_0 to G_2, a_l, b_l, c_l, a_r, b_r, c_r and i' are created from G_2 to G_5. If one defines, for any proof-net G, Can_G as the set of nodes which can appear during reduction and U_G as $|Can_G|$, then $G \to_{cut} H$ implies $U_G > U_H$ ($Can_H \subseteq Can_G$ and any *cut*-elimination step deletes at least 1 node). So U_G is a bound on the longest *cut*-elimination sequence starting from G. For instance, in Figure 2.6, we have:

$$
\begin{aligned}
Can_{G_0} &= \{a, b, c,\ d, f, g, h,\ f', i, j, k, l, a_l, b_l, c_l, a_r, b_r, c_r, i'\} && U_{G_0} = 19\\
Can_{G_2} &= \{a, b, c,\ f', i, j, k, l, a_l, b_l, c_l, a_r, b_r, c_r, i'\} && U_{G_2} = 15\\
Can_{G_5} &= \{\ l, a_l, b_l, c_l, a_r, b_r, c_r, i'\} && U_{G_5} = 8
\end{aligned}
$$

How can we prove bounds on U_G? In Figure 2.6, the creation of new nodes mostly happens during the $!P/?C$ step. Whenever a box B is duplicated, for every node $n \in B$, two new edges corresponding to n are created. Those edges are *residues* of n. We consider "is a residue of" as a reflexive and transitive relation. Let n be a node of G, $Can(n)$ refers to all the residues of n. For instance $Can(a) = \{a, a_l, a_r\}$, $Can(b) = \{b, b_l, b_r\}$ and $Can(e) = \{e\}$.

We define new weights V_G and W_G by $V_G = \sum_{n \in \mathrm{N}_G} |Can(n)|$ and $W_G = 2 \cdot \sum_{e \in \mathrm{E}_G} |Can(e)|$. Let us observe that V_G is not precisely equal to U_G: for example, in Figure 2.6, there are no $n \in G_0$ such that $Can(n)$ contains f'. Observing every *cut*-elimination rule, we can verify that $G \to H$ implies $V_G > V_H$. For instance, for the proof-nets of Figure 2.6, $V_{G_0} = 17$ (a, b and c have three residues, the other nodes have only one), $V_{G_2} = 14$ and $V_{G_5} = 8$. The weight W_G does not necessarily decrease at each reduction step. However, for every proof-net we have $V_G \leq W_G$ so W_G is also a bound on the length of $\to_{cut}$ sequences starting from G. We prefer to use W_G than V_G because our paths are defined on edges and it will be more natural to prove bounds on the canonical sets of edges.

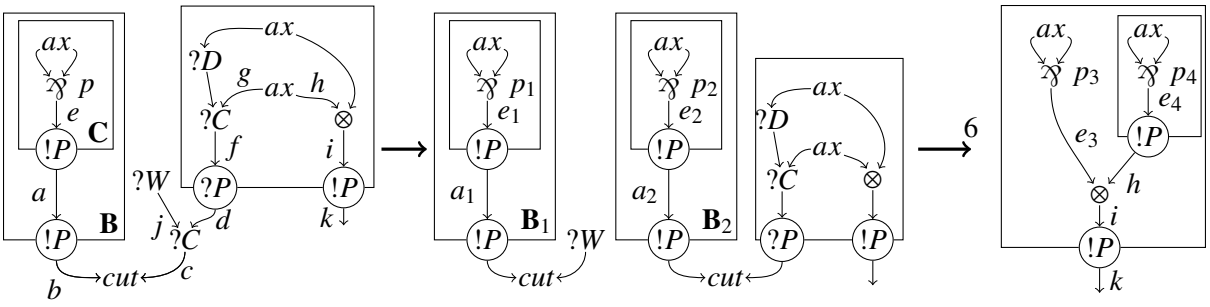


Figure 2.7: Cut-elimination of a proof-net.

To prove bounds on the number of residues of nodes, we will simulate *cut*-elimination by paths in the proof-net. Those paths will be generated by *contexts* travelling across the proof-net according to some rules. The paths of context semantics in a proof-net G are exactly the paths which are preserved by *cut*-elimination (such paths are called *persistent* in the literature [25]). Computing those paths is somehow like reducing the proof-net, and the persistent paths starting at the principal edge of a box correspond to the residues of this box. Proving bounds on the number of residues thanks to those paths rather than proving bounds directly on the reduction will offer two advantages:

- Complex properties on the behaviour of the proof-net, which may be hard to manipulate formally, are transformed into existence (or absence) of paths of a certain shape.
- The rules generating the paths are local, in the same way as typing constraints. We will define subsystems of proof-nets: sets S of proof-nets verifying stricter typing constraints than Figure 2.3. We will prove that, along a path in a proof-net of S, the contexts verify some invariant. From this invariant, we will deduce that it is impossible for the paths to have certain shapes. Finally, because paths only have a certain shape, we will prove a complexity bound for the proof-nets of S.

To represent lists we will use the notation $[a_1;\cdots;a_n]$. To represents concatenation, we will use @: $[a_1;\cdots;a_n]@[b_1;\cdots;b_k]$ is defined as $[a_1;\cdots;a_n;b_1;\cdots;b_k]$ and . represents "push" ($[a_1;\cdots;a_n].b$ is defined as $[a_1;\cdots;a_n;b]$). $|[a_1;\cdots;a_j]|$ refers to j, the length of the list. If X is a set, $|[a_1;\cdots;a_j]|_X$ is the number of indices i such that a_i is in X.

A context is a pair $((e,P),T)$ composed of a *potential edge* (e,P) representing a residue of a directed edge (e is a directed edge of the proof-net) and a *trace* T used to remember some information about the beginning of the path. This information is necessary to ensure that the paths are preserved by *cut*-elimination. Indeed, let us suppose that in Figure 2.6, there is a path of the shape $((e_1,\_),\_)\mapsto((e_2,\_),\_)\mapsto((\overline{e_3},\_),\_)$. If the next edge is $\overline{e_4}$, the path will not be persistent because e_1 and e_4 will be separated during the $⅋/\otimes$ *cut*-elimination step. So the next edge must be $\overline{e_5}$. Information will be put on the trace to remember that we have crossed a $⅋$ from its right premise to its conclusion, this information will be used to force the path to go on $\overline{e_5}$. The following definition introduces the components of potential edges and traces.

To denote signatures, we will usually use the letters t, u and v. The language Sig of *signatures* is defined by induction by the following grammar:

$$Sig = \mathtt{e} \mid \mathtt{l}(Sig) \mid \mathtt{r}(Sig) \mid \mathtt{p}(Sig) \mid \mathtt{n}(Sig, Sig)$$

A signature corresponds to a list of choices of premises of ?C nodes, to designate a particular residue of a box. The signature $\mathtt{r}(t)$ means: "I choose the right premise, and in the next ?C nodes I will use t to make

my choices". The construction $\mathtt{n}(t,u)$ allows to encapsulate two sequels of choices into one. It corresponds to the digging rule $(!!A \vdash B \rightsquigarrow !A \vdash B$, represented by the $?N$ node in proof-nets) which "encapsulates" two ? modalities into one. The $\mathtt{p}(t)$ construction is a degenerated case of the $\mathtt{n}$ construction. Intuitively, $\mathtt{p}(t)$ corresponds to $\mathtt{n}(\varnothing, t)$.

A *potential* is a list of signatures: a signature corresponds to the duplication of one box, but an element is copied whenever any of the boxes containing it is cut with a $?C$ node. The set of potentials is written Pot. A potential edge is a pair (e,P) with e an edge and $P \in Pot$ such that $|P| = \partial(e)$ (a signature for each box containg e). For $e \in E_G$, we define $Pot(e)$ as $\{(e,P) \mid P \in Pot \text{ and } |P| = \partial(e)\}$. We define similarly the notion of potential boxes, potential nodes and the notations $Pot(B)$ and $Pot(n)$.

Potentials will be used to represent residues. For instance, the residues of e in Figure 2.7, (e, e_1, e_2, e_3 and e_4) will be respectively represented by the potential edges $(e, [\mathtt{e};\mathtt{e}])$, $(e, [\mathtt{l}(\mathtt{e});\mathtt{e}])$, $(e, [\mathtt{r}(\mathtt{e});\mathtt{r}(\mathtt{e})])$ and $(e, [\mathtt{r}(\mathtt{e});\mathtt{l}(\mathtt{e})])$.

A *trace element* is one of the following characters: $⅋_l, ⅋_r, \otimes_l, \otimes_r, \forall, \exists, !_t, ?_t$ with t a signature. A trace element means "I have crossed a node with this label, from that premise to its conclusion". A *trace* is a non-empty list of trace elements. The set of traces is Tra. A trace is a memory of the path followed, up to cut-eliminations. We define duals of trace elements: $⅋_l^\perp = \otimes_l$, $!_t^\perp = ?_t$,... and extend the notion to traces by $([a_1;\cdots;a_k])^\perp = [a_1^\perp;\cdots;a_k^\perp]$.

A *context* is a tuple $((e,P),T)$ with (e,P) a potential edge and $T \in Tra$. It can be seen as a state of a token that will travel around the net. It is located on edge e (more precisely its residue corresponding to P) and carries information T about its past travel. The set of contexts of G is written $Cont_G$. We extend the mapping $(\_)^\perp$ on contexts by $((e,P),T)^\perp = ((\overline{e},P),T^\perp)$.

The nodes define two relations $\rightsquigarrow$ and $\hookrightarrow$ on contexts. The rules are presented in Figure 2.8. Observe that these rules are deterministic. For any rule $C \rightsquigarrow D$ presented in Figure 2.8, we also define the dual rule $D^\perp \rightsquigarrow C^\perp$. We define $\mapsto$ as the union of $\rightsquigarrow$ and $\hookrightarrow$. In other words, $\mapsto$ is the smallest relation on contexts including every instance of $\rightsquigarrow$ rules in Figure 2.8 together with every instance of their duals and every instance of the $\hookrightarrow$ rule.

Let us notice that the rules are sound: if $((e,P),T) \mapsto ((f,Q),U)$, then $\partial(e) = |P|$ if and only if $\partial(f) = |Q|$. The only rules which modify the length of potentials are the rules entering and leaving a box. Let us also notice that if e is the conclusion of a $?N$ or $?P$, then the context C such that $((e,P),T.!_t) \mapsto C$ depends on the size of T: there is a rule in the case $T = []$ and another in the case $T \neq []$.

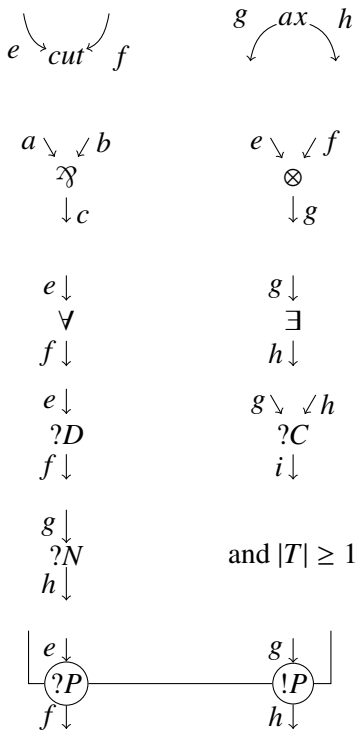
For every sequence $((e_1,P_1),T_1) \rightsquigarrow ((e_2,P_2),T_2) \rightsquigarrow \cdots \rightsquigarrow ((e_n,P_n),T_n)$, the sequence of directed edges $e_1,\cdots,e_n$ is a path (i.e the head of e_i is the same node as the tail of e_{i+1}). The $\hookrightarrow$ relation breaks this property as it is non-local, in the sense that it deals with two non-adjacent edges. It is the main difference between Dal Lago's context semantics and Girard's geometry of interaction. The study of $\mapsto$*-paths*, sequences of the shape $C_1 \mapsto C_2 \mapsto \ldots$, will give us information on complexity. The trace keeps track of the history of previously crossed nodes to enforce path persistence: the $\mapsto$-paths are preserved by cut-elimination.

As an example, the path in the first proof-net of Figure 2.7:

$$((e,[\mathtt{r}(\mathtt{e});\mathtt{r}(\mathtt{e})]),[⅋_r]) \mapsto ((a,[\mathtt{r}(\mathtt{e})]),[⅋_r;!_{\mathtt{r}(\mathtt{e})}]) \mapsto ((b,[]),[⅋_r;!_{\mathtt{r}(\mathtt{e})};!_{\mathtt{r}(\mathtt{e})}]) \mapsto$$
$$((\overline{c},[]),[⅋_r;!_{\mathtt{r}(\mathtt{e})};!_{\mathtt{r}(\mathtt{e})}]) \mapsto ((\overline{d},[]),[⅋_r;!_{\mathtt{r}(\mathtt{e})};!_{\mathtt{e}}]) \mapsto ((\overline{f},[\mathtt{e}]),[⅋_r;!_{\mathtt{r}(\mathtt{e})}]) \mapsto ((\overline{g},[\mathtt{e}]),[⅋_r;!_{\mathtt{e}}]) \mapsto$$
$$((h,[\mathtt{e}]),[⅋_r;!_{\mathtt{e}}]) \mapsto ((i,[\mathtt{e}]),[⅋_r;!_{\mathtt{e}};\otimes_r]) \mapsto ((k,[]),[⅋_r;!_{\mathtt{e}};\otimes_r;!_{\mathtt{e}}])$$

becomes $((e_4,[\mathtt{e};\mathtt{e}]),[⅋_r]) \mapsto ((h,[\mathtt{e}]),[⅋_r;!_{\mathtt{e}}]) \mapsto ((i,[\mathtt{e}]),[⅋_r;!_{\mathtt{e}};\otimes_r]) \mapsto ((k,[]),[⅋_r;!_{\mathtt{e}};\otimes_r;!_{\mathtt{e}}])$ in the third proof-net of Figure 2.7.

Let us notice, with Lemma 3, that $\rightsquigarrow$ is injective. It is not the case for the $\mapsto$ relation. Indeed, if B is a box with two auxiliary doors then, for every potential P and signature t, we have $((\overline{\sigma_1(B)},P),[!_t]) \mapsto$



$$
\begin{array}{llcll}
((e,P), & T) & \rightsquigarrow & ((\overline{f},P), & T) \\
((\overline{g},P), & T) & \rightsquigarrow & ((h,P), & T)
\end{array}
$$

$$
\begin{array}{llcll}
((a,P), & T) & \rightsquigarrow & ((c,P), & T.⅋_l) \\
((b,P), & T) & \rightsquigarrow & ((c,P), & T.⅋_r) \\
((e,P), & T) & \rightsquigarrow & ((g,P), & T.\otimes_l) \\
((f,P), & T) & \rightsquigarrow & ((g,P), & T.\otimes_r)
\end{array}
$$

$$
\begin{array}{llcll}
((e,P), & T) & \rightsquigarrow & ((f,P), & T.\forall) \\
((g,P), & T) & \rightsquigarrow & ((h,P), & T.\exists)
\end{array}
$$

$$
\begin{array}{llcll}
((e,P), & T) & \rightsquigarrow & ((f,P), & T.?_{\mathtt{e}},+) \\
((g,P), & T.?_t) & \rightsquigarrow & ((i,P), & T.?_{\mathtt{l}(t)},+) \\
((h,P), & T.?_t) & \rightsquigarrow & ((i,P), & T.?_{\mathtt{r}(t)},+)
\end{array}
$$

$$
\begin{array}{llcll}
((g,P), & T.?_{t_1}.?_{t_2}) & \rightsquigarrow & ((h,P), & T.?_{\mathtt{n}(t_1,t_2)}) \\
((g,P), & [?_t]) & \rightsquigarrow & ((h,P), & [?_{\mathtt{p}(t)}])
\end{array}
$$

$$
\begin{array}{llcll}
((e,P.t), & T) & \rightsquigarrow & ((f,P), & T.?_t) \\
((g,P.t), & T) & \rightsquigarrow & ((h,P), & T.!_t) \\
((\overline{f},P), & [!_t]) & \hookrightarrow & ((h,P), & [!_t])
\end{array}
$$

Figure 2.8: Rules of the context semantics

$((\sigma(B),P),[!_t])$ and $((\overline{\sigma_2(B)},P),[!_t]) \mapsto ((\sigma(B),P),[!_t])$.

Lemma 3. *If $C_1 \rightsquigarrow D$ and $C_2 \rightsquigarrow D$ then $C_1 = C_2$*

2.2.1 Capturing the notion of residue

Let $e \in \mathrm{E}_G$, there are potential nodes in $Pot(e)$ which do not correspond to residues of e. For instance, in Figure 2.7 a has three residues: a, a_1 and a_2. The residue a_1 is obtained by choosing the left box during the duplication of box B, so it will be represented by $(a,[\mathtt{l}(\mathtt{e})])$. Similarly, a_2 will be represented by $(a,[\mathtt{r}(\mathtt{e})])$ and a by $(a,[\mathtt{e}])$. However, $(a,[\mathtt{r}(\mathtt{l}(\mathtt{e}))])$ does not represent any residue. The potential node $(a,[\mathtt{r}(\mathtt{l}(\mathtt{e}))])$ means that whenever the box B_2 is cut with a $?C$ node, we chose the left box. But this situation never happens. It can be observed by the following path:

$$((\sigma(B),[]),[!_{\mathtt{r}(\mathtt{l}(\mathtt{e}))}]) \mapsto ((\overline{c},[]),[!_{\mathtt{r}(\mathtt{l}(\mathtt{e}))}]) \mapsto ((\overline{d},[]),[!_{\mathtt{l}(\mathtt{e})}]) \not\mapsto$$

The $\mathtt{l}(\_)$ has not been used because we did not encounter a second $?C$ node. On the contrary, the signatures corresponding to residues are entirely used:

$$((\sigma(B),[]),[!_{\mathtt{e}}]) \mapsto^0 ((b,[]),[!_{\mathtt{e}}])$$
$$((\sigma(B),[]),[!_{\mathtt{l}(\mathtt{e})}]) \mapsto^2 ((\overline{j},[]),[!_{\mathtt{e}}])$$
$$((\sigma(B),[]),[!_{\mathtt{r}(\mathtt{e})}]) \mapsto^2 ((\overline{d},[]),[!_{\mathtt{e}}])$$

So, as a first try, we could say that for any object x at depth 1 (with B the box containing x), and $t \in Sig$, $(x,[t])$ represents a residue of x if and only if $((\sigma(B),[]),[!_t]) \mapsto^* ((\_,\_),[!_{\mathtt{e}}]@T)$. Throughout the thesis,

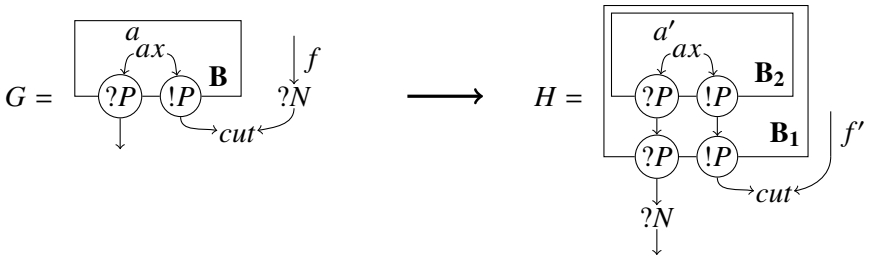


Figure 2.9: The potential edge $(a, [\mathtt{n}(t_2, t_1)])$ corresponds to $(a', [t_1; t_2])$.

we use _ to denote an object whose name and value are not important to us, for example $C \mapsto \_$ means $\exists D \in Cont_G, C \mapsto D$.

Now, let us consider an object x at depth ≥ 2 and $B = \rho_G(x)$ (for example the $\parr$ node p in Figure 2.7). By analogy with the case $\partial(x) = 1$, we could say that "$(x, P.t)$ represents a residue of x if $((\sigma(B), P), [!_t]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@T)$. However, in Figure 2.7, for any $t \in Sig$, $(p, [\mathtt{r}(t), \mathtt{e}])$ would satisfy this condition. Indeed $((\sigma(C), [\mathtt{r}(t)]), [!_{\mathtt{e}}]) \mapsto^3 ((\overline{d}, []), [!_{\mathtt{e}}; !_t]) \mapsto ((\overline{f}, [t]), [!_{\mathtt{e}}])$. But n has only 5 residues (p, p_1, p_2, p_3 and p_4), not an infinity, so the condition we tried is too weak.

If $n \in B_{\partial(n)} \subset \cdots \subset B_1$, then in the potential node $(n, [t_1; \cdots; t_{\partial(e)}])$, the signature t_i corresponds to the choices we make whenever B_i is cut with a $?C$ node. So, $(n, [t_1; \cdots; t_{\partial(e)}])$ corresponds to a residue of n if and only if for every $1 \leq i \leq \partial(n)$, $((\sigma(B_i), [t_1; \cdots; t_{i-1}]), [!_{t_i}]) \mapsto^* (\_, [!_{\mathtt{e}}])$.

2.2.2 Dealing with the digging

Now we will consider what happens when $?N$ nodes are allowed. Let us consider the node a in Figure 2.9. The residues of a are exactly the residues of a' and a itself. So "$(a', [t_1; t_2])$ corresponds to a residue of a" is successively equivalent to:

$$\begin{cases} ((\sigma(B_2), [t_1]), [!_{t_2}]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_) \\ ((\sigma(B_1), []), [!_{t_1}]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_) \end{cases}$$

$$\begin{cases} ((\overline{f'}, []), [!_{t_2}; !_{t_1}]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_) \\ ((\overline{f'}, []), [!_{t_1}]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_) \end{cases}$$

$$\begin{cases} ((\overline{f}, []), [!_{t_2}; !_{t_1}]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_) \\ ((\overline{f}, []), [!_{t_1}]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_) \end{cases}$$

$$\begin{cases} ((\sigma(B), []), [!_{\mathtt{n}(t_2,t_1)}]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_) \\ ((\sigma(B), []), [!_{\mathtt{p}(t_1)}]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_) \end{cases}$$

In this case, we write that $(a, [\mathtt{n}(t_2, t_1)])$ corresponds to a residue of a. In fact, the $\mathtt{p}(\ )$ construction never appears in the signatures of potential nodes corresponding to residues: any node in B will be inside both B_1 and B_2 during reduction, so we need a signature describing the sequences of choices for both B_1 and B_2. However, we need to check that both sequences of choices are valid: both $\mathtt{n}(t_2, t_1)$ and $\mathtt{p}(t_1)$ must be entirely used during the $\mapsto$-paths. Let us notice that a box may encounter several $?N$ nodes during *cut*-elimination. To check every box, we will define a relation $\sqsubseteq$ on signatures such that, in particular $\mathtt{n}(t_2, t_1) \sqsubseteq \mathtt{n}(t_2, t_1)$ and $\mathtt{n}(t_2, t_1) \sqsubseteq \mathtt{p}(t_1)$. Then, $(a, [t])$ will correspond to a residue of a if and only if for every u such that $t \sqsubseteq u$, $((\sigma(n), []), [!_u]) \mapsto^* (\_, [!_{\mathtt{e}}]@\_)$.

Definition 4 (standard signature)**.** *A signature is said* standard *if it does not contain the constructor* $\mathtt{p}$*. A signature t is* quasi-standard *iff for every subtree $\mathtt{n}(t_1, t_2)$ of t, the signature t_2 is standard.*

The binary relation $\sqsubseteq$ on Sig is defined as follows:

$$\begin{aligned}
\mathtt{e} &\sqsubseteq \mathtt{e} \\
\mathtt{l}(t) &\sqsubseteq \mathtt{l}(t') && \Leftrightarrow t \sqsubseteq t' \\
\mathtt{r}(t) &\sqsubseteq \mathtt{r}(t') && \Leftrightarrow t \sqsubseteq t' \\
\mathtt{p}(t) &\sqsubseteq \mathtt{p}(t') && \Leftrightarrow t \sqsubseteq t' \\
\mathtt{n}(t_1, t_2) &\sqsubseteq \mathtt{p}(t') && \Leftrightarrow t_2 \sqsubseteq t' \\
\mathtt{n}(t_1, t_2) &\sqsubseteq \mathtt{n}(t'_1, t'_2) && \Leftrightarrow t_1 \sqsubseteq t'_1 \text{ and } t_2 = t'_2
\end{aligned}$$

If $t \sqsubseteq t'$, then t' is a *simplification* of t. We also write $t \sqsubset t'$ for "$t \sqsubseteq t'$ and $t \neq t'$"". We can observe that $\sqsubseteq$ is an order and $\sqsubset$ a strict order.

Our notation is reversed compared to Dal Lago's notation in [20]. Intuitively, $\sqsubseteq$ corresponds to an inclusion of future duplicates, but with the notation of [20], $\sqsubseteq$ corresponds to $\supseteq$. We find this correspondence counter-intuitive, so we reversed the symbol.

Lemma 5. *Let $t \in Sig$, then $\sqsubseteq$ is a total order on $\{u \in Sig \mid t \sqsubseteq u\}$.*

Proof. Let $t \in Sig$ and $u, v \in Sig$ such that $t \sqsubseteq u$ and $t \sqsubseteq v$. We will prove the lemma by induction on t that either $u \sqsubseteq v$ or $v \sqsubseteq u$.

If $t = \mathtt{e}$, then $u = v = \mathtt{e}$ so $u \sqsubseteq v$ and $v \sqsubseteq u$. If $t = \mathtt{l}(t')$ then $u = \mathtt{l}(u')$ with $t' \sqsubseteq u'$ and $v = \mathtt{l}(v')$ with $t' \sqsubseteq v'$. By induction hypothesis, either $u' \sqsubseteq v'$ (and in this case $u \sqsubseteq v$) or $v' \sqsubseteq u'$ (and in this case $v \sqsubseteq u$). The cases $t = \mathtt{r}(t')$ and $t = \mathtt{p}(t')$ are similar.

If $t = \mathtt{n}(t_1, t_2)$ then either $u = \mathtt{n}(u_1, t_2)$ with $t_1 \sqsubseteq u_1$ or $u = \mathtt{p}(u_2)$ with $t_2 \sqsubseteq u_2$. And either $v = \mathtt{n}(v_1, t_2)$ with $t_1 \sqsubseteq v_1$ or $v = \mathtt{p}(v_2)$ with $t_2 \sqsubseteq v_2$.

- If $u = \mathtt{n}(u_1, t_2)$ and $v = \mathtt{n}(v_1, t_2)$ then, by induction hypothesis, either $u_1 \sqsubseteq v_1$ (and in this case $u \sqsubseteq v$) or $v_1 \sqsubseteq u_1$ (and in this case $v \sqsubseteq u$).
- If $u = \mathtt{p}(u_2)$ and $v = \mathtt{p}(v_2)$ then, by induction hypothesis, either $u_2 \sqsubseteq v_2$ (and in this case $u \sqsubseteq v$) or $v_2 \sqsubseteq u_2$ (and in this case $v \sqsubseteq u$).
- If $u = \mathtt{n}(u_1, t_2)$ and $v = \mathtt{p}(v_2)$, then $u \sqsubseteq v$.
- If $u = \mathtt{p}(v_2)$ and $v = \mathtt{n}(v_1, t_2)$, then $v \sqsubseteq u$.

□

Definition 6. *A context $((e, [P_1; \cdots; P_j]), [T_0; T_1; \cdots; T_k])$ is said* quasi-standard *if:*

- *For $1 \leq i \leq j$, P_i is standard.*
- *For $1 \leq i \leq k$, if $T_i = !_u$ or $T_i = ?_u$, then u is standard.*
- *If $T_0 = !_t$ or $T_0 = ?_t$, then t is quasi-standard.*

If we additionally suppose that either T_0 is of the shape $!_t$ with t a standard signature or T_0 is not an exponential trace element (i.e. T_0 is not of the shape $!\_$ or $?\_$), then $((e, [P_1; \cdots ; P_j]), [T_0; T_1; \cdots ; T_k])$ is said standard*.*

The following Lemma proves that those contexts are stable along $\mapsto$. For most of the contexts C' we will study, there exists a context $C = ((\sigma(B), [p_1; \cdots ; p_k]), [!_t])$ with $p_1, \cdots , p_k$ standard signatures and t a quasi-standard signature such that $C \mapsto C'$. By Lemma 7, C' is a quasi-standard context. Thus, unless we explicitly state otherwise, all the contexts we define are quasi-standard.

Lemma 7. *If $C \mapsto D$ then C is quasi-standard if and only if D is quasi-standard.*

Proof. The only steps where a $\mathtt{n}(\_, \_)$ or $\mathtt{p}(\_)$ appears or disappears during a $\mapsto$-step is crossing a $?N$ node.

- If $C = ((\overline{f}, P), [!_{\mathtt{p}(u)}]) \mapsto ((\overline{e}, P), [!_u]) = D$, let us notice that $\mathtt{p}(u)$ is quasi-standard iff u is quasi-standard. So C is quasi-standard iff D is quasi-standard.
- If $C = ((\overline{f}, P), [!_{\mathtt{n}(t,u)}]) \mapsto ((\overline{e}, P), [!_t; !_u]) = D$, let us notice that $\mathtt{n}(t, u)$ is quasi-standard iff t is quasi-standard and u is standard. So C is quasi-standard iff D is quasi-standard.
- If $C = ((\overline{f}, P), T.!_{\mathtt{n}(t,u)}) \mapsto ((\overline{e}, P), T.!_t.!_u)$ with $|T| \geq 1$, then $\mathtt{n}(t, u)$ is standard iff t is standard and u is standard. So C is quasi-standard iff D is quasi-standard.
- If $C = ((e, P), [?_u]) \mapsto ((f, P), [?_{\mathtt{p}(u)}]) = D$, let us notice that $\mathtt{p}(u)$ is quasi-standard iff u is quasi-standard. So D is quasi-standard iff C is quasi-standard.
- If $C = ((e, P), [?_t; ?_u]) \mapsto ((f, P), [?_{\mathtt{n}(t,u)}]) = D$, let us notice that $\mathtt{n}(t, u)$ is quasi-standard iff t is quasi-standard and u is standard. So D is quasi-standard iff C is quasi-standard.
- If $C = ((e, P), T.?_t.?_u) \mapsto ((f, P), T.?_{\mathtt{n}(t,u)})$ with $|T| \geq 1$, then $\mathtt{n}(t, u)$ is standard iff t is standard and u is standard. So D is quasi-standard iff C is quasi-standard.

□

Lemma 8. *If $C \mapsto D$ and C is standard then D is standard.*

Proof. The only steps where a $\mathtt{n}(\_, \_)$ or $\mathtt{p}(\_)$ appears or disappears during a $\mapsto$-step is crossing a $?N$ node.

- If $C = ((\overline{f}, P), T.!_{\mathtt{n}(t,u)}) \mapsto ((\overline{e}, P), T.!_t.!_u)$, then $\mathtt{n}(u, v)$ is standard iff t is standard and u is standard. So C is standard iff D is standard.
- If $C = ((e, P), T.?_t.?_u) \mapsto ((f, P), T.?_{\mathtt{n}(t,u)})$ with $|T| \geq 1$, then $\mathtt{n}(t, u)$ is standard iff t is standard and u is standard. So C is standard iff D is standard.

□

If $C = ((e, P), T) \mapsto ((f, Q), U)$ and this step only depends on the rightmost trace element of T, then for every trace V, $((e, P), V@T) \mapsto ((f, Q), V@U)$. The only $\mapsto$-steps which are not in this case are steps of the shape $C = ((e, P), [?_t]) \rightsquigarrow ((f, P), [?_{\mathtt{p}(t)}])$ and $C = ((\overline{f}, P), [!_{\mathtt{p}(t)}]) \rightsquigarrow ((\overline{e}, P), [!_t])$ (in those cases, C is not standard) and $((\sigma_i(B), P), [!_t]) \hookrightarrow ((\sigma(B), P), [!_t])$. Thus, as stated in Lemma 8, if we only consider the relation on $\rightsquigarrow$ on standard contexts, then path are stable by the concatenation of traces on the left.

Lemma 9. *If $((e, P), T)$ is a standard context, $((e, P), T) \rightsquigarrow^* ((f, Q), U)$ and $V \in Tra$, $((e, P), V@T) \rightsquigarrow^* ((f, Q), V@U)$.*

Proof. It is enough to prove the lemma for one $\rightsquigarrow$ step. To prove this lemma, we just have to examine the $\rightsquigarrow$ rules which depend on other trace elements than the rightmost one. If $((e, P), [!_{\mathrm{p}(t)}]) \rightsquigarrow ((f, P), [!_t])$, then $((e, P), T)$ is not a standard context which contradicts our hypothesis.

Let us notice that the Lemma would not stand if we replaced $((e, P), T) \overline{\rightsquigarrow}^* ((f, Q), U)$ by $((e, P), T) \mapsto^* ((f, Q), U)$. Indeed we can have $((\overline{\sigma_i(B)}, P), [!_t]) \mapsto ((\sigma(B), P), [!_t])$ and $((\overline{\sigma_i(B)}, P), V.!_t) \mapsto ((f, P.t), V)$. □

2.3 Dal Lago's weight theorem

We capture the notion of residue by *canonical potentials*. The definition of canonical potentials relies on *copies*. A copy represents the choices for one box, a canonical potential for an element x is a list of copies: one copy for each box containing x.

Definition 10. *A copy context is a context of the shape* $((e, P), [!_t]@T)$ *such that for every* $u \sqsupseteq t$*, there exists a path of the shape* $((e, P), [!_u]@T) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_)$.

Let $(B, P) \in Pot(B_G)$*, the set* $Cop(B, P)$ *of copies of* (B, P) *is the set of quasi-standard signatures* t *such that* $((\sigma(B), P), [!_t])$ *is a copy context.*

Definition 11. *Let* x *be an element (box, edge or node) of* G *such that* $x \in B_{\partial(x)} \subset ... \subset B_1$*. The set* $Can(x)$ *of canonical potentials for* x *is the set of tuples* $(x, [p_1; ...; p_{\partial(e)}])$ *with* $p_1, \cdots, p_{\partial(e)}$ *signatures such that:*

$$\forall 1 \leq i \leq \partial(x), p_i \in Cop(B_i, [p_1; \cdots; p_{i-1}])$$

For example, in Figure 2.6, $Cop(B, []) = \{\mathtt{e}, \mathtt{l(e)}, \mathtt{r(e)}\}$ so $Can(a) = \{a\} \times \{[\mathtt{e}], [\mathtt{l(e)}], [\mathtt{r(e)}]\}$. In Figure 2.7, $Cop(B, []) = \{\mathtt{e}, \mathtt{l(e)}, \mathtt{r(e)}\}$, $Cop(C, [\mathtt{e}]) = Cop(C, [\mathtt{l(e)}]) = \{\mathtt{e}\}$ and $Cop(C, [\mathtt{r(e)}]) = \{\mathtt{e}, \mathtt{l(e)}, \mathtt{r(e)}\}$, so:

$$Can(p) = \{p\} \times \{[\mathtt{e}; \mathtt{e}], [\mathtt{l(e)}; \mathtt{e}], [\mathtt{r(e)}; \mathtt{e}], [\mathtt{r(e)}; \mathtt{l(e)}]; [\mathtt{r(e)}; \mathtt{r(e)}]\}$$

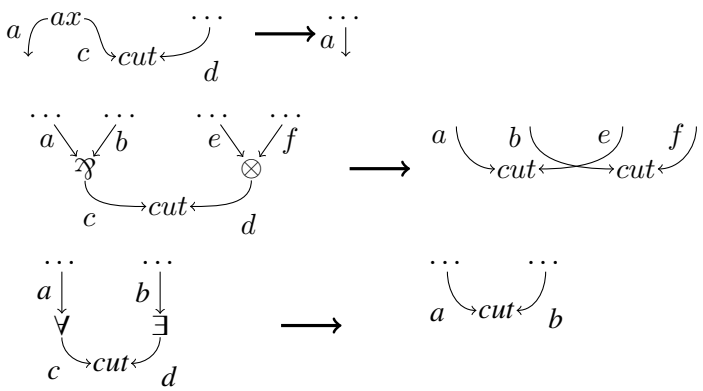
Let us notice that the canonical potentials of e only depend on the boxes containing e. More formally, if e and f are contained in the same boxes then $Can(e) = \{(e, P) \mid (f, P) \in Can(f)\}$.

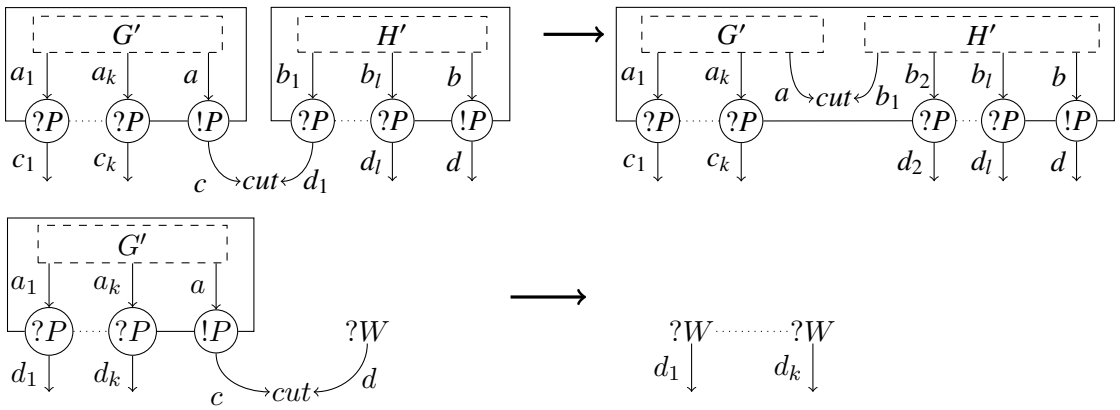
A *canonical edge* (resp. node, box) is a tuple $(x, P) \in Can(x)$ with x an edge (resp. node, box). The set of canonical edges of G is represented by $Can(\mathrm{E}_G)$. More generally, if f is a mapping from A to B and A' is a subset of A then $f(A')$ refers to the image of A' (the set $\{f(x) \mid x \in A'\}$).

The main result of this section is the weight theorem (Theorem 18) stating that whenever $G \rightarrow_{cut} H$, $\sum_{n \in \mathrm{N}_G} |Can(n)| > \sum_{n \in \mathrm{N}_H} |Can(n)|$. It is a slight variation of the Lemma 6 of Dal Lago in [20]. This result allows to prove strong complexity bounds for several systems.

Definition 12. *Let us suppose that* $G \rightarrow_{cut} H$ *then we define a partial mapping* $\pi_{G \to H}(\_)$ *(or simply* π *when the reduction considered can be deduced from the context) from* $Cont_H$ *to* $Cont_G$*. If* $G_1 \rightarrow_{cut} G_2 \cdots \rightarrow_{cut} G_k$*, then we define* $\pi_{G_1 \to G_k}(C)$ *as* $\pi_{G_1 \to G_2}(\pi_{G_2 \to G_3}(\cdots \pi_{G_{k-1} \to G_k}(C) \cdots))$*. Below, we only define* $\pi((e, P), T)$ *whenever* e *is a positive arrow. For any* $e \in E_H$ *such that* $\pi((e, P), T) = ((f, Q), U)$*, we define* $\pi((\overline{e}, P), T^{\perp}) = ((\overline{f}, Q), U^{\perp})$.

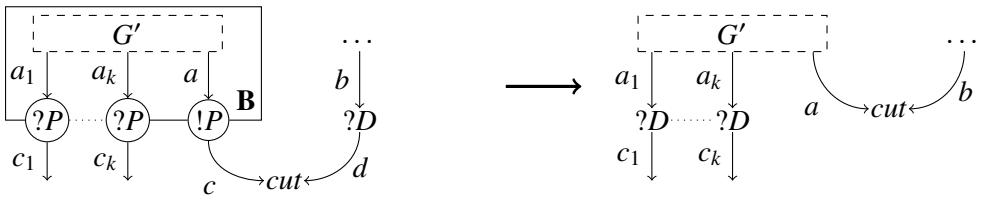
Let us consider the following reduction steps

- 



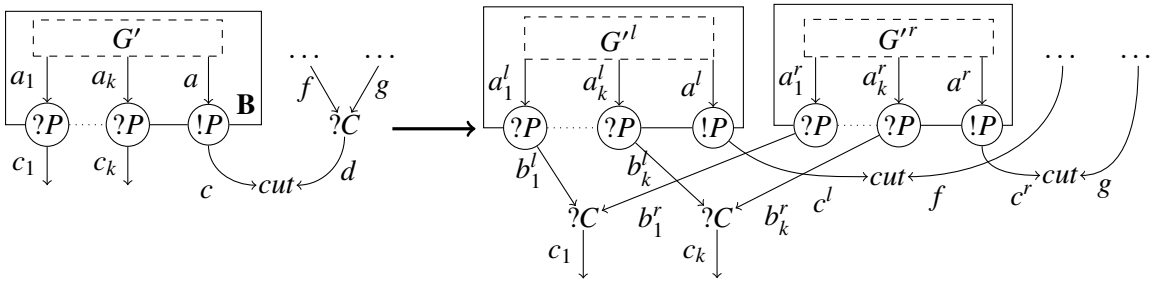
In those reduction steps, for any $(e, P) \in Pot(E_H)$ *and* $T \in Tra$,

$$\pi((e, P), T) = ((e, P), T)$$



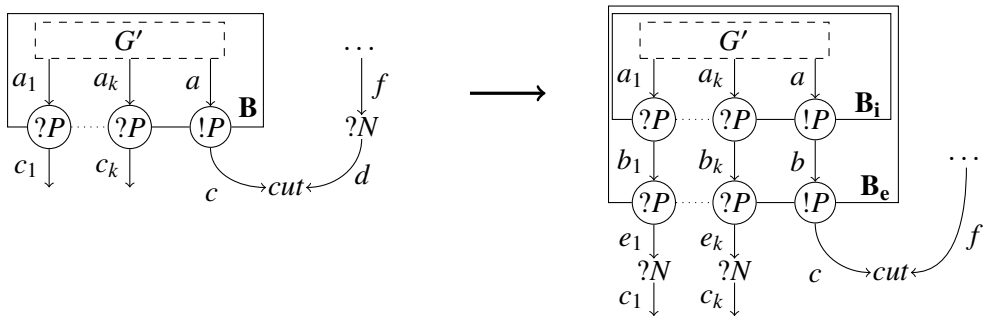
Let $P \in Pot$ *with* $|P| = \partial(B)$, *and* $T \in Tra$,

- *For any* $(e, P@Q) \in Pot(E_{G'})$, $\pi((e, P@Q), T) = ((e, P.\mathsf{e}@Q), T)$.
- *Else,* $\pi((e, P), T) = ((e, P), T)$

- 

For every $e \in E_{G'}$, *we name* e^l *(resp.* e^r*) its residue in* G'^l *(resp.* G'^r*). Let us consider* $P \in Pot$ *with* $|P| = \partial(B)$, *and* $T \in Tra$,

- *For any* $(e, P.t@Q) \in Pot(E_{G'})$, $\pi((e^l, P.t@Q), T) = ((e, P.\mathtt{l}(t)@Q), T)$ *and* $\pi((e^r, P.t@Q), T) = ((e, P.\mathtt{r}(t)@Q), T)$.
- *If* $T = U.!_t$, *then we set* $\pi((c^l, P), T) = ((c, P), U.!_{\mathtt{l}(t)})$ *and* $\pi((c^r, P), T) = ((c, P), U.!_{\mathtt{r}(t)})$. *Else,* $\pi((c^l, P), T)$ *and* $\pi((c^r, P), T)$ *are undefined.*
- *For any* $1 \leq i \leq k$, *if* $T = U.?_t$ *then* $\pi((b_i^l, P), T) = ((c_i, P), U.?_{\mathtt{l}(t)})$ *and* $\pi((b_i^r, P), T) = ((c_i, P), U.?_{\mathtt{r}(t)})$. *Else* $\pi((b_i^l, P), T)$ *and* $\pi((b_i^r, P), T)$ *are undefined.*
- *Else,* $\pi((e, P), T) = ((e, P), T)$.

- 

Let $P \in Pot$ with $|P| = \partial(B)$ and $T \in Tra$,

- *For any $(e, Q) \in Pot(E_{G'})$, $\pi((e, P.t.u@Q), T) = ((e, P.\mathrm{n}(t,u)@Q), T)$.*
- *If $T = [!_t]$, $\pi((c, P), T) = ((c, P), [!_{\mathrm{p}(t)}])$. If $T = U.!_t.!_u$, $\pi((c, P), T) = ((c, P), U.!_{\mathrm{n}(t,u)})$. Else, $\pi((c, P), T)$ is undefined.*
- *If $T = U.!_t$, $\pi((b, P.u), T) = ((c, P), U.!_{\mathrm{n}(t,u)})$. else $\pi((b, P.u), T)$ is undefined.*
- *For any $1 \leq i \leq k$, if $T = [?_t]$ then $\pi((e_i, P), T) = ((c_i, P), [?_{\mathrm{p}(t)}])$. If $T = U.?_t.?_u$ then $\pi((e_i, P), T) = ((c_i, P), U.?_{\mathrm{n}(t,u)})$. Else $\pi((e_i, P), T)$ is undefined.*
- *For any $1 \leq i \leq k$, if $T = U.?_u$ then $\pi((b_i, P.t), T) = ((c_i, P), U.?_{\mathrm{n}(t,u)})$. Else $\pi((b_i, P.t), T)$ is undefined.*
- *Else, $\pi((e, P), T) = ((e, P), T)$*

Lemma 13. *Let us suppose that $G \to_{cut} H$ and C, D are contexts of H such that $\pi(C)$ and $\pi(D)$ are defined. Then,*

$$C \mapsto^* D \Rightarrow \pi(C) \mapsto^* \pi(D)$$
$$C \mapsto^+ D \Leftarrow \pi(C) \mapsto^+ \pi(D)$$

Proof. One can observe every possible reduction. Let us notice that:

- It is enough to check both implications for minimal paths: i.e. $C \mapsto^* D$ and none of the intermediary contexts in the path are in the domain of π, or $\pi(C) \mapsto^+ \pi(D)$ and none of the other contexts in the path are in the codomain of π.
- The context $((e, P), T)$ is invariant by π whenever e is not concerned by the *cut*-elimination step.
- In the exponential steps, if a path stays inside G' or inside H', both implications are trivially true. Indeed, even if the contexts are not invariant by π, they are all modified in the same way (a modification of the potential) which does not affect the $\mapsto$ rules.

So, for each possible reduction step, there are only a few short $\mapsto$-paths in H which are interesting to check. Here are some of the most interesting cases:

- In the $!P/?D$ reduction step. If $\pi(C) = ((\overline{c_i}, P), T.!_t) \mapsto ((\overline{a_i}, P.t), T) = \pi(D)$. Then, by definition of π, we have $t = \mathsf{e}$, $C = ((\overline{c_i}, P), T.!_{\mathsf{e}}) \mapsto ((\overline{a_i}, P), T)$ so $C \mapsto D$.
- In the $!P/?C$ reduction step. If $\pi(C) = ((c, P), T.!_{1(t)}) \mapsto^2 ((\overline{f}, P), T.!_t) = \pi(D)$, then $C = ((c_l, P), T.!_t)$ and $D = ((\overline{f}, P), T)$. So $C \mapsto D$.

- In the $!P/?C$ reduction step. If $C = ((a_r, P.t), T) \mapsto ((c_r, P), T.!_t) = D$ then $\pi(C) = ((a, P.\mathtt{r}(t)), T)$ and $\pi(D) = ((c, P), T.!_{\mathtt{r}(t)})$. So $\pi(C) \mapsto \pi(D)$.

- In the $!P/?C$ reduction step. If $C = ((\overline{c_i}, P), T.!_{\mathtt{l}(t)}) \mapsto^1 ((\overline{b_i^l}, P), T) = D$ then $\pi(C) = \pi(D) = ((\overline{c_i}, P), T.!_{\mathtt{l}(t)})$ so $\pi(C) \mapsto^0 \pi(D)$.

- In the $!P/?N$ reduction step. If $C = ((c, P), [!_u]) \mapsto ((\overline{f}, P), [!_u]) = D$, then $\pi(C) = ((c, P), [!_{\mathtt{p}(u)}])$ and $\pi(D) = ((\overline{f}, P), [!_{\mathtt{p}(u)}])$. So $\pi(C) \mapsto^2 \pi(D)$.

- In the $!P/?N$ reduction step. If $\pi(C) = ((\overline{c_i}, P), [!_v]) \mapsto ((c, P), [!_v]) = D$, then by definition of π either $v = \mathtt{n}(t, u)$ or $v = \mathtt{p}(u)$.

  - If $v = \mathtt{n}(t, u)$, C is equal to either $((\overline{c_i}, P), [!_{\mathtt{n}(t,u)}])$, $((\overline{e_i}, P), [!_t; !_u])$ or $((\overline{b_i}, P.u), [!_t])$. And D is equal to either $((c, P), [!_{\mathtt{n}(t,u)}])$ or $((b, P.u), [!_t])$. We can notice that $((\overline{c_i}, P), [!_{\mathtt{n}(t,u)}]) \mapsto ((\overline{e_i}, P), [!_t; !_u]) \mapsto ((\overline{b_i}, P.u), [!_t]) \mapsto ((b, P.u), [!_t]) \mapsto ((c, P), [!_{\mathtt{n}(t,u)}])$. So, in every case we have $C \mapsto^+ D$.
  - If $v = \mathtt{p}(u)$, C is equal to either $((\overline{c_i}, P), [!_{\mathtt{p}(u)}])$ or $((\overline{e_i}, P), [!_u])$. And D is equal to $((c, P), [!_{\mathtt{p}(u)}])$. We can notice that $((\overline{c_i}, P), [!_{\mathtt{p}(u)}]) \mapsto ((\overline{e_i}, P), [!_u]) \mapsto ((c, P), [!_{\mathtt{p}(u)}])$. So, in every case we have $C \mapsto^+ D$.

□

Lemma 14. *If $\pi((\sigma(B'), P'), [!_{t'}]) = ((\sigma(B), P), [!_t])$ then:*

- *For every $u' \sqsupseteq t'$, there exists $u \sqsupseteq t$ such that $\pi((\sigma(B'), P'), [!_{u'}]) = ((\sigma(B), P), [!_u])$.*

- *For every $u \sqsupseteq t$,*

  - *Either there exists $u' \sqsupseteq t'$ such that $\pi((\sigma(B'), P'), [!_{u'}]) = ((\sigma(B), P), [!_u])$.*
  - *Or there exists $C' \supset B'$, $Q'.v'@\_ = P'$ and $w' \sqsupseteq v'$ such that $\pi((\sigma(C'), Q'), [!_{w'}]) = ((\sigma(B), P), [!_u])$.*

Proof. Most of the cases are trivial. Let us consider a $!P/?C$ reduction step. We suppose that $\pi((\sigma(B_l), P), [!_t]) = ((\sigma(B), P), [!_{\mathtt{l}(t)}])$. For every $u' \sqsupseteq t$, $\mathtt{l}(u') \sqsupseteq \mathtt{l}(t)$ and $\pi((\sigma(B_l), P), [!_{u'}]) = ((\sigma(B), P), [!_{\mathtt{l}(u')}])$. For every $u \sqsupseteq \mathtt{l}(t)$, $u = \mathtt{l}(u')$ with $u' \sqsupseteq t$ and $\pi((\sigma(B_l), P), [!_{u'}]) = ((\sigma(B), P), [!_{\mathtt{l}(u')}])$.

The only interesting case is the $!P/?N$ step. Let us first suppose that $\pi((\sigma(B_e), P), [!_t]) = ((\sigma(B), P), [!_{\mathtt{p}(t)}])$. For every $u' \sqsupseteq t$, $\mathtt{p}(u') \sqsupseteq \mathtt{p}(t)$ and $\pi((\sigma(B_e), P), [!_{u'}]) = ((\sigma(B), P), [!_{\mathtt{p}(u')}])$. For every $u \sqsupseteq \mathtt{p}(t)$, $u = \mathtt{p}(u')$ with $u' \sqsupseteq t$ and $\pi((\sigma(B_e), P), [!_{u'}]) = ((\sigma(B), P), [!_{\mathtt{p}(u')}])$.

Then, let us suppose that $\pi((\sigma(B_i), P.t_2), [!_{t_1}]) = ((\sigma(B), P), [!_{\mathtt{n}(t_1,t_2)}])$. For every $u_1 \sqsupseteq t_1$, $\mathtt{n}(u_1, t_2) \sqsupseteq \mathtt{n}(u_1, t_2)$ and $\pi((\sigma(B_i), P.t_2), [!_{u_1}]) = ((\sigma(B), P), [!_{\mathtt{n}(u_1,t_2)}])$. For every $u \sqsupseteq \mathtt{n}(t_1, t_2)$,

- Either $u = \mathtt{n}(u_1, t_2)$ with $u_1 \sqsupseteq t_1$ and $\pi((\sigma(B_i), P.t_2), [!_{u_1}]) = ((\sigma(B), P), [!_{\mathtt{n}(u_1,t_2)}])$.

- Or $u = \mathtt{p}(u_2)$ with $u_2 \sqsupseteq t_2$ and $\pi((\sigma(B_e), P), [!_{u_2}]) = ((\sigma(B), P), [!_{\mathtt{p}(u_2)}])$.

□

Lemma 15. *Let us suppose that $G \to_{cut} H$. Then, for any $B' \in B_H$, there exists $B \in B_G$ such that for every $(B', P') \in Can(B')$ and $t' \in Sig$, there exist $(B, P) \in Pot(B)$ and $t \in Sig$ such that:*

$$\pi((\sigma(B'), P'), [!_{t'}]) = ((\sigma(B), P), [!_t])$$

Furthermore the mapping from (B', P', t') *to* (B, P, t) *is a injection and*

$$((\sigma(B'), P'), [!_{t'}]) \textit{ is a copy context} \Leftrightarrow ((\sigma(B), P), [!_t]) \textit{ is a copy context}$$

Proof. Examining all possible *cut*-elimination steps, one can observe that the image by π of a principal edge $\sigma(B') \in B_H$ is always a principal edge $\sigma(B) \in B_G$ which does not depend on the potential and the trace.

The fact that the mapping from (B', P', t') to (B, P, t) is an injection is also straightforward by considering the definition of π. The most complex part of the lemma is to prove that t' is a copy of (B', P') if and only if t is a copy of (B, P).

$\Rightarrow$ Let us suppose that $((\sigma(B'), P'), [!_{t'}])$ is a copy context, let us prove that $((\sigma(B), P), [!_t])$ is a copy context. Let us consider $u \sqsupseteq t$, we have to show that there exists a path $((\sigma(B), P), [!_u]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_)$. By Lemma 14:

- Either there exists $u' \sqsupseteq t'$ such that $\pi((\sigma(B'), P'), [!_{u'}]) = ((\sigma(B), P), [!_u])$. Then, by definition of copy contexts, there exists a context $((e', Q'), [!_{\mathtt{e}}]@T)$ such that $((\sigma(B'), P'), [!_{u'}]) \mapsto^* ((e', Q'), [!_{\mathtt{e}}]@T)$. We can suppose that $((e', Q'), [!_{\mathtt{e}}]@T)$ is the first context in the $\mapsto$-path to have $!_{\mathtt{e}}$ as its leftmost trace element. Thus, we are in one of the following cases:
  - $((e', Q'), [!_{\mathtt{e}}]@T) = ((\sigma(B'), P'), [!_{u'}])$, so in particular $u' = \mathtt{e}$.
    - If the reduction step is a $!P/?C$ step with box B, then $u = \mathtt{l}(e)$ or $u = \mathtt{r}(e)$. In the first case, $((\sigma(B), P), [!_u]) \mapsto^* ((\overline{f}, P), [!_{\mathtt{e}}])$, in the other case $((\sigma(B), P), [!_u]) \mapsto^* ((\overline{g}, P), [!_{\mathtt{e}}])$.
    - If the reduction step is a $!P/?N$ step with box B, then $u = \mathtt{p}(\mathtt{e})$ or $u = \mathtt{n}(\mathtt{e}, \mathtt{e})$. In both cases $((\sigma(B), P), [!_u]) \mapsto ((\overline{f}, P), [!_{\mathtt{e}}]@\_)$.
    - Else $u = \mathtt{e}$ so $((\sigma(B), P), [!_u]) \mapsto^0 ((\sigma(B), P), [!_{\mathtt{e}}])$.
  - The last $\mapsto$ step is crossing a $?C$ or a $?N$ upwards with a trace of the shape $[!_v]$. In all cases, $\pi((e', Q'), [!_{\mathtt{e}}])$ is defined and of the shape $((e, Q), [!_{\mathtt{e}}])$. By Lemma 13, $((\sigma(B), P), [!_t]) \mapsto^+ ((e, Q), [!_{\mathtt{e}}])$.
- Or, there exists $C' \supset B'$, $Q'.v'@\_ = P'$ and $w' \sqsupseteq v'$ such that $\pi((\sigma(C'), Q'), [!_{w'}]) = ((\sigma(B), P), [!_u])$. Then by definition of copies, there exists a context $((e', Q'), [!_{\mathtt{e}}]@T)$ such that $((\sigma(C'), Q'), [!_{w'}]) \mapsto^* ((e', Q'), [!_{\mathtt{e}}]@T)$. In this case, we can prove as in the previous case that $((\sigma(B), P), [!_u]) \mapsto ((\_, \_), [!_{\mathtt{e}}]@\_)$.

$\Leftarrow$ Let us suppose that $((\sigma(B), P), [!_t])$ is a copy context. Let us prove that $((\sigma(B'), P'), [!_{t'}])$ is a copy context. Let us consider $u' \sqsupseteq t'$, we have to show that there exists a path of the shape $((\sigma(B'), P'), [!_{u'}]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_)$. By Lemma 14, there exists $u \sqsupseteq t$ such that $\pi((\sigma(B'), P'), [!_{u'}]) = ((\sigma(B), P), [!_u])$. By definition of copy contexts, there exists a context $((e, Q), [!_{\mathtt{e}}]@T)$ such that $((\sigma(B), P), [!_u]) \mapsto^* ((e, Q), [!_{\mathtt{e}}]@T)$. We suppose that $((e, Q), [!_{\mathtt{e}}]@T)$ is the first context in the $\mapsto$-path to have $!_{\mathtt{e}}$ as its leftmost trace element. Thus, we are in one of the following cases:

- $((e, Q), [!_{\mathtt{e}}]@T) = ((\sigma(B), P), [!_u])$, so in particular $u = \mathtt{e}$. So $u' = \mathtt{e}$ and $((\sigma(B'), P'), [!_{u'}]) \mapsto^0 ((\_, \_), [!_{\mathtt{e}}]@\_)$.
- The last $\mapsto$ step is crossing a $?C$ or a $?N$ upwards with a trace of the shape $[!_v]$. In all cases, there exists $(e', Q') \in Pot(\vec{\mathrm{E}}_G)$ such that $\pi((e', Q'), [!_{\mathtt{e}}]@T) = ((e, Q), [!_{\mathtt{e}}]@T)$. So $((\sigma(B), P), [!_u]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}])$.

□

Lemma 16. *If $G \to_{cut} G'$, and $((e,P),T) = \pi((e',P'),T')$ then*

$$(e,P) \in Can(\vec{\mathrm{E}}_G) \Leftrightarrow (e',P') \in Can(\vec{\mathrm{E}}_{G'})$$

Proof. We prove the lemma by induction on $\partial(e)$. If $\partial(e) = 0$, then we have $\partial(e') = 0$, $P = P' = []$ so $(e,P) \in Can(\vec{\mathrm{E}}_G)$ and $(e',P') \in Can(\vec{\mathrm{E}}_{G'})$.

Else, we set $[p_1;\cdots;p_{\partial(e)}] = P$ and $[p'_1;\cdots;p'_{\partial(e')}] = P'$. Let C (resp B') be the deepest box containing e (resp. e'). Then, by definition of π, we are in one of the following cases:

- The $\to_{cut}$ step is a $!P/?D$ step involving the box C thus $p_{\partial(e)} = \mathtt{e}$ and $P' = [p_1;\cdots;p_{\partial(e)-1}]$. We have the following equivalences (with b the premise of the $?D$ node):

$$\begin{aligned}
&(e,[p_1;\cdots;p_{\partial(e)-1};\mathtt{e}]) \in Can(\vec{\mathrm{E}}_G) \Leftrightarrow (\sigma(C),[p_1;\cdots;p_{\partial(e)-1}]) \in Can(\vec{\mathrm{E}}_G) \Leftrightarrow\\
&(b,[p_1;\cdots;p_{\partial(e)-1}]) \in Can(\vec{\mathrm{E}}_G) \Leftrightarrow (b,[p_1;\cdots;p_{\partial(e)-1}]) \in Can(\vec{\mathrm{E}}_{G'}) \Leftrightarrow\\
&(e',[p_1;\cdots;p_{\partial(e)-1}]) \in Can(\vec{\mathrm{E}}_{G'})
\end{aligned}$$

- The $\to_{cut}$ step is a $!P/?N$ step on the box C thus $P = [p_1;\cdots;p_k;\mathtt{n}(t,u)]$ and $P' = [p_1;\cdots;p_k;t;u]$. We have the following equivalences

$$\begin{aligned}
(e,[p_1;\cdots;p_k;\mathtt{n}(t,u)]) \in Can(\vec{\mathrm{E}}_G) &\Leftrightarrow \begin{cases} (\sigma(C),[p_1;\cdots;p_k]) \in Can(\vec{\mathrm{E}}_G)\\ \mathtt{n}(t,u) \in Cop(B,[p_1;\cdots;p_k]) \end{cases}\\
&\Leftrightarrow \begin{cases} (\sigma(B_e),[p_1;\cdots;p_k]) \in Can(\vec{\mathrm{E}}_{G'})\\ \mathtt{n}(t,u) \in Cop(B,[p_1;\cdots;p_k]) \end{cases} \text{ by induction hypothesis}\\
&\Leftrightarrow \begin{cases} (\sigma(B_e),[p_1;\cdots;p_k]) \in Can(\vec{\mathrm{E}}_{G'})\\ ((\sigma(B),[p_1;\cdots;p_k]),[!_{\mathtt{n}(t,u)}]) \text{ is a copy context} \end{cases}\\
&\Leftrightarrow \begin{cases} (\sigma(B_e),[p_1;\cdots;p_k]) \in Can(\vec{\mathrm{E}}_{G'})\\ ((\sigma(B_i),[p_1;\cdots;p_k;u]),[!_t]) \text{ is a copy context} \end{cases} \text{ by Lemma 15}\\
&\Leftrightarrow (e',[p_1;\cdots;p_k;t;u]) \in Can(\vec{\mathrm{E}}_{G'})
\end{aligned}$$

- Else, let $[p'_1;\cdots;p'_{\partial(e')}] = P'$ and $[p_1;\cdots;p_{\partial(e)}] = P$. Then $\pi((\sigma(B'),[p'_1;\cdots;p'_{\partial(e)-1}]),[!_{p'_{\partial(e)}}])$ is defined and has the shape $((\sigma(B),[p_1;\cdots;p_{\partial(e)-1}]),[!_{p_{\partial(e)}}])$, B and C are contained in the same box and $((\sigma(C),[p_1;\cdots;p_{\partial(B)}]),[!_p]) \mapsto^* ((\sigma(B),[p_1;\cdots;p_{\partial(e)-1}]),[!_p])$. We have the following equivalences:

$$\begin{aligned}
(e,[p_1;\cdots;p_{\partial(e)}]) \in Can(\vec{\mathrm{E}}_G) &\Leftrightarrow \begin{cases} (\sigma(B),[p_1;\cdots;p_{\partial(e)-1}]) \in Can(\vec{\mathrm{E}}_G)\\ p_{\partial(e)} \in Cop(B,[p_1;\cdots;p_{\partial(e)-1}]) \end{cases}\\
&\Leftrightarrow \begin{cases} (\sigma(B'),[p'_1;\cdots;p'_{\partial(e)-1}]) \in Can(\vec{\mathrm{E}}_{G'})\\ p_{\partial(e)} \in Cop(B,[p_1;\cdots;p_{\partial(e)-1}]) \end{cases} \text{ by induction hypothesis}\\
&\Leftrightarrow \begin{cases} (\sigma(B'),[p'_1;\cdots;p'_{\partial(e)-1}]) \in Can(\vec{\mathrm{E}}_{G'})\\ p'_{\partial(e)} \in Cop(B',[p'_1;\cdots;p'_{\partial(e)-1}]) \end{cases} \text{ by Lemma 15}\\
&\Leftrightarrow (e',[p'_1;\cdots;p'_{\partial(e)}]) \in Can(\vec{\mathrm{E}}_{G'})
\end{aligned}$$

□

We can formalize the weigh V_G we described intuitively in the introduction of Section 2.2. As hinted in the introduction, Lemma 18 shows that the weight decreases along reduction. However, in the general case, V_G may be infinite. If G is a *LL* proof-net, then G is strongly normalizing and we will prove that, because G is strongly normalizing, V_G is finite. To do so, we will reduce the $!P/?W$ cuts last.

Definition 17. *For any proof-net G, we define weights V_G and W_G in $\mathbb{N} \cup \{\infty\}$ by $V_G = \sum_{n \in \mathrm{N}_G} |Can(n)|$ and $W_G = 2 \cdot \sum_{e \in \mathrm{E}_G} |Can(e)| = \sum_{e \in \vec{\mathrm{E}}_G} |Can(e)|$.*

Theorem 18. *If $G \rightarrow_{cut} H$, then $V_G \geq V_H + 1$. If, moreover, V_H is finite and (if this is a $!P/?W$ step then all cuts are $!P/?W$ cuts) then V_G is finite.*

Proof. The proof depends on the scheme of *cut*-elimination used for the reduction from G to H. From Lemma 16 and the definition of π, we can deduce that the nodes which are not represented in the scheme have exactly the same canonical potentials in G and in H. The same goes for the pending edges of the schemes of *cut*-elimination. Thus, it is enough to show the lemma supposing this scheme is at depth 0.

We write c the *cut* node we reduce. In each case, we will define an injection ϕ from the canonical nodes of H to the canonical nodes of G such that $1 \leq |Can(\mathrm{N}_G) - \mathrm{Codom}(\phi)|$ (this proves that $V_G \geq V_H + 1$) and if V_H is finite then $|Can(\mathrm{N}_G) - \mathrm{Codom}(\phi)| < \infty$ (this proves that V_G is finite). The fact that the image of the mappings ϕ we define is indeed included in $Can(\mathrm{N}_G)$, and the claims we do on the value of $Can(\mathrm{N}_G) - \mathrm{Codom}(\phi)$ are based on Lemma 16.

In the case of a $!P/?W$ reduction, let $a_1, \cdots, a_k$ be the auxiliary doors of the box deleted in the step and $w_1, \cdots, w_k$ the $?W$ nodes created. We define $\phi(w_i, P) = (a_i, P)$ and $\phi(n, P) = (n, P)$ otherwise. So $Can(\mathrm{N}_G) - \mathrm{Codom}(\phi) = \{Can(n) \mid n \in \mathrm{N}_{G'}\}$, $1 \leq |Can(\mathrm{N}_G) - \mathrm{Codom}(\phi)|$. Moreover, if we suppose that the premises of every *cut* node are $!P$ and $?W$ nodes. Then for every node n of G, $Can(n) = \{(n, [\mathtt{e}; \cdots; \mathtt{e}])\}$, so $|Can(\mathrm{N}_G) - \mathrm{Codom}(\phi)| \leq |\mathrm{N}_G| < \infty$.

In the following steps, we will prove that $1 \leq |Can(\mathrm{N}_G) - \mathrm{Codom}(\phi)| \leq k \cdot |V_H|$ for some $k \in \mathbb{N}$. Thus, if V_H is finite, so is $|Can(\mathrm{N}_G) - \mathrm{Codom}(\phi)|$.

In the case of the reduction of an *ax* node a, we define ϕ by $\phi(n, P) = (n, P)$. Then, $Can(\mathrm{N}_G) - \mathrm{Codom}(\phi) = Can(c) \cup Can(a)$. So $|Can(\mathrm{N}_G) - \mathrm{Codom}(\phi)| \geq 2$. Because there is at least one node in H included in the same boxes as c, $|Can(c)| \leq 2 \cdot |V_H|$.

In the case of a $⅋/\otimes$ step, let p and t be the $⅋$ and $\otimes$ nodes of G which are deleted in the step and c_1, c_2 be the *cut* nodes which are created during the step. Then, we define ϕ by $\phi(c_1, P) = (p, P)$, $\phi(c_2, P) = (t, P)$ and $\phi(n, P) = (n, P)$ otherwise. Thus $|Can(\mathrm{N}_G) - \mathrm{Codom}(\phi)| = |Can(c)|$, so $1 \leq |Can(\mathrm{N}_G) - \mathrm{Codom}(\phi)| \leq V_H$. The $\forall/\exists$ step is similar.

For the $!P/?P$ scheme, let p be the $!P$ node and a be the $?P$ node deleted during the step and c' be the *cut* node created during the step. We define ϕ by $\phi(c', P) = (b, P)$ and $\phi(e, P) = (e, P)$ otherwise. Then, $Can(\mathrm{N}_G) - \mathrm{Codom}(\phi) = Can(c) \cup Can(a) \cup Can(p)$. So $3 \leq |Can(\mathrm{N}_G) - \mathrm{Codom}(\phi)| \leq 3 \cdot Can(c') \leq 3 \cdot V_H$.

For the $!P/?D$ scheme, let $a_1, \cdots, a_k, p, d$ be the auxiliary ports, principal port and $?D$ node deleted in the step, and let $d_1, \cdots, d_k, c'$ be the $?D$ nodes and *cut* node created in the step. We define ϕ by $\phi(c', P) = (c, P)$, $\phi(d_i, P) = (a_i, P)$, $\phi(n, P) = (n, P.\,\mathtt{e})$ (if n is a node of G') and $\phi(n, P) = (n, P)$ otherwise.

For the $!P/?N$ scheme, let p (resp. p_i, p_e) be the principal door of B (resp. B_i, B_e), let c' be the *cut* node created by the step. We set $\phi(c', []) = (c, [])$, $\phi(p_i, [t; u]) = (p, [\mathtt{n}(u, t)])$ and $\phi(p_e, [t]) = (p, [\mathtt{p}(t)])$.

- For every $(n, P) \in Can_{G'}$, we set $\phi(n, [t; u]@P) = (n, [\mathtt{n}(u, t)]@P)$.
- For every $0 \leq j \leq k$, if m_j (resp. m_j^i, m_j^e) is the j-th auxiliary door of B (resp. B_i, B_e) and n_j is the j-th $?N$ node of H, we set: $\phi(m_j^i, [t; u]) = (m_j, [\mathtt{n}(u, t)])$, $\phi(m_j^e, [t]) = (m_j, [\mathtt{p}(t)])$ and $\phi(n_j, []) = (m_j, [\mathtt{e}])$.

We can notice that $Can(\mathrm{N}_G) - \mathrm{Codom}(\phi) = \{(n, \mathtt{[e]}@P) \in Can(n) \mid n \in G'\} \cup \{(p, \mathtt{[e]}), (n, [])\}$ so we have $1 \leq |Can(\mathrm{N}_G) - \mathrm{Codom}(\phi)| \leq V_H$.

For the $!P/?C$ scheme, let c_l and c_r be the cuts on the right side, c be the cut on the left side and d the $?C$ node on the left side,

- We set $\phi(c_l, []) = (c, [])$ and $\phi(c_r, []) = (d, [])$.
- For $(n_l, [t]@P) \in Can_{G'_l}$ we set $\phi(n_l, [t]@P) = (n, [\mathtt{l}(t)]@P)$ and for $(n_r, [t]@P) \in Can_{G'_r}$ we set $\phi(n_r, [t]@P) = (n, [\mathtt{r}(t)]@P)$.
- For every $0 \leq i \leq k$, let us name c_i the i-th contraction node. We set $\phi(\sigma_i(B_l), [t]) = (\sigma_i(B), [\mathtt{l}(t)])$, $\phi(\sigma_i(B_r), [t]) = (\sigma_i(B), [\mathtt{r}(t)])$ and $\phi(c_i, []) = (\sigma_i(B), \mathtt{[e]})$.

We can notice that $Can(\mathrm{N}_G) - \mathrm{Codom}(\phi) = \{(n, \mathtt{[e]}@P) \in Can(n) \mid n \in G'\}\{(p, \mathtt{[e]}), (n, [])\}$ so we have $1 \leq |Can(\mathrm{N}_G) - \mathrm{Codom}(\phi)| \leq V_H$. □

Corollary 19. *If G is a strongly normalizing proof-net, then $V_G \in \mathbb{N}$, and the length of any path of reduction is bounded by V_G*

Proof. Let us suppose that G is a proof-net. We consider a reduction sequence $G \to_{cut} G_1 \to_{cut} \cdots \to_{cut} G_m$. We can observe that it is possible to commute rules such that the $!P/?W$ reductions happen last (an observation also used in [20] with the definition of the $\Longrightarrow$ reduction strategy). In particular, there exists a reduction sequence $G \to_{cut} H_1 \to_{cut} \cdots \to_{cut} H_n \not\to_{cut}$ such that $m \leq n$ and, if one of the $H_i \to_{cut} H_{i+1}$ steps is a $!P/?W$, then every *cut* nodes of H_i is a $!P/?W$ *cut*. We know that H_n is in normal form so, by definition of V_{H_n}, $V_{H_n} = |\{(n, \mathtt{[e;\cdots;e]}) \mid n \in \mathrm{N}_{H_n}\}| = |\mathrm{N}_{H_n}|$ which is finite. So, according to Theorem 18, we can prove by induction on $n - i$ that V_{H_i} is finite and $V_G > V_{H_1} > \cdots > V_{H_n} \geq 0$. So $m \leq n < V_G$. □

Theorem 20. *If G is a strongly normalizing proof-net, then $W_G \in \mathbb{N}$, and the length of any path of reduction is bounded by W_G*

Proof. By Corollary 19, it is enough to prove that $W_G = 2 \cdot |Can(\mathrm{E}_G)| \geq |Can(\mathrm{N}_G)| = V_G$. In fact, we will prove by induction on subproof-nets H of G that:

$$2 \cdot \sum_{e \in \mathrm{E}_H} |Can(e)| \geq \sum_{n \in \mathrm{N}_H} |Can(n)| + \Big|\{\text{pending edges of } H\}\Big| \cdot \min_{e \in \mathrm{E}_H} |Can(e)|$$

Where we are referring to the canonical potentials in the proof-net G (the edges and nodes of H are edges and nodes in G). The main ingredient is that, if every box containing the node n contains the edge e, then $|Can(e)| \geq |Can(n)|$.

The inequality stands in the case of axioms because there are two edges, one node and two pending edges (and for every x, $2 \cdot 2 \cdot x \geq x + 2 \cdot x$). Otherwise, we consider a subproof-net H, such that the inequality stands for the strict sub-proofnets H' of H. When we add a $?W$ node n then we also add an edge e with $|Can(e)| = |Can(n)|$ and there is one more pending edge. By induction hypothesis and because $\forall x \in \mathbb{R}, 2 \cdot x \geq x + x$, the inequation stands for H. When we add a $\forall$, $\exists$, $\otimes$, $\parr$, $?D$, $?N$, or $?C$ node n then it is the same as in the $?W$ case and the gap is larger because we do not add pending edges. When we add a *cut* node, we add one node but compensate it by deleting two pending edges.

The most interesting case is whenever H is constructed by creating a box B around a sub-proofnet H'. In this case, we add k nodes $n_1, \cdots, n_k$ (the doors) and k edges $e_1, \cdots, e_k$ with $|Can(n_1)| = \cdots = |Can(n_k)| = \min_{e \in \mathrm{E}_{H'}} |Can(e)|$ $|Can(e_1)| = \cdots = |Can(e_k)| = \min_{e \in \mathrm{E}_H} |Can(e)|$, and H and H' have both k pending edges.

$$
\begin{aligned}
2 \cdot \sum_{e \in \mathrm{E}_H} |Can(e)| &\geq 2 \cdot k \cdot |Can(B)| + 2 \cdot \sum_{e \in \mathrm{E}_{H'}} |Can(e)| \\
&\geq 2 \cdot k \cdot |Can(B)| + \sum_{n \in \mathrm{N}_{H'}} |Can(n)| + \quad \left|\{\text{pending edges of } H'\}\right| \cdot \min_{e \in \mathrm{E}_{H'}} |Can(e)| \\
&\geq 2 \cdot k \cdot |Can(B)| + \sum_{n \in \mathrm{N}_{H'}} |Can(n)| + \quad k \cdot \min_{e \in \mathrm{E}_{H'}} |Can(e)| \\
&\geq 2 \cdot k \cdot |Can(B)| + \sum_{n \in \mathrm{N}_H} |Can(n)| - k \cdot \min_{e \in \mathrm{E}_{H'}} |Can(e)| + \quad k \cdot \min_{e \in \mathrm{E}_{H'}} |Can(e)| \\
&> \sum_{n \in \mathrm{N}_H} |Can(n)| + k \cdot |Can(B)| \\
2 \cdot \sum_{e \in \mathrm{E}_H} |Can(e)| &> \sum_{n \in \mathrm{N}_H} |Can(n)| + \left|\{\text{pending edges of } H\}\right| \cdot \min_{e \in \mathrm{E}_H} |Can(e)|
\end{aligned}
$$

□

Lemma 21. *Let G be a normalizing proof-net, there is no path of the shape $((e,P),[!_t]) \mapsto^+ ((e,P),[!_u])$ with (e,P) a canonical edge.*

Proof. First we will prove it on proof-nets where all *cut* nodes are $!P/?W$. Then we will prove that if there is such a path in a proof-net G and $G \to_{cut} H$ (and this is not a $!P/?W$ step) then there is such a path in H.

Let G be a proof-net where all the *cut* nodes are $!P/?W$, and let us suppose that $((e,P),[!_t]) \mapsto^+ ((e,P),[!_u])$. If e is the conclusion of a $?W$ node, then $\not\mapsto ((e,P),[!_u])$, which is a contradiction. Else, if e is downward and all the edges of the path are downwards, then e is strictly under itself in the proof-net which is impossible. Else if e is downward and there is an upward edge in the path, then the first upward edge of the path is the conclusion of a $?W$ node so the path stops there, which is a contradiction because the path stops at e, which we supposed downward. If $\overline{e}$ is the conclusion of a $?W$ node, then $((e,P),[!_t]) \not\mapsto$ which is a contradiction. Else, if e is upward and all the edges of the path are upwards, then e is strictly above itself in the proof-net which is impossible. Else if e is upward and there is a downward edge in the path, then the last downward edge of the path is the conclusion of a $?W$ node so the path starts there, which is a contradiction because the path starts at e, which is a upward.

Let us suppose that $G \to_{cut} H$ and there exist $(e,P) \in Pot(\vec{\mathrm{E}}_G)$ and $t,u \in Sig$ such that $((e,P),[!_t]) \mapsto^+ ((e,P),[!_u])$.

1. Let us suppose that the $G \to_{cut} H$ reduction is a $!P/?C$ step involving a box B and either $e = \sigma(B)$ or $\overline{e}$ is the other premise of the *cut* node. Then the $\mapsto$-path is of the shape $((e,P),[!_t]) \mapsto^+ ((\overline{\sigma_i(B)},P),[!_u]) \mapsto^* ((e,P),[!_u])$. Thus $((\overline{\sigma_i(B)},P),[!_t]) \mapsto^+ ((\overline{\sigma_i(B)},P),[!_u])$ is a path of G. Let c_i be the conclusion of the i-th $?C$ node created in this reduction step. Then $\pi_{G\to H}(((\overline{c_i},P),[!_t])) = ((\overline{\sigma_i(B)},P),[!_t])$ and $\pi_{G\to H}(((\overline{c_i},P),[!_u])) = ((\overline{\sigma_i(B)},P),[!_u])$ so by Lemma 13, we can deduce that $((\overline{c_i},P),[!_t]) \mapsto^+ ((\overline{c_i},P),[!_u])$ is a path of H.

2. Let us suppose that the $G \to_{cut} H$ reduction is a $!P/?N$ step involving a box B and either $e = \sigma(B)$ or $\overline{e}$ is the other premise of the *cut* node. We prove similarly to the previous case, that there exists a conclusion n_i of a $?N$ node such that $((\overline{n_i},P),[!_t]) \mapsto^+ ((\overline{n_i},P),[!_u])$ is a path of H.

3. Else, if $((e,P),[!_t])$ or $((e,P),[!_u])$ is in $\mathrm{Codom}(\pi_{G\to H}(\_))$ they both are and there exists $(e',P') \in Pot(\vec{\mathrm{E}}_H)$ and $t',u' \in Sig$ such that $\pi_{G\to H}(((e',P'),[!_{t'}])) = ((e,P),[!_t])$ and $\pi_{G\to H}(((e',P'),[!_{u'}])) = ((e,P),[!_u])$. We can conclude by Lemma 13.

4. Else, if the $G \to_{cut} H$ reduction is a ax step e is one of the premise of the *cut*, then either $((a, P), [!_t]) \mapsto^+ ((a, P), [!_u])$ or $((\overline{a}, P), [!_t]) \mapsto^+ ((\overline{a}, P), [!_u])$ (with a the conclusion of the ax node which is not a premise of the reduced cut). So we can use point 3 of this list to conclude.

5. Else, if the $G \to_{cut} H$ reduction is a $\otimes/\parr$ or $\forall/\exists$ step, e can not be one of the premises of the *cut* because $((e, P), [!_t])$ can not cross a $\parr$, $\otimes$, $\forall$ or $\exists$ node upwards.

6. Else, if the $G \to_{cut} H$ reduction is a $!P/?P$ step between the principal door of B and an auxiliary door of C and e is one of the premise of the *cut* involved, then $((\sigma(C), P), [!_t]) \mapsto^+ ((\sigma(C), P), [!_u])$ so we can use point 3 of this list to conclude.

7. Else, if the $G \to_{cut} H$ reduction is a $!P/?D$ involving a box B, e can not be one of the premises of the *cut* because $((e, P), [!_t])$ can not cross a $?D$ or $!P$ node upwards. So e is in B and $P = Q.v@R$ with $|Q| = \partial(B)$ and $v \neq \mathtt{e}$. This would contradict our hypothesis that (e, P) is canonical.

8. Let us suppose that the $G \to_{cut} H$ is a $!P/?C$ involving a box B, $e \in B$ and $P = Q.v@R$ with $|P| = \partial(B)$. Then v is not of the shape $\mathtt{e}$, $\mathtt{l}(\_)$ or $\mathtt{r}(\_)$. This contradicts our assumption that (e, P) is canonical.

9. Let us suppose that the $G \to_{cut} H$ is a $!P/?N$ involving a box B, $e \in B$ and $P = Q.v@R$ with $|P| = \partial(B)$. Then v is not of the shape $\mathtt{e}$, $\mathtt{p}(\_)$ or $\mathtt{n}(\_, \_)$. This contradicts our assumption that (e, P) is canonical.

□

Finally, as we defined (quasi)-standard contexts and proved their stability by the $\mapsto$ relation to be able to consider only contexts of this shape, we will define canonical contexts based on the definition of canonical edges and copies.

Definition 22. *A quasi-standard context $C = ((e, P), [T_1; \cdots; T_k])$ is* canonical *if $(e, P) \in Can(e)$ and:*

- *For every $T_i = !_t$, and $u \sqsupseteq t$, $((e, [P_1; \cdots; P_{\partial(e)}]), [!_u; T_{i+1}; \cdots; T_k]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_)$*
- *For every $T_i = ?_t$, and $u \sqsupseteq t$, $((\overline{e}, [P_1; \cdots; P_{\partial(e)}]), [!_u; T_{i+1}^\perp; \cdots; T_k^\perp]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_)$*

Lemma 23. *If $C \mapsto D$, then C is canonical if and only if D is canonical.*

Proof. We set $((e, P), T) = C$ and $((f, Q), U) = D$. If C is canonical then C is quasi-standard so, by Lemma 7, D is quasi-standard. Conversely, if D is canonical, D is quasi-standard so C is quasi-standard. We will prove the other points.

- Let us suppose that the step from C to D does not cross an exponential node (i.e. a node labelled by $?D$, $?C$, $?N$, $?P$ and $!P$). Then, in particular, the step neither enters nor leaves a box, so C and D are of the shape $((e, P), \_)$ and $((f, P), \_)$. Thus (e, P) is canonical if and only if (f, P) is canonical. Verifying the conditions on the trace is straightforward. As an example, let us examine the case where $C = ((e, P), [T_1; \cdots; T_k]) \mapsto ((f, P), [T_1; \cdots; T_k; \parr_l])$ crossing a $\parr$ node downwards. Then we have the following equivalences:

  $$((e, P), [!_u; T_{i+1}; \cdots; T_k]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_) \Leftrightarrow ((f, P), [!_u; T_{i+1}; \cdots; T_k; \parr_l]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_)$$
  $$((\overline{e}, P), [!_u; T_{i+1}^\perp; \cdots; T_k^\perp]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_) \Leftrightarrow ((\overline{f}, P), [!_u; T_{i+1}^\perp; \cdots; T_k^\perp; \otimes_l]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]@\_)$$

  Similarly, even in the case of exponential $\mapsto$ steps, proving the conditions on trace elements is straightforward except for the rightmost ones.

- If $C = ((\overline{\sigma_i(B)}, P), U.!_t) \mapsto ((f, P.t), U) = D$ (entering a box by an auxiliary door), then:

$$P.t \in Can(f) \Leftrightarrow P \in Can(\sigma(B)) \wedge t \in Cop(B, P) \quad \Leftrightarrow \begin{cases} P \in Can(\sigma(B)) \\ \forall u \sqsupseteq t, ((\sigma(B), P), [!_u]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]) \end{cases}$$

$$P.t \in Can(f) \Leftrightarrow \begin{cases} P \in Can(\sigma(B)) \\ \forall u \sqsupseteq t, ((\overline{\sigma_i(B)}, P), [!_u]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]) \end{cases}$$

- If $C = ((e, Q.t), U) \mapsto ((\sigma_i(B), Q), U.?_t) = D$ (leaving a box by an auxiliary door), then:

$$Q.t \in Can(f) \Leftrightarrow Q \in Can(\sigma(B)) \wedge t \in Cop(B, Q) \quad \Leftrightarrow \begin{cases} Q \in Can(\sigma(B)) \\ \forall u \sqsupseteq t, ((\sigma(B), Q), [!_u]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]) \end{cases}$$

$$Q.t \in Can(f) \Leftrightarrow \begin{cases} Q \in Can(\sigma(B)) \\ \forall u \sqsupseteq t, ((\overline{\sigma_i(B)}, Q), [!_u]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]) \end{cases}$$

- If $C = ((e, P), [!_{\mathtt{1}(t)}]) \mapsto ((f, P), [!_t]) = D$ (crossing a $?C$ node upwards) then we can notice that the simplifications of $\mathtt{1}(t)$ are the signatures of the shape $\mathtt{1}(u)$ with u a simplification of t. Then the lemma follows because we have the following equivalence:

$$((e, P), [!_{\mathtt{1}(u)}]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}]) \Leftrightarrow ((f, P), [!_u]) \mapsto^* ((\_, \_), [!_{\mathtt{e}}])$$

- The other exponential steps are similar.

□

2.3.1 Comparison with Dal Lago's context semantics

The definition of copies does not correspond exactly to the copies defined by Dal Lago [20] (which we will refer to as *maximal copies* in this section). In fact, by our definition, $t \in Sig$ is a copy of (B, P) if and only if there exists a maximal copy u of (B, P) such that t is a pruning of u: t can be obtained by replacing some branches by $\mathtt{e}$. The notion of "pruning" will be formally captured by the relation $\blacktriangleleft$ defined in Definition 54, section 3.2.2.

The change of the definition of copies had two major benefits:

- Our definition of copies is simpler and our weight W_G is simpler than the weight T_G defined by Dal Lago. Thus, they are both easier to understand and to work with.
- In the general case, $W_G \geq T_G$, which may seem a disadvantage (the bound given by T_G is tighter than the bound given by W_G). However, we are never interested in computing the exact value of W_G or T_G. In [20] and this thesis, we make some assumptions on G, and prove bounds $T'_G \geq T_G$ or $W'_G \geq W_G$ based on those assumptions. And in most cases, using the same techniques, we get a better bound on W_G than on T_G (i.e. $W'_G \leq T'_G$).

In this section, we include a definition of maximal copies. It is equivalent to Dal Lago's definition of copies. For the readers familiar with the definitions of Dal Lago, it may ease the understanding of ours. First we consider a relation $\rightarrow$ on contexts and define notions of $\rightarrow$-reducible contexts, $\rightarrow$-normal contexts and $\rightarrow$-maximal contexts. In this section we only use these notions with $\rightarrow = \mapsto$, the general definition will be needed in the next sections.

Definition 24 (normal contexts)**.** *Let* $((e,P),[!_t]@T)$ *be a context of* G*,*

- $((e,P),[!_t]@T)$ *is* $\to$-reducible *if there exists paths of the following shape (with* $(f_1,|U_1|)\neq(f_2,|U_2|)$*)*

$$((e,P),[!_{t_1}]@T)\to^k((f_1,Q_1),U_1)$$
$$((e,P),[!_{t_2}]@T)\to^k((f_2,Q_2),U_2)$$

- *Else,* $((e,P),[!_t]@T)$ *is said* $\to$-normal

Definition 25. *A context* C *is said* $\to$*-maximal if for every context* D*,* $(C\to^* D\not\to)\Rightarrow$ *D is* $\to$*-normal.*

Definition 26. *Let* (B,P) *be a potential box, the maximal copies of* (B,P) *are the elements of:*

$$MaxCop(B,P)=\{t\in Cop(B,P)\mid\forall u\sqsupseteq t,((\sigma(B),P),[!_u])\ is\ \mapsto\text{-}maximal\}$$

Let x *be an element (box, node or edge) with* $x\in B_{\partial(x)}\subset\cdots\subset B_1$*, the maximal canonical potentials for* x *are the elements of the following set:*

$$MaxCan(x)=\left\{(B,[P_1;\cdots;P_{\partial(x)}])\;\middle|\;\forall 1\leq i\leq\partial(x),P_i\in MaxCop(B_i,[P_1;\cdots;P_{i-1}])\right\}$$

Finally, the weight T_G is defined as follows:

$$T_G=\sum_{n\in \mathrm{N}_G-\mathrm{N}_G^{?P}-\mathrm{N}_G^{!P}}|MaxCan(n)|+\sum_{(B,P)\in MaxCan(B_G)}\left(\Big|\{\text{auxiliary doors of }B\}\Big|\cdot\sum_{t\in MaxCop(B,P)}|t|\right)$$

Chapter 3

Paths criteria for elementary and polynomial complexity

3.1 An introduction to paths criteria

3.1.1 History and motivations

A stratification refers to a restriction of a framework, which forbids the contraction (or identification) of two subterms belonging to two morally different "strata". Russell's paradox in naive set theory relies on the identification of two formulae which belong morally to different strata. The non-terminating λ-term $(\lambda x.(x)x)\lambda y.(y)y$ relies on the identification of an argument with the function duplicating it. In recursion theory, to create from the elementary sequences $\theta_m(n) = 2^n_m$ (tower of exponential of height m in n), the non elementary sequence $n \mapsto 2^n_n$, we also need to identify n and m which seem to belong to different strata. Stratification restrictions might be applied to those frameworks (naive set theory, linear logic, lambda calculus and recursion theory) to entail coherence or complexity properties [8].

Let us consider more precisely the cases of λ-calculus and Linear Logic. To define a stratification condition, one has to define, for every term t, a *stratification relation* $>$ between the subterms of t. Then, we will consider that u belongs to a higher stratum than v if $u(>)^+v$. The relation $>$ must be defined such that the number of residues of every subterm u of t is bounded by the maximum number of residues of subterms in lower strata. It is to say there exists a function f such that:

$$|Residues(u)| \leq f\left(\max_{u>v} |Residues(v)|, |t|\right) \tag{3.1}$$

One says that t is $>$-stratified if $>$ is acyclic. If t is $>$-stratified, then for every subterm u of t, one defines the $>$-stratum of u as the depth of u for the relation $>$ (written $s_>(u)$). This depth is formally defined as follows:

Definition 27. *Let S be a set and $>$ be a relation on S, for any $e \in S$, we define $s_>(e)$ as the greatest $i \in \mathbb{N} \cup \{\infty\}$ such there exists $e_2, \cdots, e_i \in S$ such that $e > e_2 > e_3 > \cdots > e_i$. We define $|>|$ as $\max_{e\in S} s_>(e)$.*

We can notice that, by definition, for any relation $>$ on S and $e \in S$, $s_>(e) \geq 1$. To keep the notations coherent, whenever we consider a pre-order R, if aRb, we say that a is *smaller* than b for R. Thus, if $s_>(e) = 1$, then e is *maximal* for the order $>$. Usually, we use the reverse symbol to denote the inverse relation: for instance $e < f$ if and only if $f > e$. Thus, e is maximal for the order $>$ if and only if it is minimal for the order $<$.

If t is $>$-stratified, the $>$-stratum of every subterm is in $\mathbb{N}$ because the number of subterms is finite. One can bound the number of residues of subterms u of t by induction on $s_>(u)$.

In most previous works, the stratum $s(\ )$ is rather explicit while $>$ is left implicit (it can be defined by "$u > v$ iff $s(u) > s(v)$"). Concretely, in [38] and [24], the stratum of a box is defined as its depth (the number of boxes containing it). To enforce Equation 3.1, digging and dereliction ($?N$ and $?D$ nodes) are forbidden. In [8], Baillot and Mazza label the edges with their strata. To enforce Equation 3.1, Baillot and Mazza define some local conditions that those labels have to satisfy. Those works are presented as subsystems of Linear Logic: ELL [38] and L^3 [8]. In both cases, the function f in Equation 3.1 is an elementary function (tower of exponential of fixed height), thus ELL and L^3 proof-nets normalize in a number of steps bounded by an elementary function of its size, and this function only depends on $\max_{B\in B_G} s(B) \leq |B_G|$.

In this work, we will consider several relations, whose acyclicity entail complexity bounds (elementary time, polynomial time and primitive recursive). We want to find characterizations of complexity classes which are as intensionally expressive as possible. So we will try to find the smallest possible relations $>$ (with respect to inclusion) whose acyclicity entails a bound of the shape of Equation 3.1. Indeed if for every proof-net $\succ \subseteq >$, then the acyclicity of $>$ implies the acyclicity of $\succ$. So more proof-nets are $\succ$-stratified than

>-stratified. As the relations become smaller, the proofs that they entail the wanted complexity bounds tend to become very complex. This is why we will first study simple examples.

For any λ-term t, we will define a relation $\twoheadrightarrow_\lambda$ on variables of t. We will prove that, if $\twoheadrightarrow_\lambda$ is acyclic, then t normalizes in an elementary number of β-reduction steps. Then we will define a similar relation on proof-nets.

3.1.2 Stratification on λ-calculus

The λ-terms of λ-calculus are generated by the following grammar, where x ranges over a countable set of variables:

$$\Lambda = x \mid \lambda x.\Lambda \mid (\Lambda)\Lambda$$

For $t \in \Lambda$, $|t|$ is the size of the construction of t: for every variable x, and $t, u \in \Lambda$ we set $|x| = 1$, $|\lambda x.t| = 1 + |t|$ and $|(t)u| = |t| + |u|$.

Throughout this thesis, whenever we will define a set "something" by induction, hole-something will refer to the elements of something where a subterm has been replaced by $\circ$. For instance, hole-λ-terms will refer to the set $\Lambda_\circ$ with $\Lambda_\circ$ defined by:

$$\Lambda_\circ = \circ \mid \lambda x.\Lambda_\circ \mid (\Lambda_\circ)\Lambda \mid (\Lambda)\Lambda_\circ$$

Then, if h is a hole-something and t is a something, then $h[t]$ refers to the something obtained by replacing $\circ$ by t in h. For instance $(\lambda x.(\circ)x)[y] = \lambda x.(y)x$. In the literature, hole-somethings are often referred-to as contexts. However, in this thesis, we give another meaning to the word "context".

To differentiate the different occurrences of a variable, we will use indexes. For example, we write $\lambda x\lambda y.(x_1)(x_2)(x_3)y$ for $\lambda x.\lambda y.(x)(x)(x)y$. Hole-$\lambda$-terms allow us to isolate one of the variable occurrences: $t = h[x]$ means that x is an occurrence of variable of t.

We define a relation $\to_\beta$, named β-reduction, on Λ by $h[(\lambda x.t)u] \to_\beta h[t[u/x]]$ for every $h \in \Lambda_\circ$. If $h[x] \to_\beta^* t'$, the *residues* of x are the copies of x, by β-reduction. If x' is a residue of x then x is the *lift* of x'. We give an example to guide the intuition: let us consider $t = (\lambda x.\lambda y.((y)x_1)x_2)\lambda z.z \to_\beta \lambda y.((y)\lambda z.z_1)\lambda z.z_2 = t'$, then the residues of z are z_1 and z_2. The residue of y is the only occurrence of y in t'. Finally x_1 and x_2 do not have residues in t'.

We want to give a condition on λ-terms entailing a bound on the length of $\to_\beta$ normalization sequences. In Section 3.1.1, we wrote that the non-terminating λ-term $\Omega = (\lambda x.(x)x)\lambda y.(y)y$ relies on the identification of an argument with the function duplicating it: $(\lambda x.(x)x)\lambda y.(y)y$ reduces to $(\lambda y.(y)y)\lambda y.(y)y$. So the "function" $(\lambda y.(y)y)$ and its argument $\lambda y.(y)y$ are residues of the same subterm of Ω. To prevent this, we define a relation $\twoheadrightarrow_\lambda$ on variables such that $x \twoheadrightarrow_\lambda y$ if the variable x will be in the argument of a function $\lambda y._-$, and we will require $\twoheadrightarrow_\lambda$ to be acyclic.

Definition 28. *Let $t \in \Lambda$ and x, y be occurrences of variables in t, $x \twoheadrightarrow_\lambda y$ if $t \to_\beta^* g[(\lambda y.h_1[y'])h_2[x']] = t'$ (with x' and y' residues of x and y). A λ-term t is said $\twoheadrightarrow_\lambda$-stratified if the relation $\twoheadrightarrow_\lambda$ is acyclic on the occurrences of variables of t.*

We will analyse the term $t = (\lambda y.\,(\lambda w.(w)(w)y)\,(\lambda x.(x)(x)y)\,z)\,z$. We have the following reduction (for each step, the λ being reduced is written in bold):

$$\begin{aligned}(\lambda y.\,(\lambda w.(w_1)(w_2)y_1)\,(\boldsymbol{\lambda x}.(x_1)(x_2)y_2)\,z_1)\,z_2 &\to_\beta \left(\lambda y.\,(\boldsymbol{\lambda w}.(w_1)(w_2)y_1)\,(z_1^1)(z_1^2)y_2\right)z_2\\ &\to_\beta \left(\boldsymbol{\lambda y}.((z_1^3)(z_1^4)y_2^1)((z_1^5)(z_1^6)y_2^2)y_1\right)z_2\\ &\to_\beta \left(((z_1^3)z_1^4)z_2^1\right)\left(((z_1^5)z_1^6)z_2^2\right)z_2^3\end{aligned}$$

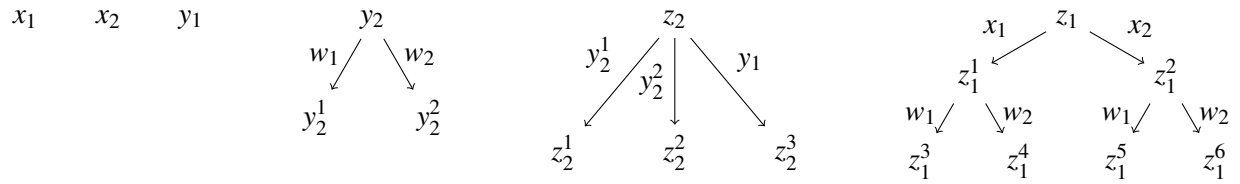


Figure 3.1: Trees of residues for $(\lambda y.\,(\lambda w.(w)(w)y)\,(\lambda x.(x)(x)y)\,z)\,z$.

The relation $\twoheadrightarrow_\lambda$ is exactly: $\{z_2 \twoheadrightarrow_\lambda y_1, z_2 \twoheadrightarrow_\lambda y_2, z_2 \twoheadrightarrow_\lambda w_1, z_2 \twoheadrightarrow_\lambda w_2, z_1 \twoheadrightarrow_\lambda x_1, z_1 \twoheadrightarrow_\lambda x_2, z_1 \twoheadrightarrow_\lambda w_1, z_1 \twoheadrightarrow_\lambda w_2, y_2 \twoheadrightarrow_\lambda w_1, y_2 \twoheadrightarrow_\lambda w_2, x_1 \twoheadrightarrow_\lambda w_1, x_1 \twoheadrightarrow_\lambda w_2, x_2 \twoheadrightarrow_\lambda w_1, x_2 \twoheadrightarrow_\lambda w_2\}$. In this $\to_\beta$ sequence there is no term of the shape $g[(\lambda w.h_1[w_1'])h_2[z_2']]$ with w_1' and z_2' residues of w_1 and z_2, but we have $z_2 \twoheadrightarrow_\lambda w_1$ nonetheless because of another $\to_\beta$ path (the situation is similar for $z_2 \twoheadrightarrow_\lambda w_2$).

Thus, t is $\twoheadrightarrow_\lambda$-stratified and we have $s_{\twoheadrightarrow_\lambda}(w_1) = s_{\twoheadrightarrow_\lambda}(w_2) = s_{\twoheadrightarrow_\lambda}(y_1) = 1$, $s_{\twoheadrightarrow_\lambda}(x_1) = s_{\twoheadrightarrow_\lambda}(x_2) = s_{\twoheadrightarrow_\lambda}(y_2) = 2$ and $s_{\twoheadrightarrow_\lambda}(z_1) = s_{\twoheadrightarrow_\lambda}(z_2) = 3$. We can bound the number of residues of every variable by induction on their $\twoheadrightarrow_\lambda$-stratum. The variables whose stratum is 1 (w_1, w_2 and y_1) can not be in the argument of a reduction, so they can not be duplicated, and they have exactly one residue. Else, the residues of an occurrence of a variable a form a tree where each node corresponds to a substitution (the trees are displayed in Figure 3.1). Each substitution is of the shape $t \to_\beta^* g[(\lambda b.h_1[b'])h_2[a']] \to_\beta g[h_1'[h_2[a']]]$ with b' the residue of an occurrence of variable b. Let us notice that $a \twoheadrightarrow_\lambda b$, so by induction hypothesis we have a bound on the number of such b'. This gives us a bound on the number of residues of a.

Let us consider non-terminating λ-term $\Omega = (\lambda x.(x)x)\lambda y.(y_1)y_2$. We have $\Omega \to_\beta (\lambda y.(y_1^1)y_2^1)\lambda y.(y_1^2)y_2^2$. So, if we set $g = \circ$, $h_1 = \lambda y.(y_1^1)\circ$ and $h_2 = \lambda y.(y_1^2)\circ$, we can notice that $y_2 \twoheadrightarrow_\lambda y_2$. So Ω is not $\twoheadrightarrow_\lambda$-stratified. The $\twoheadrightarrow_\lambda$-stratified terms will satisfy an elementary bound (tower of exponential of fixed height), to express this bound we introduce the following notation: let $a, b, c \in \mathbb{N}$, then $a_0^b = b$ and $a_{c+1}^b = a^{a_c^b}$.

Theorem 29. *If $\twoheadrightarrow_\lambda$ is acyclic on t, then every $\to_\beta$ sequence beginning by t has length at most*

$$|t| \cdot 2^{|t|}_{2\cdot|\twoheadrightarrow_\lambda|}$$

Proof. Let us consider a reduction $t \to_\beta t_1 \to_{cut} \to_\beta t_n$ and an occurrence of variable x in t. Then, the residues of x in $(t_i)_{1\le i\le n}$ form a tree: every residue of x in t_{i+1} "comes" from a residue of x in t_i. The set of leaves of this forest is written L_x. We prove by induction on i that $\sum_{s_{\twoheadrightarrow_\lambda}(x)\le i} |L_x| \le 2^{|t|}_{2\cdot i}$.

If $s_{\twoheadrightarrow_\lambda}(x) = 1$ then, whenever $t_i = g[(\lambda y.u)h[x']] \to_\beta g[u[h[x']/y]]$ with x' a residue of x, then there is no occurrence of y in u (otherwise we would have $x \twoheadrightarrow_\lambda y$ which contradicts the assumption that $s_{\twoheadrightarrow_\lambda}(x) = 1$). So the residues of x are never duplicated, and $|L_x| = 1$.

If $s_{\twoheadrightarrow_\lambda}(x) = i + 1 > 1$, let us consider a branch $x = x_0 \xrightarrow{y_1} x_1 \xrightarrow{y_2} \cdots \xrightarrow{y_k} x_k$ of the residues tree associated with x. We can notice that x_k is uniquely determined by the sequence $y_1, \cdots, y_k$. For $1 \le j < k$, t reduces to a term of the shape $t_{i_j} = g[(\lambda y.h_1[y_{j+1}])h_2[x_j]] \to_\beta t_{i_j+1}$ with y_{j+1} a residue of an occurrence of variable z_{j+1}. Thus, we have $x \twoheadrightarrow_\lambda z_{j+1}$ so $s_{\twoheadrightarrow_\lambda}(z_{j+1}) \le i$. We can observe that y_{j+1} is a leaf of the residues tree associated with z_{j+1} and y_{j+1} has no residue in t_{i_j+1}. So $y_1, \cdots, y_k$ are distinct leaves of occurrences of variables $z_1, \cdots, z_k$ whose $\twoheadrightarrow_\lambda$-strata are at most i.

By induction hypothesis, there exist at most $2^{|t|}_{2i}$ such leaves. Thus, there are at most $2^{|t|}_{2i}$ possibilities for each y_j and $k \le 2^{|t|}_{2i}$. So, for each x with $s_{\twoheadrightarrow_\lambda}(x) = i + 1$, there are at most $\left(2^{|t|}_{2i}\right)^{2^{|t|}_{2i}}$ leaves of the residues tree associated with x.

$$\sum_{s_{\twoheadrightarrow_\lambda}(x)\leq i+1} |L_x| \leq |t|.\left(2_{2i}^{|t|}\right)^{2_{2i}^{|t|}}$$

$$\sum_{s_{\twoheadrightarrow_\lambda}(x)\leq i+1} |L_x| \leq 2^{\log|t|}.\left(2^{2_{2i-1}^{|t|}}\right)^{2_{2i}^{|t|}}$$

$$\sum_{s_{\twoheadrightarrow_\lambda}(x)\leq i+1} |L_x| \leq 2^{\log|t|+2_{2i-1}^{|t|}\cdot 2_{2i}^{|t|}}$$

$$\log|t| + 2_{2i-1}^{|t|}\cdot 2_{2i}^{|t|} \leq |t| + 2_{2i-1}^{|t|}\cdot 2_{2i}^{|t|} \leq 2_{2i-1}^{|t|} + 2_{2i-1}^{|t|}\cdot 2_{2i}^{|t|}$$

$$\log|t| + 2_{2i-1}^{|t|}\cdot 2_{2i}^{|t|} \leq 2^{2_{2i-1}^{|t|}} = 2_{2i+1}^{|t|}$$

$$\sum_{s_{\twoheadrightarrow_\lambda}(x)\leq i+1} |L_x| \leq 2_{2(i+1)}^{|t|}$$

Let us notice that at each $t_i = g[(\lambda x.u)v] \to_\beta g[u[v/x]] = t_{i+1}$ step, $\sum_{x\in t_i}|L_x| > \sum_{x\in t_{i+1}}|L_x|$. Indeed, there is an injection from $\cup_{x\in t_{i+1}}L_x$ to $\cup_{x\in t_{i+1}}L_x$ (every occurrence of variable of t_{i+1} comes from an occurrence of variable of t_i). And there is at least one occurrence of variable y of t_i which is a leaf (no occurrence of variable of t_{i+1} comes from y): if there is a free occurrence of x in u then x has no residue in t_{i+1}, else the variables of v (and there is at most one by definition of λ-terms) have no residues in t_{i+1}.

So, the length of any $\to_\beta$ sequence is bounded by $2_{2\cdot|\twoheadrightarrow_\lambda|}^{|t|}$. □

3.1.3 Stratification on proof-nets

We would like to define a type-system on λ-calculus such that $\twoheadrightarrow_\lambda$ would be acyclic on every typed term (thus every typed term normalizes in an elementary number of steps). Because Linear Logic pays a special attention to structural rules, we will define our type system as a subsystem of *LL*. In fact, to prove that the type-system enforces an elementary bound, we found it easier to define a relation $\twoheadrightarrow$ on boxes of proof-nets (similar to $\twoheadrightarrow_\lambda$) such that if $\twoheadrightarrow$ is acyclic on the boxes of G, then G normalizes in an elementary number of steps. In Section 3.1.5, we will prove that if the proof-net G "corresponds" to the λ-term t, the acyclicity of $\twoheadrightarrow$ on the boxes of G implies the acyclicity of $\twoheadrightarrow_\lambda$ on the variables of t. The intended meaning of $\twoheadrightarrow$ is that $B \twoheadrightarrow C$ if and only if there exists a *cut*-elimination sequence such that a residue B' of B is inside a residue C' of C.

Definition 30. *Let (B,P), (C,Q) be potential boxes of a proof-net G. We write $(B,P)\dot{\twoheadrightarrow}(C,Q)$ if and only if there exists $e \in C$, $R \in Pot$, $t \in Sig$ and $T \in Tra$ such that:*

$$((\sigma(B),P),[!_t]) \mapsto^* ((e,Q@R),T)$$

Let B,C be potential boxes, we write $B \twoheadrightarrow C$ if there exists $P,Q \in Pot$ such that $(B,P)\dot{\twoheadrightarrow}(C,Q)$.

For example in the proof-net of Figure 3.2, we have $(B_y,[\mathtt{e}])\dot{\twoheadrightarrow}(B_x,[])$ because $((\sigma(B_y),[\mathtt{e}]),[!_\mathtt{e}]) \mapsto^0 ((\sigma(B_y),[\mathtt{e}]),[!_\mathtt{e}])$ and $\sigma(B_y) \in B_x$ (the box B_y is already inside B_x). We have $(B_z,[])\dot{\twoheadrightarrow}(B_x,[])$ because $((\sigma(B_z),[]),[!_{1(\mathtt{n}(\mathtt{e},\mathtt{e}))}]) \mapsto^6 ((\overline{d_x},[\mathtt{e}]),[!_\mathtt{e}])$ (we enter B_x by its auxiliary door). In this case let us notice that, along *cut*-elimination, there is a residue B_z^1 of B_z inside a residue B'_x of B_x. Finally, let us notice

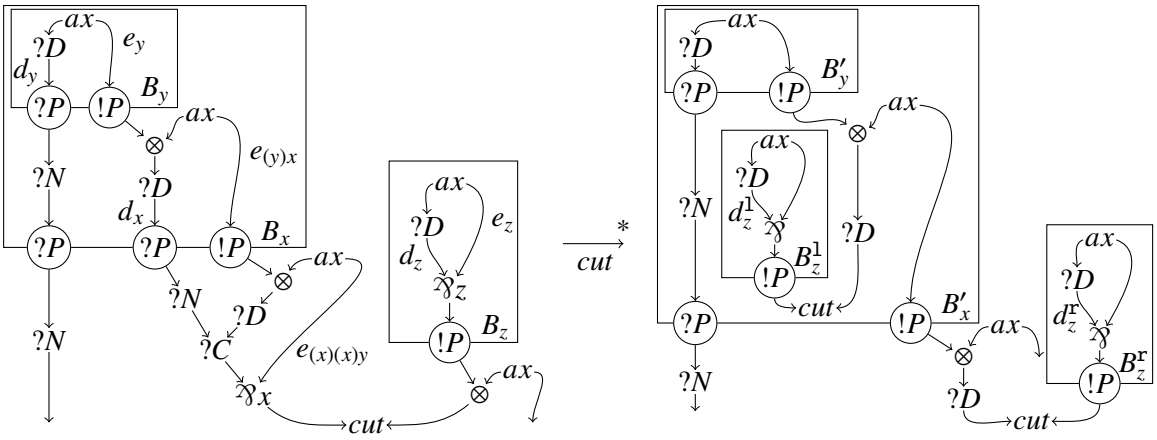


Figure 3.2: Proof-net "corresponding" to $(\lambda x.(x)(x)y)\lambda z.z$

that $(B_x, [])\dot{\twoheadrightarrow}(B_z, [])$ and $(B_y, [\mathtt{e}])\dot{\twoheadrightarrow}(B_z, [])$ because we have $((\sigma(B_x), []), [!_{\mathtt{e}}]) \mapsto^8 ((\overline{d_z}, [\mathtt{r}(\mathtt{e})]), [!_{\mathtt{e}}])$ and $((\sigma(B_y), [\mathtt{e}]), [!_{\mathtt{e}}]) \mapsto^8 ((\overline{d_z}, [\mathtt{l}(\mathtt{n}(\mathtt{e}, \mathtt{e}))]), [!_{\mathtt{e}}])$ (we enter B_z by its principal door). In this case there is never a residue of B_x and B_y inside a residue of B_z, because the $?D$ nodes "open" the boxes. But the principal doors of B_x and B_y are cut with edges $d_z^{\mathtt{r}}$ and $d_z^{\mathtt{l}}$ appearing inside residues $B_z^{\mathtt{r}}$ and $B_z^{\mathtt{l}}$ of B_z.

Because canonical boxes represent residues of boxes, the sets of the shape $Can(B)$ play the same role in the proof of Theorem 32 as the residues trees play in the proof of Theorem 29. We will bound $|Can(B)|$ by induction on $s_{\twoheadrightarrow}(B)$.

Lemma 31. *The number of signatures whose depth is $\leq d$ is at most $2^{2^{2\cdot d}}$*

Proof. For $d \in \mathbb{N}$, we write M_d for the number of signatures whose depth is $\leq d$. We will prove by induction hypothesis that $M_d \leq 2^{2^{2\cdot d}}$. For $d = 1$, we have $M_1 = 1 \leq 2^4$. And for $d > 1$, a signature of depth $\leq d + 1$ is either $\mathtt{e}$, $\mathtt{l}(t)$, $\mathtt{r}(t)$, $\mathtt{p}(t)$ or $\mathtt{n}(t, u)$ with t, u signatures of depth $\leq d$. So,

$$\begin{aligned} M_{d+1} &= 1 + 3M_d + (M_d)^2 && \leq 1 + 3 \cdot 2^{2^{2\cdot n}} + \left(2^{2^{2\cdot n}}\right)^2 \\ M_{d+1} &\leq 4 \cdot 2^{2^{2\cdot n}} + 2^{2^{1+2\cdot n}} && < 2^{2^{1+2\cdot n}} + 2^{2^{1+2\cdot n}} \\ M_{d+1} &\leq 2^{1+2^{1+2\cdot n}} \leq 2^{2^{2\cdot(n+1)}} \end{aligned}$$

□

Theorem 32. *If $\twoheadrightarrow$ is acyclic on B_G and $x = |\vec{\mathrm{E}}_G|$, then*

$$W_G \leq 2^x_{3\cdot|\twoheadrightarrow|}$$

Proof. We first prove by induction on $s_{\twoheadrightarrow}(B)$ that, for every $Q \in Pot$, $|Cop(B, Q)| \leq 2^x_{3\cdot s_{\twoheadrightarrow}(B)-2}$.

Let $(B, P) \in Pot(B_G)$ with $s_{\twoheadrightarrow}(B) = k + 1$, let us consider $t \in Cop(B, P)$ and $u \sqsupseteq t$. By definition of copies, there exists a path of the shape $((\sigma(B), P), [!_u]) \mapsto^* ((f, R), [!_{\mathtt{e}}])$. Let us consider the contexts in the path of the shape $((e, Q), [!_v])$ with $\overline{e}$ the conclusion of a $?N$ node or $?C$ node. Because of the acyclicity of proof-nets (Lemma 21), a given potential edge $(e, [q_1; \cdots; q_{\partial(e)}])$ appears only once among those contexts. How many such potential edges are there? There at most x choices for e. Let us observe that for every every box C containing e, we have $B \twoheadrightarrow C$. So, by induction hypothesis, if $[q_1; \cdots; q_{\partial(C)}]$ is fixed, there are at most at most 2^x_{3k-2} choices for q_i. So there are at most $x \cdot \left(2^x_{3k-2}\right)^{\partial_G}$ choices for $(e, [q_1; \cdots; q_{\partial(e)}])$.

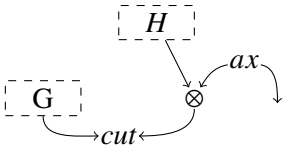


Figure 3.3: This proof-net, written $(G)H$, corresponds to the application of a function G to an argument H.

If we consider u as a tree, those contexts are the only ones where the length of the leftmost branch of u can decrease, and it decreases by at most 1. So, for any $t \in Cop(B,P)$ and $u \sqsupseteq t$ the length of the leftmost branch of u is at most $x.\left(2^x_{3k-2}\right)^{\partial_G}$. Let us notice that there exists a simplification u of t such that the leftmost branch of u corresponds to the deepest branch of t. Thus, for every $t \in Cop(B,P)$ the depth d_t of t is at most $x \cdot \left(2^x_{3k-2}\right)^{\partial_G}$. So we have the following inequalities:

$$2 \cdot d_t \leq 2 \cdot x \cdot \left(2^x_{3k-2}\right)^{\partial_G} \leq 2^x \cdot \left(2^x_{3k-2}\right)^{\partial_G} \leq 2^{x+(2^x_{3k-3})\cdot\partial_G}$$
$$\log(2 \cdot d_t) \leq x + (2^x_{3k-3}) \cdot \partial_G \leq x \cdot 2^x_{3k-3} \leq 2^{2^x_{3k-3}} \leq 2^x_{3k-2}$$

Thus $2 \cdot d_t \leq 2^x_{3k-1}$ and, by Lemma 31, we have $|Cop(B,P)| \leq 2^{2^{2^x_{3k-1}}} = 2^x_{3k+1} = 2^x_{3(k+1)-2}$.

We proved that, for every potential box (B,P), $|Cop(B,P)| \leq 2^x_{3|\twoheadrightarrow|-2}$. So, $|Can(\mathrm{E}_G)| \leq x.\left(2^x_{3|\twoheadrightarrow|-2}\right)^{\partial_G}$, as in the above sequence of inequalities, we can deduce that $W_G = 2 \cdot |Can(\mathrm{E}_G)| \leq 2^x_{3|\twoheadrightarrow|-1} \leq 2^x_{3|\twoheadrightarrow|}$ which is the expected result. □

3.1.4 A *LL*-subsystem characterizing elementary time

Elementary Linear Logic (*ELL*, defined by Danos and Joinet [24]) is the subsystem of Linear Logic where dereliction ($?D$ nodes) and ($?N$ nodes) are forbidden. Every proof-net of *ELL* is $\twoheadrightarrow$-stratified, as will be proved by Lemma 34. The idea is that the only $\mapsto$-steps which increase or decrease the number of signatures in the context are the ones crossing a $?D$ or $?N$ node. Indeed, when a context leaves a box we delete a signature in the potential ($P.t$ becomes P) and create one in the trace (T becomes either $T.!_t$ or $T.?_t$), so the total number of signatures is constant. When a context enters a box, we create a signature in the potential and delete one in the trace. The other steps (except the steps crossing a $?D$ or $?N$ node) neither create nor delete signatures. Because of this property we will deduce that, in a *ELL* proof-net, whenever $B \twoheadrightarrow C$ we have $\partial(B) > \partial(C)$, which entails the acyclicity of $\twoheadrightarrow$.

Definition 33. *Let $[T_1;\cdots;T_k] \in Tra$, we define $|[T_1;\cdots;T_k]|_{!,?}$ as $|\{i \mid \exists t \in Sig, T_i = !_t \text{ or } T_i = ?_t\}|$.*

Lemma 34. *If the proof-net G does not contain any $?D$ or $?N$ node, then G is $\twoheadrightarrow$-stratified*

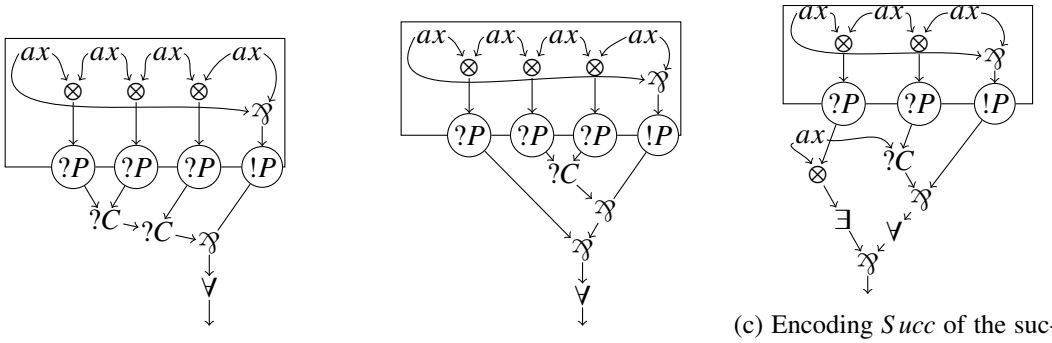
Proof. We can prove by induction on k that, in the absence of $?D$ and $?N$ nodes,

$$((e,P),T) \mapsto^k ((f,Q),U) \Rightarrow \partial(e) + |T|_{!,?} = \partial(f) + |U|_{!,?}$$

If $B \twoheadrightarrow C$, there exists $e \in C$, $P,Q \in Pot$ and $t,u \in Sig$ such that $((\sigma(B),P),[!_t]) \mapsto^* ((e,Q),[!_u]@U)$. Thus,

$$\partial(\sigma(B)) + |[!_t]|_{!,?} = \partial(e) + |[!_u]@U|_{!,?}$$
$$\partial(B) + 1 \geq \partial(e) + 1$$
$$\partial(B) \geq \partial(e) > \partial(C)$$

So $\twoheadrightarrow$ is acyclic and for every $B \in B_G$, $s_{\twoheadrightarrow}(B) \leq \partial(B)$. In particular, we have $|\twoheadrightarrow| \leq \partial_G$. □



(a) Encoding $Enc_N(3)$ of 3

(b) $Enc_B([0;1;1])$

(c) Encoding $Succ$ of the successor function.

Figure 3.4: Encoding of natural numbers and binary lists in *ELL*.

We proved that the length of any $\rightarrow_{cut}$ sequence starting from a *ELL* proof-net G depends on an elementary function of $|\,\mathrm{E}_G\,|$. However, we are not trying to prove a compexity bound on a single proof-net but on a function. So we would like to prove that for every *ELL* proof-net G there exists an elementary function e such that for every *ELL* proof-net H, the proof-net $(G)H$ (corresponding to the application of G to H and represented in Figure 3.3) normalizes in at most $e\,(|\,\mathrm{E}_H\,|)$ steps. However, in the general case, the elementary function depends on $|\twoheadrightarrow|$ which depends on H.

So, in what meaning does *ELL* correspond to elementary time? We encode the set $\mathbb{N}$ of natural numbers (resp. the set $\mathbb{B}$ of binary lists) in *ELL* by the proof-nets whose only pending edge is labelled by the formula $\underline{N}_{ELL}$ (resp. $\underline{B}_{ELL}$) with:

$$\underline{N}_{ELL} = \forall X.!(X \multimap X) \multimap !(X \multimap X)$$
$$\underline{B}_{ELL} = \forall X.!(X \multimap X) \multimap !(X \multimap X) \multimap !(X \multimap X)$$

Figure 3.4 shows the encoding $Enc_N(3)$ of 3 and the encoding $Enc_B([0;1;1])$ of $[0;1;1]$. Figure 3.4c shows the encoding $Succ$ of the successor (the function $n \mapsto n+1$). Then, the set of elementary time functions from $\mathbb{B}$ to $\mathbb{B}$ is captured by the set of proof-nets G of *ELL* whose only pending edge is labelled by a formula of the shape $\underline{B}_{ELL} \multimap !!\cdots!\underline{B}_{ELL}$. To prove the soundness, let us consider such a proof-net G. Then, for every $l \in \mathbb{B}$, $\left|\vec{\mathrm{E}}_{(G)Enc_B(l)}\right| \leq \left|\vec{\mathrm{E}}_G\right| + 4\cdot|l| + 10$ and $|\twoheadrightarrow| \leq |B_{(G)Enc_B(l)}| \leq |B_G| + 1$. So, by Theorem 32,

$$W_{(G)Enc_B(l)} \leq 2^{4\cdot|l|+10+2|\,\mathrm{E}_G\,|}_{3|B_G|+1}$$

So there exists an elementary function e_G such that for every binary list l, $(G)Enc_B(l)$ normalizes in at most $e_G(|l|)$ steps. The completeness of this characterization (the fact that any elementary-time function from binary lists to binary lists can be encoded in such a way) is proved by Danos and Joinet [24]. This is in this way that *ELL* characterizes elementary time. Let us formalize the notion of Linear Logic subsystems, soundness and completeness.

Definition 35. *A subsystem S of Linear Logic is a tuple $(\mathcal{F}_S, \Pi_S, G_S, \underline{B}_S, Enc\_(\_))$ with:*

- *$\mathcal{F}_S$ a set and Π_S is a mapping from $\mathcal{F}_S$ to $\mathcal{F}_{\mathrm{LL}}$. So $\mathcal{F}_S$ can be understood as a refinement of the LL formulae.*
- *G_S is a set of LL proof-nets whose edges are labelled by formulae of $\mathcal{F}_S$ compatible with $\beta_G(\_)$: if $\beta_G(e) = A$ and e is labelled by $B \in \mathcal{F}_S$ then $\Pi_S(B) = A$.*

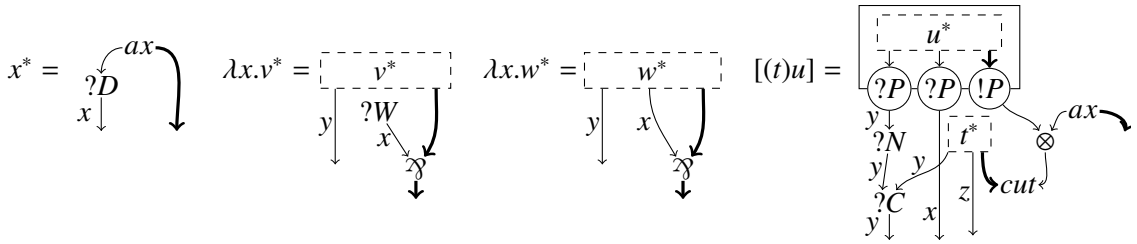


Figure 3.5: Encoding of λ-calculus in untyped proof-nets.

- $\underline{B_S}$ *is a subset of* $\mathcal{F}_S$ *representing the formulae of the encoding of binary lists. The encoding is represented by* $Enc_{\_}(\_)$*: for every* $A \in \underline{B_S}$*,* $Enc_A(\_)$ *is a mapping from binary lists to distinct sets proof-nets in* G_S *whose conclusion is labelled by* A *(if* $l \neq m$ *then* $Enc_A(l) \cap Enc_A(m) = \varnothing$*). Finally, we require that the size of the encoding of a binary list depends linearly on the size of the list. Formally, for every* $A \in \underline{B_S}$*, there exist* $a, b \in \mathbb{N}$ *such that for every binary list* l *and* $G \in Enc_A(l)$*,* $|\mathrm{E}_G| \leq a \cdot |l| + b$*.*

Definition 36. *Let us fix a subsystem* S *of LL, and a set* C *of mappings from* $\mathbb{N}$ *to* $\mathbb{N}$*.*

We say that S is sound for C *if, for every proof-net* G *whose only conclusion's label is of the shape* $A \multimap B$ *with* $A \in \underline{B_S}$*, there exists* $c_G \in C$ *such that, for every binary list* l *and* $H \in Enc_A(l)$ *such that* $(G)H \in G_S$*, we have* $W_{(G)H} \leq c_G(|l|)$*.*

We say that S is complete for C *if for every function* f *from* $\mathbb{B}$ *to* $\mathbb{B}$ *whose time complexity is in* C*, and* $A \in \underline{B_S}$ *there exists a proof-net* G_f *whose only conclusion is labelled by* $A \multimap B$ *with* $B \in \underline{B_S}$ *such that for every binary list* l *and* $H \in Enc_A(l)$*,* $(G_f)H$ *reduces to* $H' \in Enc_B(f(l))$*.*

Finally, we say that S characterizes C *if* S *is sound and complete for* C*.*

In the case of ELL, we can set $\mathcal{F}_{ELL} = \mathcal{F}_{\mathrm{LL}}$, Π_S is the identity on $\mathcal{F}_{ELL}$, G_S is the set of proof-nets which does not contain $?D$ and $?N$ nodes, $\underline{B_S}$ is the set of formulae of the shape $!^i\underline{B_{ELL}}$ (the formula $\underline{B_{ELL}}$ preceded by i modalities !) and, for $i \in \mathbb{N}$, $Enc_{!^i\underline{B_{ELL}}}(l)$ is the singleton whose element is the proof-net obtained by putting $Enc_B(l)$ inside i boxes. Finally we set $Elem$ as the set $\left\{x \mapsto 2^x_k \,\middle|\, k \in \mathbb{N}\right\}$. By Lemma 34, and because every proof-net of the shape $Enc_{!^i\underline{B_{ELL}}}(l)$ has $i+1$ boxes, ELL is sound for $Elem$. Danos and Joinet proved the $Elem$ completeness of ELL with Theorem 8 of [24]. The fact that the number of boxes in proof-nets of the shape $Enc_{!^i\underline{B_{ELL}}}(l)$ is bounded will be a key feature of all of our subsystems. We define this property as *box-boundedness*:

Definition 37. *A LL subsystem is said* box-bounded *if for every* $A \in \underline{B_S}$*, there exists* $n_A \in \mathbb{N}$ *such that, for every binary list* l *and* $G \in Enc_A(l)$*, there are at most* n_A *boxes in* G*.*

Most of the characterizations S of complexity classes C we will consider are generalizations of previous characterization S' of C, i.e. S' is included in S. Thus, the completeness of S can be immediately deduced from the completeness of S'. This is why, we will focus on soundness.

3.1.5 Correspondence between λ-calculus stratification and proof-net stratification

The idea of this subsection is to state that for every λ-term t, if the proof-net corresponding to t is $\twoheadrightarrow$-stratified, then t is $\twoheadrightarrow_\lambda$-stratified. Even though λ-terms and proof-nets are related, there is no one-to-one correspondence. For any λ-term t, there are many different proof-nets corresponding to t. The correspondence will hold for a specific encoding of λ-terms in proof-nets, named Girard's encoding [36].

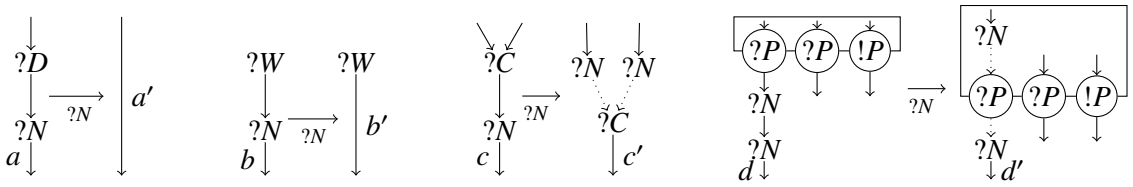


Figure 3.6: The $\to_{?N}$ relation fixes the slight mismatch between λ-terms and their encodings.

A λ-term t whose free variables are $x_1, \cdots, x_n$ will be encoded by a proof-net t^* with $n+1$ pending edges. One of the pending edge is distinguished (in Figure 3.5 it is thicker), the other ones are labelled by the variables $x_1, \cdots, x_n$. The encoding of λ-terms is defined in Figure 3.5.

In the $(\lambda x.t)^*$ case, the free variables of t are the free variables of $\lambda x.t$ (represented by the variable y) and, possibly, x. To represent both cases, we suppose that x is a free variable of w but not a free variable of v. In the definitions of $(t)u^*$, we use variables x, y and z to represent the different cases (free variables of t, free variables of both t and u, and free variables of u).

The $(\_)^*$ mapping defines a proof-net corresponding to each λ-term. If u is a subterm of t, then the definition of t^* uses u^* as a subproof-net. We define the *edge corresponding to u* (written e_u) as the thick edge created in the definition of u^*. For any occurrence of variable x, d_x refers to the conclusion of the $?D$ node appearing in the definition of x^*. And, if there exists a box containing e_x, we define B_x as the deepest such box. We can observe that the proof-net of Figure 3.2 is $((\lambda x.(x)(x)y)\lambda z.z)^*$ (up to some *ax cut*-elimination steps, to simplify the proof-net). The edge d_x, and the box B_x correspond to the second occurrence of x.

The encoding presented in [36] is done with an equivalent presentation of linear logic where, whenever the label of the conclusion of an auxiliary door is $?A$, the label of its premise is $?A$ (in our presentation, called "functorial promotion", the label of its premise is A). We chose functorial promotion because it simplifies the definition of context semantics. However, with our presentation, there is a slight mismatch between $\to_\beta$ and $\to_{cut}$. To fix this mismatch, we define a relation $\to_{?N}$ on proof-nets as described in Figure 3.6.

Lemma 38. *If $G \to_{?N} H$ then there exists a mapping $\rho_{G\to H}(\_)$ from the contexts of H whose edge is not modified by the $\to_{?N}$ step to the contexts of G such that*

$$(C \mapsto^* D) \Leftrightarrow (\rho_{G\to H}(C) \mapsto^* \rho_{G\to H}(D))$$

Proof. The dotted edges of Figure 3.6 are the edges e for which $\rho_{G\to H}(((e,P),T))$ is undefined. For each $\to_{?N}$ step, there are only a few interesting cases which we present below. In each case, when we define $\rho_{G\to H}(((e',P),T@[?_t])) = ((e,P),T@[?_u])$, then we define $\rho_{G\to H}(((\overline{e'},P),T@[!_t])) = \rho_{G\to H}(((\overline{e},P),T@[!_u]))$. Then, if $((e',P),T@[!_t]) \mapsto^* C$, we define $\rho_{G\to H}(C)$ as the context obtained from C by replacing t by u. And, if $C \mapsto^* ((\overline{e'},P),T@[!_t])$, we define $\rho_{G\to H}(C)$ as the context obtained from C by replacing t by u[1].

In the first case, we define $\rho_{G\to H}(((a',P),T@[?_t])) = ((a,P),T@[?_{\mathtt{n}(t,\mathtt{e})}])$. Similarly, in the second case we define $\rho_{G\to H}(((b',P),T@[?_t])) = ((b,P),T@[?_{\mathtt{n}(t,\mathtt{e})}])$.

In the third case, in order to have $\rho_{G\to H}(C) \mapsto^* \rho_{G\to H}(D) \Rightarrow C \mapsto^* D$ we define:

$$\rho_{G\to H}(((c',P),T@[?_{\mathtt{l}(\mathtt{n}(t,u))}])) = ((c,P),T@[?_{\mathtt{n}(t,\mathtt{l}(u))}])$$
$$\rho_{G\to H}(((c',P),T@[?_{\mathtt{l}(\mathtt{r}(t,u))}])) = ((c,P),T@[?_{\mathtt{n}(t,\mathtt{r}(u))}])$$
$$\rho_{G\to H}(((c',P),T@[?_{\mathtt{l}(\mathtt{p}(u))}])) = ((c,P),T@[?_{\mathtt{p}(\mathtt{l}(u))}])$$
$$\rho_{G\to H}(((c',P),T@[?_{\mathtt{r}(\mathtt{p}(u))}])) = ((c,P),T@[?_{\mathtt{p}(\mathtt{r}(u))}])$$

[1]The acyclicity of $\nrightarrow$, Lemma 100 in page 90 ensures that $\rho_{G\to H}(\_)$ is well-defined.

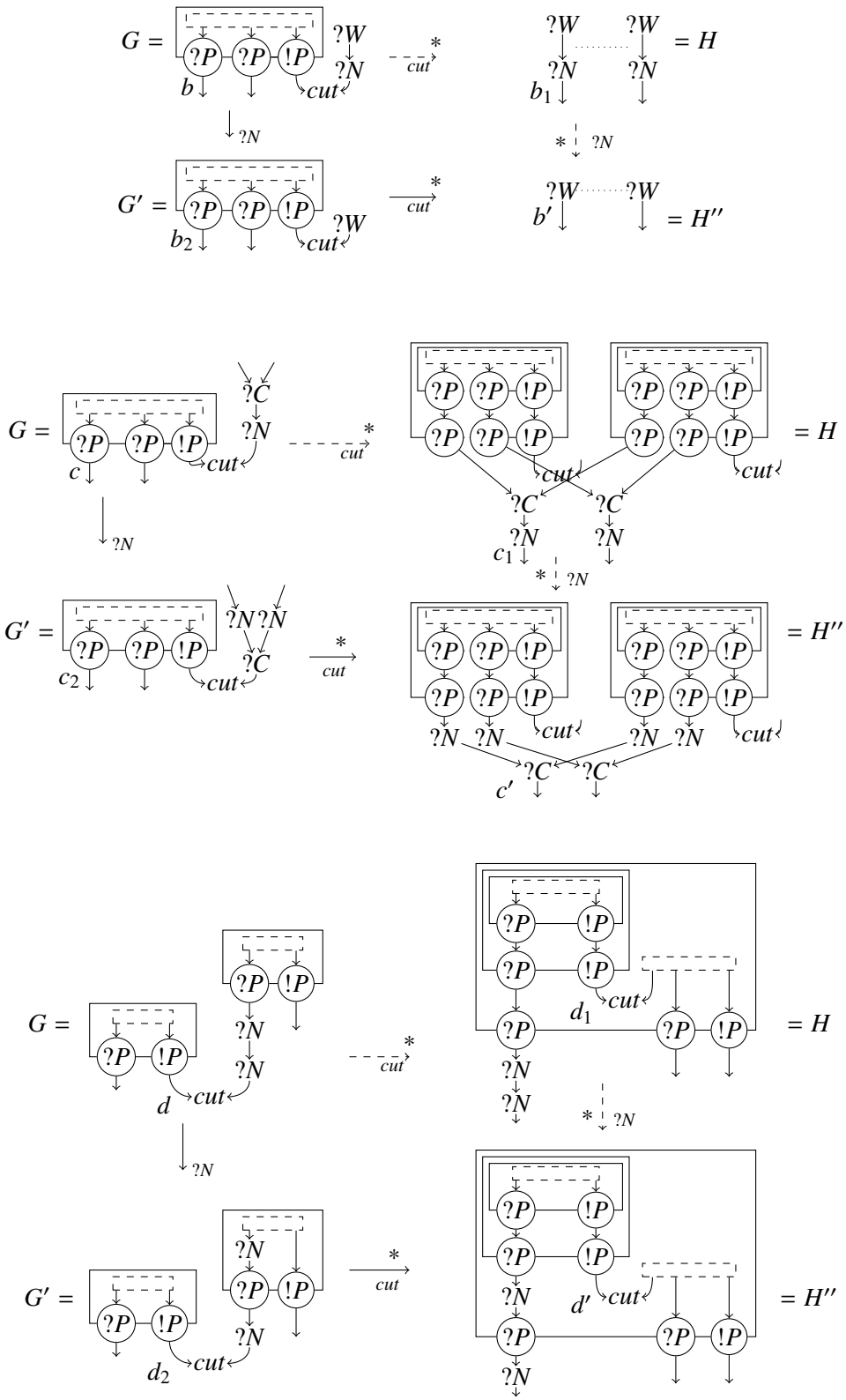


Figure 3.7: Critical pairs for the commutation of $\rightarrow_{?N}$ and $\rightarrow_{cut}$.

Similarly, in order to have $C \mapsto^* D \Rightarrow \rho_{G\to H}(C) \mapsto^* \rho_{G\to H}(D)$, we define:

$$\begin{aligned}
\rho_{G\to H}(((c', P), T@[?_{\mathtt{n}(t,\mathtt{l}(u))}])) &= ((c, P), T@[?_{\mathtt{l}(\mathtt{n}(t,u))}])\\
\rho_{G\to H}(((c', P), T@[?_{\mathtt{n}(t,\mathtt{r}(u))}])) &= ((c, P), T@[?_{\mathtt{r}(\mathtt{n}(t,u))}])\\
\rho_{G\to H}(((c', P), T@[?_{\mathtt{p}(\mathtt{l}(u))}])) &= ((c, P), T@[?_{\mathtt{l}(\mathtt{p}(u))}])\\
\rho_{G\to H}(((c', P), T@[?_{\mathtt{p}(\mathtt{r}(u))}])) &= ((c, P), T@[?_{\mathtt{r}(\mathtt{p}(u))}])
\end{aligned}$$

In the fourth case, we define:

$$\begin{aligned}
\rho_{G\to H}(((d', P), T@[?_{\mathtt{n}(\mathtt{n}(t,u),v)}])) &= ((d, P), T@[?_{\mathtt{n}(t,\mathtt{n}(u,v))}])\\
\rho_{G\to H}(((d', P), T@[?_{\mathtt{n}(\mathtt{p}(u),v)}])) &= ((d, P), T@[?_{\mathtt{p}(\mathtt{n}(u,v))}])\\
\rho_{G\to H}(((d', P), T@[?_{\mathtt{p}(v)}])) &= ((d, P), T@[?_{\mathtt{p}(\mathtt{p}(v))}])
\end{aligned}$$

□

Lemma 39. *If $G \to_{?N} G' \to^*_{cut} H'$ then there exist proof-nets H and H'' such that $H' \to^*_{cut} H''$, $G \to^*_{cut} H \to^*_{?N} H''$. Moreover, we have $\pi_{G\to H} \circ \rho_{H\to H''} = \rho_{G\to G'} \circ \pi_{G'\to H''}$.*

Proof. The critical pairs are presented in Figure 3.7. To prove the commutation of π and ρ, we will only present the most interesting cases.

$$\begin{aligned}
\pi_{G\to H} \circ \rho_{H\to H''}((\overline{a_2}, P), T.!_t) = \pi_{G\to H}((\overline{a_1}, P), T.!_{\mathtt{n}(t,\mathtt{e})}) &= ((\overline{a}, P), T.!_{\mathtt{n}(t,\mathtt{e})})\\
\rho_{G\to G'} \circ \pi_{G'\to H''}((\overline{a_2}, P), T.!_t) = \rho_{G\to G'}((\overline{a_2}, P), T.!_t) &= ((\overline{a}, P), T.!_{\mathtt{n}(t,\mathtt{e})})
\end{aligned}$$

$$\begin{aligned}
\pi_{G\to H} \circ \rho_{H\to H''}((\overline{b'}, P), T.!_t) = \pi_{G\to H}((\overline{b_1}, P), T.!_{\mathtt{n}(t,\mathtt{e})}) &= ((\overline{b}, P), T.!_{\mathtt{n}(t,\mathtt{e})})\\
\rho_{G\to G'} \circ \pi_{G'\to H''}((\overline{b'}, P), T.!_t) = \rho_{G\to G'}((\overline{b_2}, P), T.!_t) &= ((\overline{b}, P), T.!_{\mathtt{n}(t,\mathtt{e})})
\end{aligned}$$

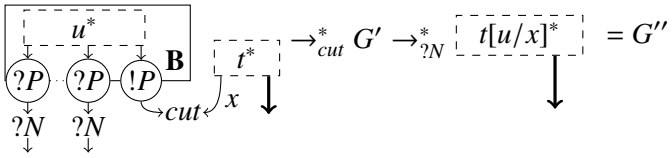
$$\begin{aligned}
\pi_{G\to H} \circ \rho_{H\to H''}((\overline{c'}, P), T.!_{\mathtt{l}(\mathtt{n}(t,u))}) = \pi_{G\to H}((\overline{c_1}, P), T.!_{\mathtt{n}(t,\mathtt{l}(u))}) &= ((\overline{c}, P), T.!_{\mathtt{n}(t,\mathtt{l}(u))})\\
\rho_{G\to G'} \circ \pi_{G'\to H''}((\overline{c'}, P), T.!_{\mathtt{l}(\mathtt{n}(t,u))}) = \rho_{G\to G'}((\overline{c_2}, P), T.!_{\mathtt{l}(\mathtt{n}(t,u))}) &= ((\overline{c}, P), T.!_{\mathtt{n}(t,\mathtt{l}(u))})
\end{aligned}$$

$$\begin{aligned}
\pi_{G\to H} \circ \rho_{H\to H''}((d', P.v), T.!_t.!_u) = \pi_{G\to H}((d_1, P.v), T.!_t; !_u) &= ((d, P), T.!_{\mathtt{n}(t,\mathtt{n}(u,v))})\\
\rho_{G\to G'} \circ \pi_{G'\to H''}((d', P.v), T.!_t.!_u) = \rho_{G\to G'}((d_2, P), T.!_{\mathtt{n}(\mathtt{n}(t,u),v)}) &= ((d, P), T.!_{\mathtt{n}(t,\mathtt{n}(u,v))})
\end{aligned}$$

□

Lemma 40 proves that the encoding is compatible with reduction. In particular: during a reduction step, the residues of the edge corresponding to a subterm t' are the edges corresponding to the residues of t'.

Lemma 40. *Let t be a λ-term and x be a free variable of t,*

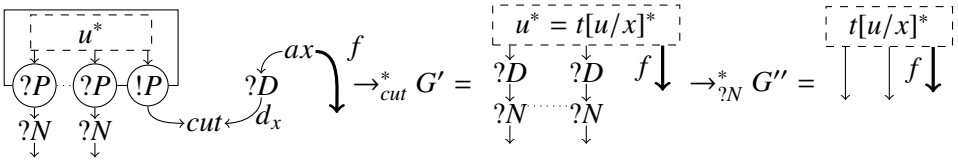
$G =$ 

- *Let us consider a subterm t' of t, for every context of $t[u/x]^*$ of the shape $((e_{t'[u/x]}, P), T)$, we have $\pi_{G\to G'} \circ \rho_{G'\to G''}((e_{t'[u/x]}, P), T) = ((e_{t'}, P), T)$. Let us recall that $\pi_{G\to G'}(\_)$ is defined in Definition 12, page 26.*

- *If u' is a strict subterm of u and $((e_{u'}, P), T) \in Cont_{G''}$, then $\pi_{G\to G'} \circ \rho_{G'\to G''}((e_{u'}, P), T) = ((e_{u'}, Q), T)$ with $Q \in Pot$.*

- *If x is a free occurrence of variable in t and $(d_x, [P_1; \cdots; P_{\partial(d_x)}]) \in Pot(d_x)$, there exists a standard signature v such that $((\sigma(B), []), [!_v]) \rightsquigarrow^* ((\overline{d_x}, [P_1; \cdots; P_{\partial(d_x)}]), [!_{\mathtt{e}}])$ and for every box C of t^* containing e_x, there exists $w \sqsupset v$ such that $((\sigma(B), []), [!_w]) \mapsto^* ((\sigma(C), [P_1; \cdots; P_{\partial(C)-1}]), [!_{P_{\partial(C)}}])$.*

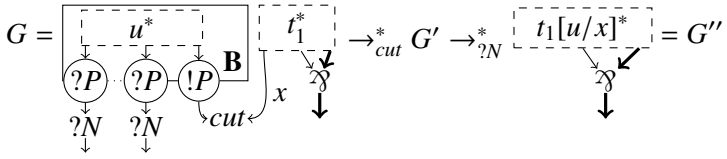
Proof. We prove it by induction on t.

- If $t = x$, then $G =$ 

$\to^*_{cut} G' \to^*_{?N} G''$

  - The only subterm of x is x, if $((f, []), T)$ is a context of G'', then $((f, []), T)$ is a context of G.
  - If u' is a strict subterm of u, and $((e_{u'}, P), T)$ is a context of G'', then by definition of π and ρ, we have $\pi_{G\to G'} \circ \rho_{G'\to G''}((e_{u'}, P), T) = ((e_{u'}, [\mathtt{e}]@P'), T')$.
  - $((\sigma(B), []), [!_{\mathtt{e}}]) \rightsquigarrow^* ((\overline{d_x}, []), [!_{\mathtt{e}}])$ and there is no box containing e_x.

- If $t = \lambda y.t_1$, we suppose that y appears free in t_1 (the other case is quite similar)

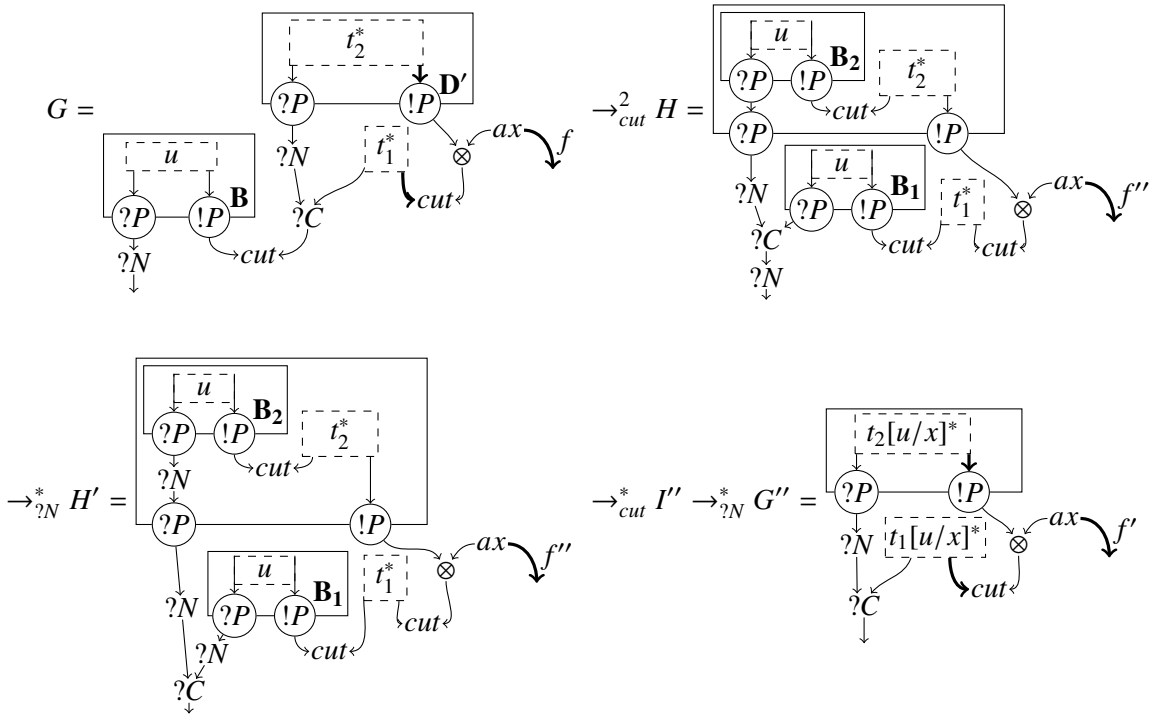


$\to^*_{cut} G' \to^*_{?N} = G''$

  which is the expected result because $(\lambda y.t_1)[u/x] = \lambda y.(t_1[u/x])$. We write G_1 the proof-net obtained by cutting the box B with the edge labelled x of t_1^*. And we write G_1' and G_1'' the proof-nets obtained by induction hypothesis.

  - If t' is a subterm of t, either $t' = t$ (and in this case we have $\pi_{G\to G'} \circ \rho_{G'\to G''}((e_{t[u/x]}, P), T) = \pi_{G\to G'}((e, P), T) = ((e, P), T) = ((e_t, P), T))$ or t' is a subterm of t_1 (and in this case we have $\pi_{G_1\to G_1'} \circ \rho_{G_1'\to G_1''}((e_{t'[u/x]}, P), T) = ((e_{t'}, P), T)$ by induction hypothesis so, by definition of π and ρ, we have $\pi_{G\to G'} \circ \rho_{G'\to G''}((e_{t'[u/x]}, P), T) = ((e_{t'}, P), T)$).
  - If u' is a strict subterm of u, for every context $((e_{u'}, P), T)$, we know by induction hypothesis that there exists $Q \in Pot$ such that $\pi_{G_1\to G_1'} \circ \rho_{G_1'\to G_1''}((e_{u'}, P), T) = ((e_{u'}, Q), T)$. So $\pi_{G\to G'} \circ \rho_{G'\to G''}((e_{u'}, P), T) = ((e_{u'}, Q), T)$.
  - If x is a free occurrence in t and $(d_x, [P_1; \cdots; P_{\partial(d_x)}])$ is a potential edge, by induction hypothesis, there exists a standard signature v such that $((\sigma(B), []), [!_v]) \rightsquigarrow^* ((\overline{d_x}, [P_1; \cdots; P_{\partial(d_x)}]), [!_{\mathtt{e}}]))$. Let C be a box containing e_x in t^*, then C also contains e_x in t_1^* so there exists $w \in Sig$ such that $((\sigma(B), []), [!_w]) \mapsto^* ((\sigma(C), [P_1; \cdots; P_{\partial(C)}]), [!_{\partial(C)+1}])$.

- If $t = (t_1)t_2$, we will suppose that x is a free variable of both t_1 and t_2 (otherwise it is simpler as we do not have to deal with the $?C$ node).



Which is the expected result as $((t_1)t_2)[u/x] = (t_1|u/x])t_2[u/x]$ and (by Lemma 39) $\to_{?N}$ and $\mapsto$ commute: there exists a proof-net G' such that $H \to^*_{cut} G' \to^*_{?N} I''$ and $\pi_{H\to G'} \circ \rho_{G'\to I''} = \rho_{H\to H'} \circ \pi_{H'\to I''}$. So $G \to^*_{cut} G' \to^*_{?N} G''$ and $\pi_{G\to G'} \circ \rho_{G'\to G''} = \pi_{G\to H} \circ \rho_{H\to H'} \circ \pi_{H'\to I''} \circ \rho_{I''\to G''}$. We write G_1 (resp G_2) the proof-net obtained by cutting the box B with the edge labelled x of t^*_1 (resp. t^*_2). And we write G'_1, G''_1, G'_2 and G''_2 the proof-nets obtained by induction hypothesis.

- If t' is a subterm of t and $((e_{t'[u/x]}, P), T) \in Cont_{G'}$, then:
  - Either we have $t = t'$. In this case $\pi_{G\to G'} \circ \rho_{G'\to G''}((e_{t'[u/x]}, []), T) = \pi_{G\to H} \circ \rho_{H\to H'} \circ \pi_{H'\to I''} \circ \rho_{I''\to G''}((f', []), T) = ((f, []), T)$ which is the expected result because $e_{t'} = f$.
  - Or $t_1 = h'[t']$. Then we know by induction hypothesis that $\pi_{G_1\to G'_1} \circ \rho_{G'_1\to G''_1}((e_{t'[u/x]}, P), T) = ((e_{t'}, P), T)$ so $\pi_{H'\to I''} \circ \rho_{I''\to G''}((e_{t'[u/x]}, P), T) = ((e_{t'}, P), T)$ and, finally, we have $\pi_{G\to G'} \circ \rho_{G'\to G''}((e_{t'[u/x]}, P), T) = ((e_{t'}, P), T)$.
  - Or $t_2 = h'[t']$ and $P = Q.q$. By induction hypothesis, $\pi_{G_2\to G'_2} \circ \rho_{G'_2\to G''_2}((e_{t'[u/x]}, Q), T) = ((e_{t'}, Q), T)$ so $\pi_{H'\to I''} \circ \rho_{I''\to G''}((e_{t'[u/x]}, P), T) = ((e_{t'}, P), T)$ and, finally, we have $\pi_{G\to G'} \circ \rho_{G'\to G''}((e_{t'[u/x]}, P), T) = ((e_{t'}, P), T)$.
- Let us suppose $t = g[x]$, u' is a strict subterm of u and $((e_{u'}, P), T) \in Cont_{G'}$. Either $t_1 = g'[x]$ or $t_2 = g'[x]$. If we suppose that $t_1 = g'[x]$, then by induction hypothesis there exists a potential $[q]@Q$ such that $\pi_{G_1\to G'_1} \circ \rho_{G'_1\to G''_1}((e_{u'}, P), T) = ((e_{u'}, [q]@Q), T)$ so $\pi_{H'\to I''} \circ \rho_{I''\to G''}((e_{u'}, P), T) = ((e_{u'}, [q]@Q), T)$, thus $\pi_{G\to G'} \circ \rho_{G'\to G''}((e_{u'}, P), T) = ((e_{u'}, [\mathtt{r}(q)]@Q), T)$. If $t_2 = g'[x]$, then $P = q_2@P'$ and by induction hypothesis there exists a potential $[q_1]@Q$ such that $\pi_{G_2\to G'_2} \circ \rho_{G'_2\to G''_2}((e_{u'}, P'), T) = ((e_{u'}, [q_1]@Q), T)$ so we can deduce that $\pi_{H'\to I''} \circ \rho_{I''\to G''}((e_{u'}, P), T) = ((e_{u'}, [q_2; q_1]@Q), T)$ and finally we have $\pi_{G\to G'} \circ \rho_{G'\to G''}((e_{u'}, P), T) = ((e_{u'}, [\mathtt{l}(\mathtt{n}(q_1, q_2))]@Q), T)$.

– If $t = g[x]$ and $(d_x, [P_1; \cdots ; P_{\partial(d_x)}]) \in Pot(\vec{\mathrm{E}}_G)$, then either $t_1 = g'[x]$ or $t_2 = g'[x]$. If $t_1 = g'[x]$, then by induction hypothesis there exists $v \in Sig$ such that $((\sigma(B), []), [!_v]) \rightsquigarrow^* ((\overline{d_x}, P), [!_{\mathtt{e}}])$ is a path of G_1 so $((\sigma(B_1), []), [!_v]) \rightsquigarrow^* ((\overline{d_x}, P), [!_{\mathtt{e}}])$ is a path of H' and $((\sigma(B), []), [!_{\mathtt{r}(v)}]) \rightsquigarrow^* ((\overline{d_x}, P), [!_{\mathtt{e}}])$ is a path of G. If C is a box of t^* containing e_x then C is a box of t_1^* containing e_x. By induction hypothesis, there exists $w \in Sig$ such that $v \sqsubseteq w$ and $((\sigma(B), []), [!_w]) \mapsto^* ((\sigma(C), [P_1; \cdots ; P_{\partial(C)}]), [!_{P_{\partial(C)+1}}])$ is a path of G_1 so we can deduce that $((\sigma(B), []), [!_{\mathtt{r}(w)}]) \mapsto^* ((\sigma(C), [P_1; \cdots ; P_{\partial(C)}]), [!_{P_{\partial(C)+1}}])$ is a path of G and $\mathtt{r}(v) \sqsubseteq \mathtt{r}(w)$.

  If $t_2 = g'[x]$ then $P = [v_2]@P'$ and, by induction hypothesis, there exists $v_1 \in Sig$ such that $((\sigma(B), []), [!_{v_1}]) \rightsquigarrow^* ((\overline{d_x}, P'), [!_{\mathtt{e}}])$ is a path of G_2 so we can deduce that $((\sigma(B_2), [v_2]), [!_{v_1}]) \rightsquigarrow^* ((\overline{d_x}, [v_2]@P'), [!_{\mathtt{e}}])$ is a path of H'. Finally, $((\sigma(B), []), [!_{\mathtt{l}(\mathtt{n}(v_1, v_2))}]) \rightsquigarrow^* ((\overline{d_x}, P), [!_{\mathtt{e}}])$ is a path of G. If C is a box of t^* containing e_x, either $C = D'$ or C is a box of t_2^* containing e_x. In the first case, let us notice that $((\sigma(B), []), [!_{\mathtt{p}(v_2)}]) \mapsto^* ((\sigma(C), []), [!_{v_2}])$. In the second case, by induction hypothesis there exists $w_1 \in Sig$ such that $v_1 \sqsubseteq w_1$ and $((\sigma(B), []), [!_{w_1}]) \mapsto^* ((\sigma(C), [P_1; \cdots ; P_{\partial(C)}]), [!_{P_{\partial(C)+1}}])$ is a path of G_2. So we can deduce that $((\sigma(B_2), [v_2]), [!_{w_1}]) \mapsto^* ((\sigma(C), [v_2; P_1; \cdots ; P_{\partial(C)}]), [!_{P_{\partial(C)+1}}])$ is a path of H' and finally $((\sigma(B), []), [!_{\mathtt{l}(\mathtt{n}(w_1, v_2))}]) \mapsto^* ((\sigma(C), [v_2; P_1; \cdots ; P_{\partial(C)}]), [!_{P_{\partial(C)+1}}])$ is a path of G (Lemma 13) and $\mathtt{l}(\mathtt{n}(w_1, v_2)) \sqsubseteq \mathtt{l}(\mathtt{n}(v_1, v_2))$.

□

Corollary 41. *If $v \to_\beta^* v'$ then there exists a proof-net H such that $v^* \to_{cut}^+ H \to_{?N}^* v'^*$ and, for every occurrence of variable y' in v' (whose lift in v is written y) and $(d_{y'}, P') \in Pot(d_{y'})$, $\pi_{v^* \to H} \circ \rho_{H \to v'^*}((d_{y'}, P'), [!_{\mathtt{e}}])$ is of the shape $((d_y, P), [!_{\mathtt{e}}])$ and, for every context $((\sigma(B_{y'}), P'), [!_{w'}])$ there exists a context $((\sigma(B_y), P), [!_w])$ such that $((\sigma(B_y), P), [!_w]) \mapsto^* \pi_{v^* \to H} \circ \rho_{H \to v'^*}((\sigma(B_{y'}), P'), [!_{w'}])$.*

Proof. It is enough to prove it in the case of one $\to_\beta$ step, then the general case is obtained by induction using Lemma 38 and Lemma 39. So, let us suppose that $v \to_\beta v'$. By definition of $\to_\beta$, v and v' are of the shape $g[(\lambda x.t)u]$ and $g[t[u/x]]$.

Let us suppose that x appears free in t. Then v^* reduces to the proof-net obtained from $v^* = g[(\lambda x.t)u]^*$ by replacing $((\lambda x.t)u)^*$ by the proof-net G of Lemma 40. Thus, v^* reduces (by $\to_{cut}$) to the proof-net H obtained from v^* by replacing $((\lambda x.t)u)^*$ by the proof-net G' of Lemma 40. And H reduces (by $\to_{?N}$) to the proof-net obtained from v^* by replacing $((\lambda x.t)u)^*$ by $t[u/x]^*$, which is exactly $g[t[u/x]] = t'$.

- If y' is in u, then the results are obtained by the first point of Lemma 40.
- If y' is in u, then $\pi_{v^* \to G} \circ \rho_{G \to v'^*}((d_{y'}, P'), [!_{\mathtt{e}}])$ is of the shape $((d_y, P), [!_{\mathtt{e}}])$ by the second point of Lemma 40. Either B_y is the box B of Lemma 40 (in this case $((\sigma(B_y), P), [!_w])$ is obtained by the third point of Lemma 40) or B_y is a box inside B and $((\sigma(B_y), P), [!_w])$ is obtained by the second point of Lemma 40.
- Else, y' is in g and is not concerned by the reduction so the result is straightforward.

If x does not appear free in t, then y' can not be in u so the proof is even simpler. □

Lemma 42. *Let t be a $\to_\beta$-normalizing λ-term, and x, y variables of t. If B_x, B_y are well-defined, then:*

$$(x \twoheadrightarrow_\lambda y) \Rightarrow (B_x \twoheadrightarrow^+ B_y)$$

Proof. Let us suppose that $x \twoheadrightarrow_\lambda y$, then $t \to_\beta^* g[(\lambda y.h_1[y'])h_2[x']] = t'$. We will prove by induction on the length of the length of this reduction sequence, that there exists a box B, $P, Q \in Pot$ and $v \in Sig$ such that $B_x \subseteq B$ and $((\sigma(B), P), [!_v]) \mapsto^* ((\overline{d_y}, Q), [!_{\mathsf{e}}])$. Once we prove this property, by definition of $\twoheadrightarrow$, either $B_x \twoheadrightarrow B_y$ or $B_x \twoheadrightarrow B \twoheadrightarrow B_y$.

Let us suppose that $t \to_\beta^0 t'$, then let B be the box created in the definition of $((\lambda y.h_1[y])h_2[x])^*$. The term $(\lambda y.h_1[y])h_2[x]$ contains x so $B_x \subseteq B$. By Lemma 40, for every $[P_1; \cdots ; P_{\partial(d_y)}] \in Pot$, there exists $v \in Sig$ such that $((\sigma(B), [P_1; \cdots ; P_{\partial(B_x)}]), [!_v]) \rightsquigarrow^* ((\overline{d_y}, [P_1; \cdots ; P_{\partial(d_y)}]), [!_{\mathsf{e}}])$.

Let us suppose that $t = h_1[(\lambda z.t_1)t_2] \to_\beta t'' = h_1[t_1[t_2/z]] \to_\beta^k t'$, and x and y have residues x'' and y'' in t'' (so x and y are not occurrences of z) such that $x'' \twoheadrightarrow_\lambda y''$. By induction hypothesis, there exists a box B'', potentials P'', Q'' and signature v'' such that $B_{x''} \subseteq B''$ and $((\sigma(B''), P''), [!_{v''}]) \mapsto^* ((\overline{d_{y''}}, Q''), [!_{\mathsf{e}}])$. By Corollary 41, there exists a proof-net G such that $t^* \to_{cut}^+ G \to_{?N}^* t''^*$, there exists a context $((B_x, P), [!_v])$ such that $((B_x, P), [!_v]) \mapsto^* \pi_{t^* \to G} \circ \rho_{G \to t''^*}((B_{x''}, P''), [!_{v''}])$ (with x the lift of x'' in t) and $\pi_{t^* \to G} \circ \rho_{G \to t''^*}((\overline{d_{y''}}, Q''), [!_{\mathsf{e}}])$ is of the shape $((\overline{d_y}, Q), [!_{\mathsf{e}}])$ (with y the lift of y'' in t). By Lemma 38, $\rho_{G \to t''^*}((\sigma(B''), P''), [!_{v''}]) \mapsto^* \rho_{G \to t''^*}((\overline{d_{y''}}, Q''), [!_{\mathsf{e}}])$. And, by Lemma 13, we have $((\sigma(B_x), P), [!_v]) \mapsto^* \pi_{t^* \to G} \circ \rho_{G \to t''^*}((B_{x''}, P''), [!_{v''}]) \mapsto^* \pi_{t^* \to G} \circ \rho_{G \to t''^*}((\overline{d_{y''}}, Q''), [!_{\mathsf{e}}]) = ((\overline{d_y}, Q), [!_{\mathsf{e}}])$. □

Corollary 43. *Let t be a λ-term. If t^* is $\twoheadrightarrow$-stratified, then t is $\twoheadrightarrow_\lambda$-stratified.*

Proof. We prove it by contraposition: if there exists variables $x_1, \cdots, x_n$ of t such that $x_1 \twoheadrightarrow_\lambda x_2 \cdots \twoheadrightarrow_\lambda x_n \twoheadrightarrow_\lambda x_1$, then by Lemma 42 $B_{x_1} \twoheadrightarrow^+ B_{x_2} \cdots \twoheadrightarrow^+ B_{x_1}$. □

3.1.6 Simple characterization of Poly

Although $\twoheadrightarrow_\lambda$-stratification and $\twoheadrightarrow$-stratification give us a bound on the length of the reduction, elementary time is not considered as a reasonable bound. The complexity class we are interested in is polynomial time. We want to capture the set $Poly = \{P \mid P \text{ is a polynomial on } \mathbb{N}\}$. To understand how the complexity arises despite stratification, let us define for $k \in \mathbb{N}$, $Enc_\lambda(k) = \lambda f.\lambda x.(f)(f)\cdots(f)x$ (k successive applications of f to x) and let us consider the term $t = \lambda n.((n)\lambda w.(w)w)u$.

$$(t)Enc_\lambda(3) \to_\beta^* (\lambda x.(x)x)(\lambda y.(y)y)(\lambda z.(z)z)u \to_\beta^* (((u)u)(u)u)((u)u)(u)u$$

More generally, for any $k \in \mathbb{N}$, $(t)Enc_\lambda(k)$ reduces to a term of size 2^k. So t does not normalize in polynomial time[2]. However, for any $k \in \mathbb{N}$ $s_{\twoheadrightarrow_\lambda}(n) = 1$, $s_{\twoheadrightarrow_\lambda}(w) = 3$ and $s_{\twoheadrightarrow_\lambda}(u) = 4$. So $(t)Enc_\lambda(k)$ is $\twoheadrightarrow_\lambda$-stratified and we have $|\twoheadrightarrow_\lambda| = 4$ which does not depend on k.

To analyse this reduction, let us first define some terminology (inspired by the terminology on assignments in [16]). These definitions are not quite formal, their only purpose is to guide the intuition in the motivation of future definitions. Let us suppose that we have two quantities Q and R. We say that Q *depends additively* on R if we have a bound on Q of the shape $Q \leq R + a$. We say that Q *depends affinely* on R if we have a bound on Q of the shape $Q \leq b \cdot R + a$. We say that Q *depends multiplicatively* on R if we have a bound on Q of the shape $Q \leq b \cdot R^c + a$). We say that Q depends non-additively on R if there is no additive dependence of Q on R (*non-affine dependence* and *non-multiplicative dependence* are defined similarly). Let us notice that, for every proof-net G, if $W_{(G)H}$ depends non-multiplicatively on $|H|$ then the complexity of G is not polynomial.

The term $\lambda z.(z)z$ creates two residues of u. For each such residue u', $\lambda y.(y)y$ creates two residues of u'. And for each such residue u'', $\lambda x.(x)x$ creates two residues of u''. In other words, the total number of

[2]The number of $\to_\beta$ steps is linear in k, but those steps are not computable in polynomial time because of the size of the term. The corresponding reduction in proof-nets requires a non-polynomial number of $\to_{cut}$ steps.

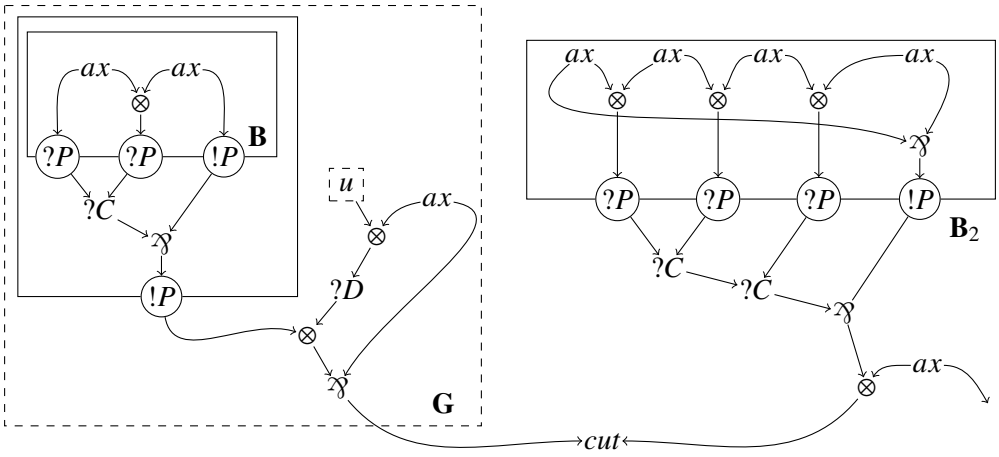


Figure 3.8: For every $k \in \mathbb{N}$ this proof-net, corresponding to $(t)Enc_\lambda(k)$, is $\twoheadrightarrow$-stratified but does not normalize in polynomial time.

residues of the argument of $\lambda z.(z)z$ depends affinely on the total number of residues of the argument of $\lambda y.(y)y$ which depends itself affinely on the total number of residues of the argument of $\lambda x.(x)x$. The composition of affine dependence is not an issue per se: let us suppose that for $1 \leq i \leq k$, we have $Q_i \leq b_i \cdot Q_{i+1}$. Then $Q_1 \leq (\Pi_{1 \leq i < k} a_i) \cdot Q_k$. Thus, Q_1 depends affinely on Q_k (for instance the number of residues of the argument of $\lambda x.(x)x$ is 4 times the number of residues of the argument of $\lambda z.(z)z$). However, if the length k of the sequence of dependence depends on R, then Q_1 depends non-multiplicatively on R. For instance, in the reduction of $(t)Enc_\lambda(n)$, the number of residues of u depends non-multiplicatively on n (so the normalization of $(t)Enc_\lambda(n)$ can not be computed in polynomial time).

To entail a polynomial bound, we add a restriction on λ-terms: if $\lambda x.u$ is a subterm of t there is at most one free occurrence of x in u. Then, the size of the λ-term strictly decreases along every $\rightarrow_{cut}$ reduction step so the λ-term reduces in linear time (so in polynomial time). However, this restriction is extremely strict. This restriction rules out most polynomial time λ-terms (for example, every λ-term which enjoys a polynomial time bound but not a linear time bound). To define subtler criteria, we will define them on proof-nets.

First, we will take a look at the proof-net of Figure 3.8, a proof-net "corresponding" to $(t)Enc_\lambda(3)$. This is not precisely the encoding we presented in Section 3.1.5, but another encoding. In terms of context semantics $(x)x$, $(y)y$ and $(z)z$ correspond respectively to $(B, [\mathtt{r}(\mathtt{e})])$, $(B, [\mathtt{l}(\mathtt{r}(\mathtt{e}))])$ and $(B, [\mathtt{l}(\mathtt{l}(\mathtt{e}))])$. Let us observe that for every signature t,

$$\begin{cases} ((\sigma(B), [\mathtt{l}(\mathtt{l}(\mathtt{e}))]), [!_{\mathtt{l}(t)}]) \mapsto^* ((\overline{\sigma_1(B)}, [\mathtt{l}(\mathtt{r}(\mathtt{e}))]), [!_t]) \mapsto ((\sigma(B), [\mathtt{l}(\mathtt{r}(\mathtt{e}))]), [!_t]) \\ ((\sigma(B), [\mathtt{l}(\mathtt{l}(\mathtt{e}))]), [!_{\mathtt{r}(t)}]) \mapsto^* ((\overline{\sigma_2(B)}, [\mathtt{l}(\mathtt{r}(\mathtt{e}))]), [!_t]) \mapsto ((\sigma(B), [\mathtt{l}(\mathtt{r}(\mathtt{e}))]), [!_t]) \end{cases}$$

Thus, for every copy t of $(B, [\mathtt{l}(\mathtt{r}(\mathtt{e}))])$, $\mathtt{l}(t)$ and $\mathtt{r}(t)$ are copies of $(B, [\mathtt{l}(\mathtt{l}(\mathtt{e}))])$. So the number of copies of $(B, [\mathtt{l}(\mathtt{l}(\mathtt{e}))])$ depends affinely on the number of copies of $(B, [\mathtt{l}(\mathtt{r}(\mathtt{e}))])$. Similarly $|Cop(B, [\mathtt{l}(\mathtt{r}(\mathtt{e}))])|$ depends affinely on $|Cop(B, [\mathtt{r}(\mathtt{e})])|$. And, in the general case the sequence of affine dependence has length $n-1$ and $|Cop(B, [\mathtt{l}(\mathtt{l}(\cdots \mathtt{l}(\mathtt{e}))])| = 2^{n-1} \cdot |Cop(B, [\mathtt{r}(\mathtt{e})])|$.

The affine dependences rely on the fact that B has 2 auxiliary doors. As we will prove with Lemma 45, the *ELL* proof-nets whose boxes have at most one auxiliary door normalize in polynomial time.

Lemma 44. *Let us consider a proof-net G whose boxes have at most one auxiliary door. Let us suppose that there exist paths $((\sigma(B),P),[!_t]) \mapsto^k ((e,Q),[!_{\mathtt{e}}])$ and $((\sigma(B),P),[!_{t'}]) \mapsto^{k'} ((e,Q),[!_{\mathtt{e}}])$. Then $t = t'$.*

Proof. Let us consider the paths, there exists sequences $(C_i)_{1\leq i\leq k}$ and $(C'_i)_{1\leq i\leq k'}$ such that

$$((\sigma(B),P),[!_t]) = C_k \mapsto C_{k-1} \cdots \mapsto C_2 \mapsto C_1 = ((e,Q),[!_{\mathtt{e}}])$$
$$((\sigma(B),P),[!_{t'}]) = C'_{k'} \mapsto C'_{k'-1} \cdots \mapsto C'_2 \mapsto C'_1 = ((e,Q),[!_{\mathtt{e}}])$$

We prove by induction on i that $C_i = C'_i$. Indeed, in the general case, the only non-injective step of $\mapsto$ is $\hookrightarrow$: for every potentiel box (B,P), signature t and $i,j \in \mathbb{N}$, $((\overline{\sigma_i(B)},P),[!_t]) \hookrightarrow ((\sigma(B),P),[!_t])$ and $((\overline{\sigma_j(B)},P),[!_t]) \hookrightarrow ((\sigma(B),P),[!_t])$. However, given the restriction that there is at most one auxiliary door by box of G, we know that $((\overline{\sigma_i(B)},P),[!_t]) \hookrightarrow ((\sigma(B),P),[!_t])$ implies $i = 1$. So this step is also injective.

If $k \neq k'$, let us assume without loss of generality that $k < k'$. Then, $((\sigma(B),P),[!_{t'}]) \mapsto^+ C'_k = C_k = ((\sigma(B),P),[!_t])$. This contradicts Lemma 21. Thus $k = k'$ and $C_k = C'_k$, so $t = t'$. □

In *LLL*, there are two types of boxes the "!-boxes" and the "§-boxes". The !-boxes have at most one auxiliary door. For every potential box (B,P) with B a §-box, we have $Cop(B,P) = \{\mathtt{e}\}$. This exactly the idea of the following lemma.

Lemma 45. *Let G be a $\twoheadrightarrow$-stratified proof-net. We suppose that for every box B having more than* 1 *auxiliary door, and $(B,P) \in Pot(B)$ we have $Cop(B,P) = \{\mathtt{e}\}$. Then, we have:*

$$|Cop(B,P)| \leq |\mathrm{E}_G|^{\partial_G{}^{2|\twoheadrightarrow|}}$$

Proof. We prove by induction on s that, if $(B,P) \in Can(B_G)$ and $s_{\twoheadrightarrow}(B) \leq s$, then $|Cop(B,P)| \leq \left(|\mathrm{E}_G|^{\partial_G}\right)^{2s}$. Indeed, let (B,P) be a canonical box with $s = s_{\twoheadrightarrow}(B,P)$ and $t \in Cop(B,P)$ then, by definition of copies:

$$((\sigma(B),P),[!_t]) \mapsto C_1 \mapsto \cdots \mapsto C_k = ((e,[q_1;\cdots;q_{\partial(e)}]),[!_{\mathtt{e}}])$$

We consider the minimal such path (for $i < k$, the leftmost trace element of C_i is not $!_{\mathtt{e}}$). For every box B' containing e, we have $s_{\twoheadrightarrow}(B') < s_{\twoheadrightarrow}(B) \leq s$. By induction hypothesis, the number of such contexts is bounded by:

$$|\mathrm{E}_G| \cdot \left(|\mathrm{E}_G|^{\partial_G^{2(s-1)}}\right)^{\partial_G} \leq |\mathrm{E}_G|^{1+\partial_G^{2\cdot s-1}} \leq |\mathrm{E}_G|^{\partial_G^{2\cdot s}}$$

If we prove that the choice of $(e,[q_1;\cdots;q_{\partial(e)}])$ entirely determines t, this gives us the expected bound. Let us consider $1 \leq i \leq k$ such that $C_i = ((\sigma(B'),P'),[!_{t'}])$. We know that $((\sigma(B),P),[!_t])$ is a canonical context so, by Lemma 23, $(B',P') \in Can(B_G)$ and $t' \in Cop(B',P')$. By assumption of minimality of the path, $t' \neq \mathtt{e}$. Thus, by assumption on the proof-net, B' has at most one auxiliary door. Thus, proving that the choice of $(e,[q_1;\cdots;q_{\partial(e)}])$ entirely determines t, can be done exactly as in the proof of Lemma 44. □

Corollary 46. *If G is a $\twoheadrightarrow$-stratified proof-net such that its boxes with more than one auxiliary door have at most* 1 *copy, then:*

$$W_G \leq |\mathrm{E}_G|^{\partial_G{}^{2(|\twoheadrightarrow|+1)}}$$

As in Subsection 3.1.4, it is realistic to suppose that the exponent does not depend on the argument of the function. So, such proof-nets normalize in polynomial time. In particular, Corollary 46 is general enough to directly imply the *Poly*-soundness of *LLL*.

3.1.7 Conclusion of this introduction

Section 3.1 is supposed to be an introduction to the more complex criteria we will define later. Here are some principles of this section which will still be used in the next criteria:

- We will define relations $\dot{\mathcal{R}}$ on potential boxes, such that $(B,P)\dot{\mathcal{R}}(C,Q)$ if the number of copies of (B,P) depends on the number of copies of (C,Q). To bound the number of copies of boxes, we have to limit the depth of $\dot{\mathcal{R}}$. Usually, we will require the collapse $\mathcal{R}$ of $\dot{\mathcal{R}}$ on boxes (i.e. the relation $\mathcal{R}$ defined by $B\mathcal{R}C$ iff $\exists P,Q,(B,P)\dot{\mathcal{R}}(C,Q)$) to be acyclic. Indeed, if $\mathcal{R}$ is acyclic, its depth is bounded by the number of boxes of the proof-net. And the number of boxes can be reasonably supposed to be independent of the argument.

- To prove elementary bounds, the idea will be to prove that it is impossible that $((\sigma(B),P),[!_t]) \mapsto^* ((e,Q),[!_u]) \mapsto^* ((e,Q'),[!_{u'}])$ with Q "similar" to Q'. So, the number of contexts of the shape $((e,Q),[!_u])$ in a path beginning by $((\sigma(B),P),[!_t])$ is bounded. This bounds the depth of any copy of (B,P), so $|Cop(B,P)|$ is bounded.

- To prove polynomial bounds, the idea will be to prove that if $((\sigma(B),P),[!_t]) \mapsto C_k \cdots \mapsto C_1 \mapsto ((e,Q),[!_{\mathtt{e}}])$, $((\sigma(B),P),[!_{t'}]) \mapsto C'_{k'} \cdots \mapsto C'_1 \mapsto ((e,Q'),[!_{\mathtt{e}}])$ and Q is "similar" to Q' then one can prove by induction on i that C_i is "similar" to C'_i. So, in particular, $k = k'$ and $t = t'$.

3.2 Elementary time

3.2.1 Definition of $\rightsquigarrow$-stratification

Examining the proof of Theorem 32, we can observe that the proof that the acyclicity of $\twoheadrightarrow$ entails an elementary bound on *cut*-elimination, relies on the following property:

Property 47 (Elementary stratification)**.** *For every potential box* (B, P) *and signature* $t \in Sig$*, if we have a bound on* $\max_{(B,P)\twoheadrightarrow(C,Q)} |Cop(C, Q)|$*, then we have a bound on the number of potential edges* (e, R) *such that* $\exists u \in Sig, ((\sigma(B), P), [!_t]) \mapsto^* ((e, R), [!_u])$.

The idea of this section is to identify *unnecessary* $(B, P)\dot{\twoheadrightarrow}(C, Q)$ pairs. It is to say, potential boxes (B, P) and (C, Q) such that $(B, P)\dot{\twoheadrightarrow}(C, Q)$ but bounding $|\{(e, R) \mid ((\sigma(B), P), [!_t]) \mapsto^* ((e, R), [!_u])\}|$ does not require to know a bound on $|Cop(C, Q)|$. Then, based on those examples, we will define a smaller relation $\rightsquigarrow$ which still enjoy the "Elementary stratification" property.

The first such example is whenever $B \subset C$ and no $\mapsto$ path from $((\sigma(B), P), [!_t])$ to $((e, R), [!_u])$ leaves the box C. In this case, the signature corresponding to C never changes along the path. So, whenever $((\sigma(B), P), [!_t]) \mapsto^* ((e, R), [!_u]) \mapsto^* ((e, R'), [!_{u'}])$ the signature corresponding to C is the same in P, R and R'. This is why, knowing $|Cop(C, Q)|$ is not necessary to bound $|Cop(B, P)|$.

More formally, let us define E_B as the set of boxes C such that $((\sigma(B), [p_1; \cdots ; p_{\partial(B)}]), [!_t]) \mapsto^* ((e, Q), [!_u])$ with $e \in C$ and at least one edge in the sequence of contexts is outside C. Let us suppose that there exists $k \in \mathbb{N}$ such that for every $C \in E_B$, we have $\max_{(C,Q)\in Pot} |Cop(C, Q)| \leq k$. Finally, let us suppose that $((\sigma(B), [p_1; \cdots ; p_{\partial(B)}]), [!_t]) \mapsto^* ((e, [q_1; \cdots ; q_{\partial(e)}]), [!_u]) \mapsto^* ((e, [q'_1; \cdots ; q'_{\partial(e)}]), [!_v])$ with $e \in B_{\partial(e)} \subset \cdots \subset B_1$ and $q_i = q'_i$ for every $B_i \in E_B$. Then, for $1 \leq i \leq \partial(e)$ either the path did not go out of B_i (so $q_i = q'_i = p_i$) or $B_i \in E_B$ (so $q_i = q'_i$ by supposition). Thus $[q_1; \cdots ; q_{\partial(e)}] = [q'_1; \cdots ; q'_{\partial(e)}]$ which is a contradiction because of Lemma 21. Thus $|\{(e, Q) \mid ((\sigma(B), P), [!_t]) \rightsquigarrow^* ((e, Q), [!_u])\}| \leq |\vec{E}_G| \cdot k^{\partial_G}$. We proved that such pairs are unnecessary to enforce the elementary stratification property.

Thus, $B \twoheadrightarrow C$ couples are necessary only if there is a $\mapsto$ path from $((\sigma(B), P), [!_t])$ which enters C by one of its doors (either auxiliary or principal). In fact, we will prove that the $B \twoheadrightarrow C$ couples are necessary only if there is a $\mapsto$ path from $((\sigma(B), P), [!_t])$ which enters C by its *principal* door. To understand why, we will study an example. In Figure 3.9, if $((\sigma(D), P), [!_t]) \mapsto^* ((\overline{w}, [q_A; q_B]), [!_{\mathtt{e}}])$ and $((\sigma(D), P'), [!_{t'}]) \mapsto^* ((\overline{w}, [q'_A; q'_B]), [!_{\mathtt{e}}])$, we only need to know q_B to trace those paths back (i.e to deduce the list of edges of those paths). Indeed the paths do not enter A by its principal door, so q_A and q'_A can only appear on $!_{\_}$ trace elements, never on $?_{\_}$ trace elements. Thus, if $q_B = q'_B$, the paths take the same edges and $t = t'$. We do not need to know that $q_A = q'_A$ so $D \twoheadrightarrow A$ is an unnecessary pair. On the contrary,

$$\begin{cases} ((\sigma(B), [\mathtt{e}]), [!_{\mathtt{l}(\mathtt{r}(\mathtt{e}))}]) \mapsto^9 ((\overline{d}, [\mathtt{r}(\mathtt{l}(\mathtt{e}))]), [!_{\mathtt{e}}]) \\ ((\sigma(B), [\mathtt{e}]), [!_{\mathtt{r}(\mathtt{r}(\mathtt{e}))}]) \mapsto^9 ((\overline{d}, [\mathtt{l}(\mathtt{l}(\mathtt{e}))]), [!_{\mathtt{e}}]) \end{cases}$$

So $B \twoheadrightarrow C$ is a necessary pair. Tracing those paths backwards, the difference in the potential corresponding to C becomes a difference in a $?_{\_}$ trace element (in the $((\overline{\sigma(C)}, []), [!_{\mathtt{e}}; ?_v]) \mapsto ((\overline{d}, [v]), [!_{\mathtt{e}}])$ step). And because of this difference on a $?_{\_}$ trace element, the reverse paths separate when the paths cross a $?C$ node downwards: $((g, []), [!_{\mathtt{e}}; ?_{\mathtt{l}(\mathtt{e})}]) \mapsto ((f, []), [!_{\mathtt{e}}; ?_{\mathtt{l}(\mathtt{l}(\mathtt{e}))}])$ and $((h, []), [!_{\mathtt{e}}; ?_{\mathtt{l}(\mathtt{e})}]) \mapsto ((f, []), [!_{\mathtt{e}}; ?_{\mathtt{r}(\mathtt{l}(\mathtt{e}))}])$.

We define a relation $\rightsquigarrow$ between boxes of proof nets. $B \rightsquigarrow C$ means that there is a path beginning by the principal door of B which enters C by its principal door. We first define the relation on potential boxes as it will be useful in some proofs. Throughout this thesis, many relations on boxes and potential boxes are defined. As a mnemotechnic mean, every time we define a relation $\dot{R}$ on potential boxes (a symbol with a dot), we will define R (the same symbol without the dot) by $B\ R\ C \Leftrightarrow \exists P, Q \in Pot, (B, P)\dot{R}(C, Q)$.

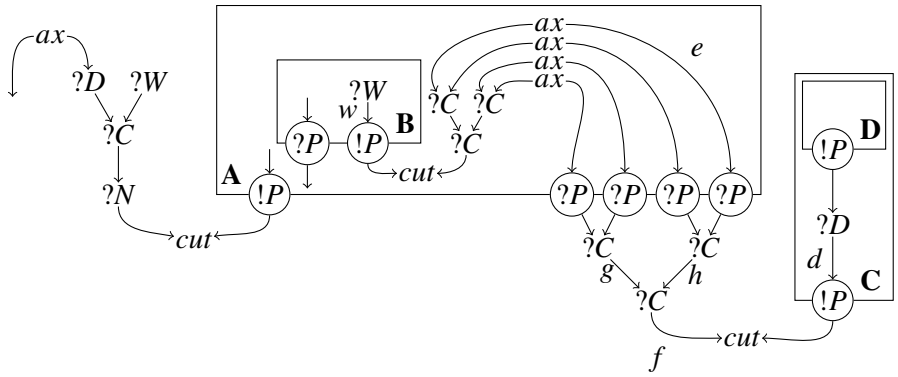


Figure 3.9: $D \twoheadrightarrow A$ but it is an "unnecessary" couple because $|Cop(B,P)$ does not depend on $|Cop(A,[])|$.

Definition 48. *Let (B,P) and (C,Q) be potential boxes. Then we write $(B,P)\dot{\rightsquigarrow}(C,Q)$ if there exist $t \in Sig$ and $T \in Tra$ such that:*

$$((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(C)},Q),T)$$

We define the relation $\rightsquigarrow$ on boxes by $B \rightsquigarrow C$ iff there exist $P,Q \in Pot$ such that $(B,P)\dot{\rightsquigarrow}(C,Q)$.

We can notice that for every proof-net, $\rightsquigarrow \subseteq \twoheadrightarrow$. For instance in the proof-net of Figure 3.9, we have $B \twoheadrightarrow A$, $B \twoheadrightarrow C$, $D \twoheadrightarrow C$, $D \twoheadrightarrow A$ and $D \twoheadrightarrow B$. Whereas, in $\rightsquigarrow$ we only have two pair: $D \rightsquigarrow B$, and $B \rightsquigarrow C$. So every $\twoheadrightarrow$-stratified proof-net is $\rightsquigarrow$-stratified, and $s_{\rightsquigarrow}(B) \leq s_{\rightsquigarrow}(C)$.

The proof-net of Figure 3.10 is $\rightsquigarrow$ stratified but not $\twoheadrightarrow$ stratified. Indeed, the only $\rightsquigarrow$ pairs are $B \rightsquigarrow C$ and $B \rightsquigarrow A$ whereas $\twoheadrightarrow$ also contains the pair $A \twoheadrightarrow B$. If a proof-net does not normalize in elementary time, then it is not $\rightsquigarrow$-stratified. For instance, the proofnet of Figure 3.12 (which represents the Ackermann function applied to 3) is not $\rightsquigarrow$-stratified because $B_1 \rightsquigarrow B_1$. Indeed if we set $U = [!_{\mathtt{e}}; \otimes_l; \exists]$ and $T = U@[\otimes_r; ?_{\mathtt{l(r(e))}}; \parr_l; \forall]$, then

$$\begin{array}{lll}
((\sigma(B_1),[\mathtt{l(l(e))}]),[!_{\mathtt{n(l(e),n(l(r(e)),e))}}]) & \rightsquigarrow^{14} ((\overline{\sigma_1(B_2)},[\mathtt{r}(e)]),[!_{\mathtt{n(l(e),n(l(r(e)),e))}};\parr_r;!_{\mathtt{e}}]) & \rightsquigarrow^{5} \\
((\sigma_2(B_2),[\mathtt{r(e)}]),[!_{\mathtt{n(l(e),n(l(r(e)),e))}};\otimes_l;?_{\mathtt{e}}]) & \rightsquigarrow^{15} ((\overline{\sigma_1(B_1)},[\mathtt{l(r(e))}]),[!_{\mathtt{l(e)}};!_{\mathtt{n(l(r(e)),e)}}]) & \rightsquigarrow^{19} \\
((\sigma(B_1),[\mathtt{l(r(e))}]),U@[\parr_l;!_{\mathtt{n(l(r(e)),e)}}]) & \rightsquigarrow^{53} ((\sigma(B_1),[\mathtt{r(e)}]),U@[\parr_l;!_{\mathtt{r(e)}};\otimes_l;\exists;\parr_l;!_{\mathtt{e}}]) & \rightsquigarrow^{18} \\
((\sigma(B_2),[\mathtt{r(e)}]),U@[\parr_l;!_{\mathtt{r(e)}};\otimes_l;\exists;\parr_l;!_{\mathtt{e}};\parr_r;!_{\mathtt{e}}]) & \rightsquigarrow^{14} ((\overline{e},[]),U@[\parr_l;!_{\mathtt{r(e)}};\otimes_l;\exists]) & \rightsquigarrow^{10} \\
((\overline{\sigma_3(B_2)},[\mathtt{l(e)}]),U@[\parr_l;!_{\mathtt{e}}]) & \rightsquigarrow^{16} ((e,[]),T) & \rightsquigarrow^{14} ((\overline{\sigma(B_2)},[\mathtt{r(e)}]),T@[\otimes_l;?_{\mathtt{e}};\otimes_r;?_{\mathtt{e}}]) & \rightsquigarrow^{5} \\
((\sigma_3(B_2),[\mathtt{r(e)}]),T@[\otimes_l;?_{\mathtt{e}};\otimes_r;?_{\mathtt{e}}]) & \rightsquigarrow^{13} ((\overline{\sigma(B_1)},[\mathtt{r(e)}]),T@[\otimes_l;?_{\mathtt{e}}]) &
\end{array}$$

Claim 1 As hinted previously, if there exist $\rightsquigarrow$-paths $((\sigma(B),P),[!_t]) \rightsquigarrow^* ((e,[q_1;\cdots;q_{\partial(e)}]),[!_u])$ and $C' \rightsquigarrow^* ((e,[q'_1;\cdots;q'_{\partial(e)}]),[!_{u'}])$ with $e \in B_{\partial(e)} \subset \cdots B_1$ and ($B \rightsquigarrow B_i \Rightarrow q_i = q'_i$), then one can trace back the two paths and they will never separate. Both paths pass exactly by the same sequences of edges. Knowing q_i for every box B_i (possibly) entered by its principal door is enough to know where the path came from.

For example, in the proof-net of Figure 3.11, $B \rightsquigarrow A''$ and $((\sigma(B),[\mathtt{n(e,l(e))};\mathtt{r(e)}]),[!_{\mathtt{r(e)}}]) \rightsquigarrow^{16} ((\overline{w},[\mathtt{r(n(e,l(e)))}]),[!_{\mathtt{e}}])$ (and we write $e_1,\cdots,e_{16}$ the edges of this path). And indeed (because $\rightsquigarrow$ is

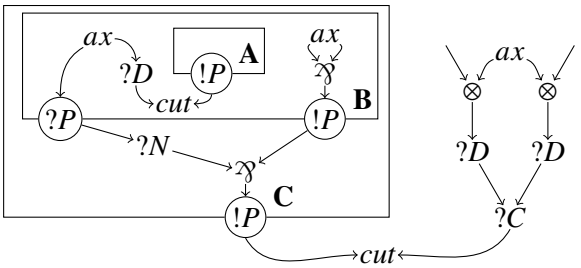


Figure 3.10: This proof-net is $\rightsquigarrow\!\!\!\!\rightarrow$ stratified but not $\twoheadrightarrow$ stratified.

injective) if $C \rightsquigarrow^{16} ((\overline{w}, [\mathtt{r}(\mathtt{n}(\mathtt{e}, \mathtt{r}(\mathtt{e})))]), [!_{\mathtt{e}}])$ then the edges of the path are $e_1, \cdots, e_{16}$. One can observe that to trace back the path, it is necessary to have information on which copy of A'' we are in. Indeed:

$$\begin{cases} ((\sigma_1(D), []), [!_{\mathtt{e}}; ⅋_r; !_{\mathtt{r}(\mathtt{e})}; ?_{\mathtt{e}}]) \rightsquigarrow^6 ((\overline{w}, [\mathtt{l}(\mathtt{e})]), [!_{\mathtt{e}}]) \\ ((\sigma_2(D), []), [!_{\mathtt{e}}; ⅋_r; !_{\mathtt{r}(\mathtt{e})}; ?_{\mathtt{e}}]) \rightsquigarrow^6 ((\overline{w}, [\mathtt{r}(\mathtt{e})]), [!_{\mathtt{e}}]) \end{cases}$$

This is coherent with the result we stated, because the path enters A'' by its principal door. However, let us notice that it is not necessary to know exactly in which copy we are. The only information needed to trace back the path is that it is of the form $\mathtt{r}(x)$. Knowing that $x = \mathtt{n}(\mathtt{e}, \mathtt{r}(\mathtt{e}))$ is useless because the information in x would only be used if the path entered D by its principal door and that is not the case.

The aim of the following definitions is to formalize the notion of the information needed to trace back the paths. The $\mapsto_S$-copies of a potential box (B, P) are the copies t of (B, P) such that there exists a path of the shape $((\sigma(B), P), [!_t]) \mapsto ((\_, \_), [!_{\mathtt{e}}])$ such that, for every $((\overline{\sigma\_(C)}, Q), [!_u]) \hookrightarrow ((\sigma(C), Q), [!_u])$ step of the path, $C \in S$. For instance, in the proof-net of Figure 3.11, the $\mapsto_{\{A''\}}$-copies of $(A'', [])$ are $\mathtt{e}$, $\mathtt{l}(\mathtt{e})$ and $\mathtt{r}(\mathtt{e})$. And the set of $\mapsto_{\{A'',D\}}$-copies of $(A'', [])$ is exactly the set of copies of $(A'', [])$: $Cop(A'', []) = \{\mathtt{e}, \mathtt{l}(\mathtt{e}), \mathtt{r}(\mathtt{e}), \mathtt{l}(\mathtt{n}(\mathtt{e}, \mathtt{e})), \mathtt{r}(\mathtt{n}(\mathtt{e}, \mathtt{e})), \mathtt{l}(\mathtt{n}(\mathtt{e}, \mathtt{l}(\mathtt{e}))), \mathtt{r}(\mathtt{n}(\mathtt{e}, \mathtt{l}(\mathtt{e}))), \mathtt{l}(\mathtt{n}(\mathtt{e}, \mathtt{r}(\mathtt{e}))), \mathtt{r}(\mathtt{n}(\mathtt{e}, \mathtt{r}(\mathtt{e})))\}$

To be a little more precise, the $\mapsto_S$-copies of a potential box (B, P) are the copies t of (B, P) such that there exists a path of the shape $((\sigma(B), P), [!_t]) \mapsto ((e, R), [!_{\mathtt{e}}])$ such that if there is a context $((\sigma(C), Q), [!_u])$ in the path with $C \notin S$, then the path stops there. So, in particular as we told, if there is a $((\overline{\sigma\_(C)}, Q), [!_u]) \hookrightarrow ((\sigma(C), Q), [!_u])$ step in the path with $C \notin S$, then $((e, R), [!_{\mathtt{e}}]) = ((\sigma(C), Q), [!_u])$ so $((\sigma(B), P), [!_t]) \mapsto^* ((\overline{\sigma\_(C)}, Q), [!_{\mathtt{e}}])$ and for every $((\overline{\sigma\_(C')}, Q'), [!_{u'}]) \hookrightarrow ((\sigma(C'), Q'), [!_{u'}])$ step of this path, $(C', Q', u') \in S$. Moreover, with this new definition, if $B \notin S$, the set of $\mapsto_S$-copies of (B, P) is $\{\mathtt{e}\}$.

Then, we will define $\mapsto_S$-canonical potentials and $\mapsto_S$-canonical contexts from the notion of $\mapsto_S$-copies in the same way as we defined canonical potentials and canonical contexts from the notion of copies.

So, in the proof-net of Figure 3.11, if we suppose that we know that $C \rightsquigarrow^k ((\overline{w}, [t]), [!_{\mathtt{e}}])$ and $C' \rightsquigarrow^k ((\overline{w}, [t']), [!_{\mathtt{e}}])$ and we want to prove that those paths take the same edges. We only need to know that the $\mapsto_{\{A''\}}$-copies of $(A'', [])$ "corresponding" to t and t' are equal. We define u (resp. u') as the "biggest" $\mapsto_{\{A''\}}$-copy of $(A'', [])$ which is a "truncation" of t (resp. t'). For instance, if $t = \mathtt{r}(\mathtt{n}(\mathtt{e}, \mathtt{l}(\mathtt{e})))$ and $t' = \mathtt{r}(\mathtt{n}(\mathtt{e}, \mathtt{r}(\mathtt{e})))$, then $t \neq t'$. But the $\mapsto_{\{A''\}}$-copy of $(A'', [])$ corresponding to t will be $u = \mathtt{r}(\mathtt{e})$. And the $\mapsto_{\{A''\}}$-copy of $(A'', [])$ corresponding to t' will be $u' = \mathtt{r}(\mathtt{e})$. Knowing that $u = u' = \mathtt{r}(\mathtt{e})$ is enough to know that t and t' are of the shape $\mathtt{r}(x)$ and $\mathtt{r}(x')$ and, as we observed before, this information is enough to trace back the paths, so to prove that the paths take the same edges.

The $\mapsto_S$-copy of (B, P) corresponding to t will be written $((\sigma(B), P), [!_t])^{/\mapsto_S}$. It represents the part of t which is used if we refuse the $\hookrightarrow$ steps over the potential boxes which are not in S. For instance, in Figure 3.11, $((\sigma(A''), []), [!_{\mathtt{r}(\mathtt{n}(\mathtt{e}, \mathtt{l}(\mathtt{e})))}])^{/\mapsto_{\{A''\}}} = \mathtt{r}(e)$ and $((\sigma(A''), []), [!_{\mathtt{r}(\mathtt{n}(\mathtt{e}, \mathtt{r}(\mathtt{e})))}])^{/\mapsto_{\{A''\}}} = \mathtt{r}(\mathtt{e})$ because, if we refuse to jump over $(D, [])$, only $\mathtt{r}(\_)$ is consumed in the $\mapsto$ paths starting from those contexts. Then,

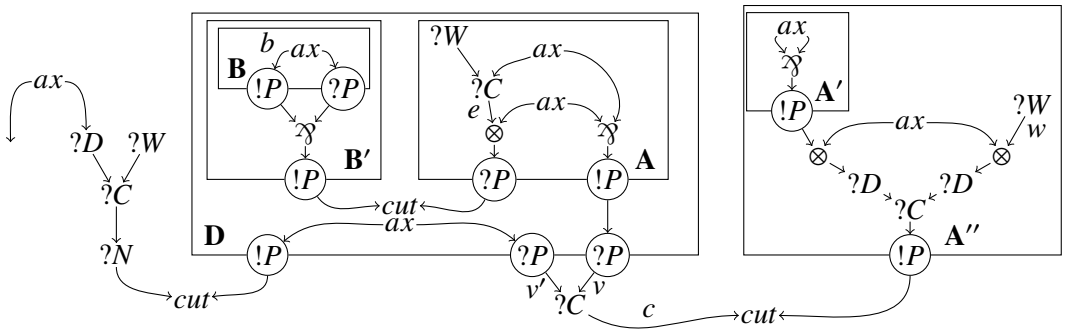


Figure 3.11: $((\sigma(B), [x_D; \mathtt{l(e)}]), [!_{\mathtt{r(r(e))}}]) \mapsto^{33} ((\sigma(B), [x_D; \mathtt{r(e)}]), [!_{\mathtt{r(e)}}]) \mapsto^{16} ((\overline{w}, [\mathtt{r}(x_D)]), [!_{\mathtt{e}}])$

$(e, P)^{\mapsto_S}$ will be defined from the $((\sigma(B), P), [!_t])^{/\mapsto_S}$ construction in the same way as canonical potentials are defined from copies: if $e \in B_n \subset \cdots \subset B_1$, then $(e, [p_1; \cdots; p_n])^{\mapsto_S} = (e, [q_1; \cdots; q_n])$ such that, for $1 \leq i \leq n$, $((\sigma(B_i), [q_1; \cdots; q_{i-1}]), [!_{p_i}])^{/\mapsto_S} = q_i$. For example, in Figure 3.11, $(\overline{w}, [\mathtt{r(n(e, l(e)))}])^{\mapsto_{\{A''\}}} = (\overline{w}, [\mathtt{r(n(e, r(e)))}])^{\mapsto_{\{A''\}}} = (\overline{w}, \mathtt{r(e)})$. Thus, if we know that $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{w}, Q), [!_{\mathtt{e}}])$, it is enough to know that $(\overline{w}, Q)^{\mapsto_{\{(C,[])\}}}$ to trace back the path.

Claim 2 We can precise Claim 1 as follows: let (B, P) be a potential box and S be the set of boxes C such that $s_{\rightsquigarrow}(C) < s_{\rightsquigarrow}(B)$. If there exist $\rightsquigarrow$-paths $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((e, Q), [!_u])$ and $C' \rightsquigarrow^* ((e, Q'), [!_{u'}])$ with $(e, Q)^{\mapsto_S} = (e, Q')^{\mapsto_S}$, then one can trace back the two paths and they will never separate.

3.2.2 Restricted copies and canonical potentials

Now that we gave the intuition and motivation behind the definition, we can give the formal definitions.

Definition 49. *Let G be a proof-net and $S \subset B_G$. We define $\mapsto_S$ and $\rightsquigarrow_S$ as follows:*

$$C \mapsto_S D \Leftrightarrow \begin{cases} C \mapsto D \\ \text{If } C = ((\sigma(B), P), [!_t]), \text{ then } B \in S \end{cases}$$

$$C \rightsquigarrow_S D \Leftrightarrow \begin{cases} C \rightsquigarrow D \\ \text{If } D = ((\overline{\sigma(B)}, P), T.?_t), \text{ then } B \in S \end{cases}$$

In Section 3.2.1, we gave some intuition on $\mapsto_S$-copies, but in further section we will need a notion of $\rightarrow$-copies for other relations on contexts. In the following, we suppose given a relation $\rightarrow$ on contexts such that $\rightarrow \subseteq \mapsto$.

Definition 50. *A $\rightarrow$-copy context is a context of the shape $((e, P), [!_t]@T)$ such that for every $u \sqsupseteq t$, there exists a path of the shape $((e, P), [!_u]@T) \rightarrow^* ((\_, \_), [!_{\mathtt{e}}])$.*

Let $(B, P) \in Pot(B_G)$, the set $Cop_{\rightarrow}(B, P)$ of copies of (B, P) is the set of standard signatures t such that $((\sigma(B), P), [!_t])$ is a $\rightarrow$-copy context.

For instance, in Figure 3.11, the set of $\mapsto_\varnothing$-copies of $(D, [])$ is $\{\mathtt{e}\}$. Similarly, the set of $\mapsto_\varnothing$-copies of $(B', [\mathtt{e}])$ is $\{\mathtt{e}\}$. Let us consider a set S containing $\{A, B'\}$, the set of $\mapsto_S$-copies of $(B', [\mathtt{e}])$ is $\{\mathtt{e}, \mathtt{l(e)}, \mathtt{r(e)}\}$. The set of $\mapsto_S$-copies of $(B, [\mathtt{e}; \mathtt{e}])$ and the set of $\mapsto_S$-copies of $(B, [\mathtt{e}; \mathtt{r(e)}])$ are both equal to $\{\mathtt{e}, \mathtt{l(e)}, \mathtt{r(e)}\}$. On the contrary, for $x \in Sig$, $((\sigma(B), [\mathtt{e}; \mathtt{l(e)}]), [!_{\mathtt{r}(x)}]) \rightsquigarrow^* ((\overline{\sigma_1(B)}, [\mathtt{e}; \mathtt{r(e)}]), [!_x])$. So, the set of $\mapsto_{\{A,B,B'\}}$-copies of $(B, [\mathtt{e}; \mathtt{l(e)}])$ is $\{\mathtt{e}, \mathtt{l(e)}, \mathtt{r(e)}, \mathtt{r(l(e))}, \mathtt{r(r(e))}\}$ whereas the set of $\mapsto_{\{A,B'\}}$-copies of $(B, [\mathtt{e}; \mathtt{l(e)}])$ is only $\{\mathtt{e}, \mathtt{l(e)}, \mathtt{r(e)}\}$.

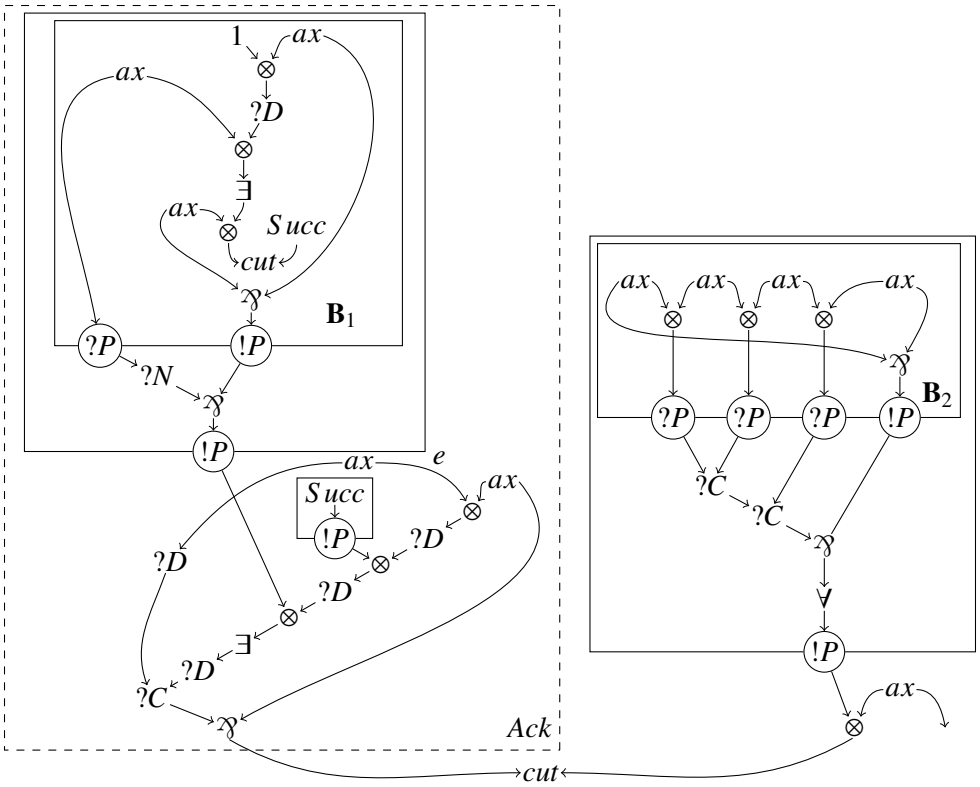


Figure 3.12: This proof-net, representing the Ackermann function applied to 3, is not $\rightsquigarrow$-stratified.

Definition 51. *Let x be an element (edge or node) of G such that $x \in B_{\partial(x)} \subset ... \subset B_1$. The set $Can_{\rightarrow}(x)$ of $\rightarrow$-canonical potentials for x is the set of potentials $[s_1; ...; s_{\partial(e)}]$ such that:*

$$\forall 1 \leq i \leq \partial(x), s_i \in Cop_{\rightarrow}(B_i, [s_1; \cdots ; s_{i-1}])$$

For instance, in Figure 3.11, we can observe that $Can_{\mapsto_{\{A,B'\}}}(D) = Can_{\mapsto_{\{A,B,B'\}}}(D) = \{(D, [])\}$. Then, because $D \in \{A, B, B'\}$, $Cop_{\mapsto_{\{A,B,B'\}}}(D, []) = Cop_{\mapsto_{\{A,B'\}}}(D, []) = \{\mathtt{e}\}$ so $Can_{\mapsto_{\{A,B'\}}}(B') = Can_{\mapsto_{\{A,B,B'\}}}(B') = \{(B', [\mathtt{e}])\}$. We noticed in page 61 that $Cop_{\mapsto_{\{A,B'\}}}(B', [\mathtt{e}]) = Cop_{\mapsto_{\{A,B,B'\}}}(B', [\mathtt{e}]) = \{e, \mathtt{l(e)}, \mathtt{r(e)}\}$. Thus, we have $Can_{\mapsto_{\{A,B'\}}}(B) = Can_{\mapsto_{\{A,B,B'\}}}(B) = \{(B, [\mathtt{e}; \mathtt{e}]), (B, [\mathtt{e}; \mathtt{r(e)}]), (B, [\mathtt{e}; \mathtt{l(e)}])\}$. Finally:

$$Can_{\mapsto_{\{A,B'\}}}(b) = \Big\{(b, [\mathtt{e}; \mathtt{e}; \mathtt{e}]), (b, [\mathtt{e}; \mathtt{e}; \mathtt{l(e)}]), (b, [\mathtt{e}; \mathtt{e}; \mathtt{r(e)}]), (b, [\mathtt{e}; \mathtt{r(e)}; \mathtt{e}]), (b, [\mathtt{e}; \mathtt{r(e)}; \mathtt{l(e)}]),$$
$$(b, [\mathtt{e}; \mathtt{r(e)}; \mathtt{r(e)}]), (b, [\mathtt{e}; \mathtt{l(e)}; \mathtt{e}]), (b, [\mathtt{e}; \mathtt{l(e)}; \mathtt{l(e)}]), (b, [\mathtt{e}; \mathtt{l(e)}; \mathtt{r(e)}])\Big\}$$

$$Can_{\mapsto_{\{A,B,B'\}}}(b) = \Big\{(b, [\mathtt{e}; \mathtt{e}; \mathtt{e}]), (b, [\mathtt{e}; \mathtt{e}; \mathtt{l(e)}]), (b, [\mathtt{e}; \mathtt{e}; \mathtt{r(e)}]), (b, [\mathtt{e}; \mathtt{r(e)}; \mathtt{e}]), (b, [\mathtt{e}; \mathtt{r(e)}; \mathtt{l(e)}]),$$
$$(b, [\mathtt{e}; \mathtt{r(e)}; \mathtt{r(e)}]), (b, [\mathtt{e}; \mathtt{l(e)}; \mathtt{e}]), (b, [\mathtt{e}; \mathtt{l(e)}; \mathtt{l(e)}]), (b, [\mathtt{e}; \mathtt{l(e)}; \mathtt{r(e)}]),$$
$$(b, [\mathtt{e}; \mathtt{l(e)}; \mathtt{r(l(e))}]), (b, [\mathtt{e}; \mathtt{l(e)}; \mathtt{r(r(e))}])\Big\}$$

In Definition 22, we defined a notion of canonical contexts stable by $\mapsto$ paths. Similarly, we define in Definition 52 a notion of $\rightarrow$-canonical contexts. In the general case, such paths are not stable by $\rightarrow$ paths. For instance, in Figure 3.13, $((\overline{c}, []), [!_{\mathtt{e}}; ?_{\mathtt{l(e)}}; \parr_r]) \mapsto_{\varnothing} ((\overline{\sigma(C)}, []), [!_{\mathtt{e}}; ?_{\mathtt{l(e)}}])$ and $((\overline{c}, []), [!_{\mathtt{e}}; ?_{\mathtt{l(e)}}; \parr_r])$ is a $\mapsto_{\varnothing}$-canonical context, but $((\overline{\sigma(C)}, []), [!_{\mathtt{e}}; ?_{\mathtt{l(e)}}])$ is not $\mapsto_{\varnothing}$-canonical because $((\sigma(C), []), [!_{\mathtt{l(e)}}]) \not\mapsto_{\varnothing}$

Definition 52. *A quasi-standard context*[3] $C = ((e, P), [T_1; \cdots; T_k])$ *is* $\to$-canonical *if* $(e, P) \in Can_{\to}(e)$ *and:*

- *For every* $T_i = !_t$, $((e, [P_1; \cdots; P_{\partial(e)}]), [!_t; T_{i+1}; \cdots; T_k])$ *is a* $\to$*-copy context.*
- *For every* $T_i = ?_t$, $((\overline{e}, [P_1; \cdots; P_{\partial(e)}]), [!_t; T_{i+1}^{\perp}; \cdots; T_k^{\perp}])$ *is a* $\to$*-copy context.*

We can notice that, in particular, the definitions of $Cop_{\mapsto}(B, P)$, $Can_{\mapsto}(x)$ and $\mapsto$-canonical contexts match respectively the definitions of $Cop(B, P)$ (Definition 10 in page 26) , $Can(x)$ (Definition 11) and canonical contexts (Definition 22, page 35).

Let us consider a potential box (B, P) and $t \in Cop(B, P)$, then there exists a context $((e, Q), [!_{\mathtt{e}}])$ such that $((\sigma(B), P), [!_t]) \mapsto^* ((e, Q), [!_{\mathtt{e}}])$. If some of the $\mapsto$ steps of this path are not in $\to$, then we may have $((\sigma(B), P), [!_t]) \to^* ((f, R), [!_v]) \not\to$ with $v \neq \mathtt{e}$. In this case, t would not be a $\to$-copy of (B, P). As we wrote in Section 3.2.1, we want to define $((\sigma(B), P), [!_t])/_{\to}$ as the "biggest" $\to$-copy t' of (B, P) which is a "truncation" of t. As an intuition, let us consider the contexts $((f, R), [!_v])$ and $((f, R), [!_{\mathtt{e}}])$. By the following Lemma 53, we deduce that there exist $t' \in Sig$ such that $((\sigma(B), P), [!_{t'}]) \mapsto^* ((f, R), [!_{\mathtt{e}}])$. And with most of the relations $\to$ we will consider, we will have $((\sigma(B), P), [!_{t'}]) \to^* ((f, R), [!_{\mathtt{e}}])$[4]. In this case and in the absence of $\mathtt{n}(\_)$ construction in t, we would have $((\sigma(B), P), [!_t])/_{\to} = t'$. Thus t' represents the part of t which is consumed in the part of the $((\sigma(B), P), [!_t]) \mapsto^* ((e, Q), [!_{\mathtt{e}}])$ path which is a $\to$-path.

Lemma 53. *If* $((e, P), [!_t]@T) \mapsto^* ((f, Q), [!_u]@U)$*, for every* $u' \in Sig$*, there exists* $t' \in Sig$ *such that* $((e, P), [!_{t'}]@T) \mapsto^* ((f, Q), [!_{u'}]@U)$.

Proof. Because traces can not be empty, the leftmost trace element is never popped. So in every context of the path, the leftmost trace element is a $!_{\_}$ trace element. Thus, it is enough to prove the lemma in the case of a single $\mapsto$ step, i.e. whenever $((e, P), [!_t]@T) \mapsto ((f, Q), [!_u]@U)$.

The only steps which modify this trace element are when we cross a $?C$ or $?N$ node upwards. If $((e, P), [!_{\mathtt{n}(u,v)}]) \mapsto ((f, P), [!_u; !_v])$, for every $u' \in Sig$, we set $t' = \mathtt{n}(u', v)$ and we can notice that we indeed have $((e, P), [!_{t'}]) \mapsto ((f, P), [!_{u'}; !_v])$. The other cases are similar. □

As we wrote, we want to define $((\sigma(B), P), [!_t])/_{\to}$ as the "biggest" $\to$-copy t' of (B, P) which is a "truncation" of t. But we did not precise the meaning of "biggest" and "truncation". First, we say that t is a truncation of u if $t \blacktriangleleft u$ with $\blacktriangleleft$ defined as follows:

Definition 54. *We define an order* $\blacktriangleleft$ *on signatures by induction. For every signature* t, t', u, u'*, we set* $\mathtt{e} \blacktriangleleft t$ *and if we suppose* $t \blacktriangleleft t'$ *and* $u \blacktriangleleft u'$ *then* $\mathtt{l}(t) \blacktriangleleft \mathtt{l}(t')$, $\mathtt{r}(t) \blacktriangleleft \mathtt{r}(t')$, $\mathtt{p}(t) \blacktriangleleft \mathtt{p}(t')$ *and* $\mathtt{n}(t, u) \blacktriangleleft \mathtt{n}(t', u')$.

As a first attempt, one might say that "biggest" means "the maximum for the order $\blacktriangleleft$". However, in the general case, the set of $\to$-copies of (B, P) which are truncations of t may not have a maximum. For instance, in Figure 3.13, if we set $t = \mathtt{n}(\mathtt{l}(\mathtt{e}), \mathtt{r}(\mathtt{e}))$ and $\to$ as the relation obtained from $\mapsto$ by removing the transition $((\sigma(D), [\mathtt{r}(\mathtt{e})]), [!_{\mathtt{l}(\mathtt{e})}]) \mapsto ((\overline{d}, [\mathtt{r}(\mathtt{e})]), [!_{\mathtt{l}(\mathtt{e})}])$, then we can observe that $\mathtt{n}(\mathtt{l}(\mathtt{e}), \mathtt{e})$ and $\mathtt{n}(\mathtt{e}, \mathtt{r}(\mathtt{e}))$ are $\to$ copies of $(B, [])$ but $\mathtt{n}(\mathtt{l}(\mathtt{e}), \mathtt{r}(\mathtt{e}))$ is not a $\to$-copy of $(B, [])$. So, the set of $\to$ copies of $(B, [])$ which are truncations of t has 2 maximal elements.

The solution we chose is to first maximize the rightmost branch. Then, once this branch is fixed, we maximize the second rightmost branch and so on. Formally, we define "biggest" as "the maximum for the order $\trianglelefteq$" with $\trianglelefteq$ defined as follows.

[3] A context is said quasi-standard if its signatures are standard except for the leftmost trace element (Definition 6, page 23).

[4] for instance $\mapsto_S$ does not depend on the leftmost trace: if $((e, P), [!_t]@T) \mapsto_S ((f, Q), [!_u]@U)$ and $((e, P), [!_{t'}]@T) \mapsto ((f, Q), [!_{u'}]@U)$ then this $\mapsto$ step is also a $\mapsto_S$ step. Else, it would mean $e = \sigma(\_)$ and $T = []$, which contradicts $((e, P), [!_t]@T) \mapsto_S$

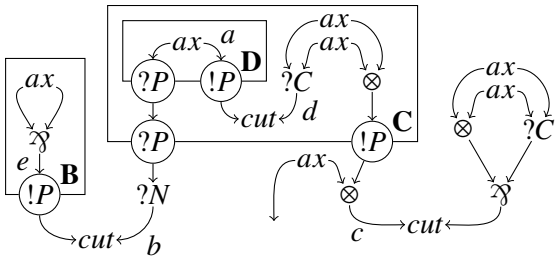


Figure 3.13: Motivation for the definition of $\trianglelefteq$

Definition 55. *We first define a strict order $\triangleleft$ on signatures by induction. For every signature t, t', u, v, $\mathtt{e} \triangleleft t$ and if we suppose $t \triangleleft t'$ then $\mathtt{l}(t) \triangleleft \mathtt{l}(t')$, $\mathtt{r}(t) \triangleleft \mathtt{r}(t')$, $\mathtt{p}(t) \triangleleft \mathtt{p}(t')$, $\mathtt{n}(u,t) \triangleleft \mathtt{n}(v,t')$ and $\mathtt{n}(t,u) \trianglelefteq \mathtt{n}(t',u)$.*

Then we define an order $\trianglelefteq$ on signatures by: $t \trianglelefteq t'$ iff either $t = t'$ or $t \triangleleft t'$.

Lemma 56. *Let $t, u \in Sig$. If $t \blacktriangleleft u$ then $t \trianglelefteq u$.*

Proof. By induction on t. □

Lemma 57. *Let t be a signature, then $\trianglelefteq$ is a total order on $\{u \in Sig \mid u \blacktriangleleft t\}$.*

Proof. Let $u, v \in Sig$ such that $u \blacktriangleleft t$ and $v \blacktriangleleft t$. We prove by induction on t that either $u \trianglelefteq v$ or $v \trianglelefteq u$.

If $t = \mathtt{e}$ then $u = v = \mathtt{e}$ so $u \trianglelefteq v$. If $t = \mathtt{l}(t')$ then either $u = \mathtt{e}$ (in this case $u \trianglelefteq v$), $v = \mathtt{e}$ (in this case $v \trianglelefteq u$) or $u = \mathtt{l}(u')$, $v = \mathtt{l}(v')$, $u' \blacktriangleleft t'$ and $v' \blacktriangleleft t'$ (in this case, by induction hypothesis, either $u' \trianglelefteq v'$ and $u \trianglelefteq v$, or $v' \trianglelefteq u'$ and $v \trianglelefteq u$). The cases $t = \mathtt{r}(t')$ and $t = \mathtt{p}(t')$ are similar.

If $t = \mathtt{n}(t_1, t_2)$, then either $u = \mathtt{e}$ (in this case $u \trianglelefteq v$), $v = \mathtt{e}$ (in this case $v \trianglelefteq u$), or $u = \mathtt{n}(u_1, u_2)$, $v = \mathtt{n}(v_1, v_2)$, $u_1 \blacktriangleleft t_1$, $u_2 \blacktriangleleft t_2$, $v_1 \blacktriangleleft t_1$ and $v_2 \blacktriangleleft t_2$. Then, by induction hypothesis, either $u_2 \trianglelefteq v_2$ or $v_2 \trianglelefteq u_2$.

- If $u_2 \triangleleft v_2$, then $u \triangleleft v$ so $u \trianglelefteq v$.
- If $v_2 \triangleleft u_2$, then $v \triangleleft u$ so $v \trianglelefteq u$.
- If $u_2 = v_2$, then let us observe that, by induction hypothesis, either $u_1 \trianglelefteq v_1$ (in this case $u \trianglelefteq v$) or $v_1 \trianglelefteq u_1$ (in this case $v \trianglelefteq u$).

□

Thus, we can define $Restr_{\rightarrow}((\sigma(B), P), [!_t])$ as the $\rightarrow$-copies of (B, P) which are smaller than t for $\blacktriangleleft$. This set is totally ordered by $\trianglelefteq$ and finite (if t is of size k, it has at most k truncations) so it admits a maximum. This maximum is the *$\rightarrow$-copy restriction* of t for (B, P), written $((\sigma(B), P), [!_t])^{/\rightarrow}$.

Definition 58. *Let $((e, P), [!_t]@T)$ be a context. We define $Restr_{\rightarrow}((e, P), [!_t]@T)$ as the set of signatures u such that $u \blacktriangleleft t$ and $((e, P), [!_u]@T)$ is a $\rightarrow$-copy context. Then, we define $((e, P), [!_t]@T)^{/\rightarrow}$ as the maximum (for the order $\trianglelefteq$) element of $Restr_{\rightarrow}((e, P), [!_t]@T)$.*

For instance in the proof-net of Figure 3.13, we can notice that $Restr_{\rightarrow}((\sigma(B), P), [!_{\mathtt{n}(\mathtt{l}(\mathtt{e}),\mathtt{r}(\mathtt{e}))}])$ (with $\rightarrow$ defined as above by removing the transition $((\sigma(D), [\mathtt{r}(\mathtt{e})]), [!_{\mathtt{l}(\mathtt{e})}]) \mapsto^* ((\overline{d}, [\mathtt{r}(\mathtt{e})]), [!_{\mathtt{l}(\mathtt{e})}])$ from $\mapsto$) is the set $\{\mathtt{n}(\mathtt{l}(\mathtt{e}), \mathtt{e}), \mathtt{n}(\mathtt{e}, \mathtt{r}(\mathtt{e}), \mathtt{n}(\mathtt{e}, \mathtt{e}), \mathtt{e}\}$. Thus, $((\sigma(B), P), [!_{\mathtt{n}(\mathtt{l}(\mathtt{e}),\mathtt{r}(\mathtt{e}))}])^{/\rightarrow} = \mathtt{n}(\mathtt{e}, \mathtt{r}(\mathtt{e}))$.

In Figure 3.13, we can notice that for any set S and $t \in Sig$, $t' = ((\sigma(B), []), [!_t])^{/\mapsto_S}$ if and only if $t' = ((\overline{b}, []), [!_t])^{/\mapsto_S}$. Similarly, $t' = ((\sigma(C), []), [!_t])^{/\mapsto_S}$ iff $t' = ((c, []), [!_t; \otimes_r])^{/\mapsto_S}$. We generalize this observation in Lemma 59.

Lemma 59. *Let us consider $t \in Sig$. Let us suppose that for every $u, v \in Sig$ such that $u \blacktriangleleft t$ and $v \sqsupseteq u$, we have $((e,P),[!_v]@T) \to ((f,Q),[!_v]@U)$. Then, $((e,P),[!_t])^{/\to} = ((f,Q),[!_t])^{/\to}$.*

Proof. Let us consider $u \in Sig$ such that $u \blacktriangleleft t$. For every $v \in Sig$, if $v \sqsupseteq u$, we have $((e,P),[!_v]@T) \mapsto_S ((f,Q),[!_v]@U)$. So $((e,P),[!_v]@T) \mapsto_S^* ((\_,\_),[!_{\mathtt{e}}]@\_)$ if and only if $((f,Q),[!_v]@U) \mapsto_S^* ((\_,\_),[!_{\mathtt{e}}]@\_)$. So, for every signature u, $u \in Restr_\to((e,P),[!_t]@T)$ iff $u \in Restr_\to((f,Q),[!_t]@U)$. In particular, the maximal element (for $\trianglelefteq$) of $Restr_\to((e,P),[!_t]@T)$ is the maximal element of $Restr_\to((f,Q),[!_t]@U)$. So $((e,P),[!_t]@T)^{/\to} = ((f,Q),[!_t]@U)^{/\to}$. □

Lemma 60. *If $((e,P),[!_t]@T)$ is a $\to$-copy context, then $t = ((e,P),[!_t]@T)^{/\to}$.*

Proof. Let us notice that $t \blacktriangleleft t$ and $((e,P),[!_t]@T)$ is a $\to$-copy context so $t \in Restr_\to((e,P),[!_t]@T)$. If we set $t' = ((e,P),[!_t]@T)^{/\to}$, by definition of $((\_,\_),\_)^{/\to}$, $t \trianglelefteq t'$. We also know that $t' \blacktriangleleft t$, so $t' \trianglelefteq t$ (Lemma 56). Finally, $\trianglelefteq$ is an order so $\trianglelefteq$ is antisymmetric, $t = t'$. □

Now, for any potential edge (e,P), we want to define $(e,P)^\to$ as the "biggest" truncation P' of P such that (e,P') is a $\to$-canonical edge. Like for the definition of $((\sigma(B),P),[!_t])^{/\to}$, we need to precise "biggest". For instance, in Figure 3.13, let us define $\to$ as the relation obtained from $\mapsto$ by removing the transition $((\sigma(D),[\mathtt{r}(\mathtt{e})]),[!_{\mathtt{l}(\mathtt{e})}]) \mapsto^* ((\overline{d},[\mathtt{r}(\mathtt{e})]),[!_{\mathtt{l}(\mathtt{e})}])$. The potential edges $(a,[\mathtt{e};\mathtt{l}(\mathtt{e})])$ and $(a,[\mathtt{r}(\mathtt{e});\mathtt{e}])$ are $\to$-canonical but $(a,[\mathtt{r}(\mathtt{e});\mathtt{l}(\mathtt{e})])$ is not $\to$-canonical. Thus, the set of truncations P' of P such that $(e,P')^\to$ is a $\to$-canonical edge does not have a maximum element.

Definition 61. *For every potential edge (e,P), we define $(e,P)^\to$ by induction on $\partial(e)$. If $\partial(e) = 0$, then we set $(e,[])^\to = (e,[])$. Else, $P = Q.t$, let B be the deepest box containing e, $(\sigma(B),Q') = (\sigma(B),Q)^\to$ and $t' = ((\sigma(B),Q'),[!_t])^{/\to}$ then we set $(e,Q.t)^\to = (e,Q'.t')$.*

For example, in the proof-net of Figure 3.13, $(a,[\mathtt{r}(\mathtt{e});\mathtt{l}(\mathtt{e})])^\to = (a,[\mathtt{r}(\mathtt{e});\mathtt{e}])$. This is coherent with our choice in the definition of $((\_,\_),\_)^{/\to}$, because $((\sigma(B),[]),[\parr_r;!_{\mathtt{n}(\mathtt{l}(\mathtt{e}),\mathtt{r}(\mathtt{e}))}]) \rightsquigarrow^5 ((a,[\mathtt{r}(\mathtt{e});\mathtt{l}(\mathtt{e})]),[\parr_r])$, $((\sigma(B),[]),[\parr_r;!_{\mathtt{n}(\mathtt{l}(\mathtt{e}),\mathtt{r}(\mathtt{e}))}])^{/\mapsto_S} = \mathtt{n}(\mathtt{e},\mathtt{r}(\mathtt{e}))$ and $((\sigma(B),[]),[\parr_r;!_{\mathtt{n}(\mathtt{e},\mathtt{r}(\mathtt{e}))}]) \rightsquigarrow^5 ((a,[\mathtt{r}(\mathtt{e});\mathtt{e}]),[\parr_r])$.

Definition 62. *We extend $\blacktriangleleft$ on potentials by $[p_1;\cdots;p_k] \blacktriangleleft [p'_1;\cdots;p'_k]$ iff for $1 \le i \le k$, $p_i \blacktriangleleft p'_i$.*

Lemma 63. *If $(e,P') = (e,P)^\to$ then $P' \blacktriangleleft P$ and $(e,P') \in Can_\to(e)$.*

Proof. We prove the Lemma by induction on $\partial(e)$. If $\partial(e) = 0$, then $P = P' = []$. We can verify that $[] \blacktriangleleft []$, $(e,[]) \in Can_\to(e)$ and $(e,[]) = (e,[])^\to$.

Else, let B be the deepest box containing e. We have $P' = Q'.t'$ with $Q' = (\sigma(B),Q)^\to$ and $t' = ((\sigma(B),Q'),[!_t])^{/\to}$. By induction hypothesis, $Q' \blacktriangleleft Q$ and, by definition of $((\_,\_),\_)^{/\to}$, $t' \blacktriangleleft t$ so $P' \blacktriangleleft P$. By induction hypothesis, $(\sigma(B),Q') \in Can_\to(\vec{\mathrm{E}}_G)$ so $(B,Q') \in Can_\to(B_G)$. Moreover, t is standard so every truncation of t is standard and, by definition of $((\_,\_),\_)^{/\to}$, $t' \in Cop_\to(B,Q')$ so $(e,P') = (e,Q'.t') \in Can_\to(\vec{\mathrm{E}}_G)$. □

Lemma 64. *If $(e,P) \in Can_\to(e)$ then $(e,P)^\to = (e,P)$*

Proof. Let us suppose that $(e,P) \in Can_\to(e)$ and we set $(e,P') = (e,P)^\to$. We prove that $P = P'$ by induction on $\partial(e)$. If $\partial(e) = 0$, then $P = P' = []$. Else, let B be the deepest box containing e, then $P = Q.t$ with $(\sigma(B),Q) \in Can_\to(\sigma(B))$ and $t \in Cop_\to(B,Q)$. By definition of $(\_)^\to$, we have $P' = Q'.t'$ with $(\sigma(B),Q') = (\sigma(B),Q)^\to$ and $t' = ((\sigma(B),Q'),[!_t])^{/\to}$. By induction hypothesis, $Q = Q'$. And by Lemma 60, $t = t'$. Thus, $P = P'$. □

We can notice that, in the same way as the definition of $Can(e)$ only depends on the boxes containing e (cf. page 26), the definition of $(e, P)^{\rightarrow}$ only depends on the boxes containing e. We formalize it with the next lemma.

Lemma 65. *Let us suppose that $e, f \in \vec{E}_G$ are included in the same boxes. If $(e, P)^{\rightarrow} = (e, P')$ then $(f, P)^{\rightarrow} = (f, P')$.*

Proof. If $\partial(e) = 0$, then $(e, P)^{\rightarrow} = (e, [])$ and $(f, P)^{\rightarrow} = (f, [])$. Else, $P = Q.t$. Let B be the deepest box containing e, then B is also the deepest box containing f. Let $(\sigma(B), Q')$ be $(\sigma(B), Q)^{\rightarrow}$ and $t' = ((\sigma(B), Q), [!_t])^{\rightarrow}$. Then $(e, Q.t)^{\rightarrow} = (e, Q'.t')$ and $(f, Q.t)^{\rightarrow} = (f, Q'.t')$. □

In the general case, for any proofnet, let us suppose that $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((e, Q), [!_{\mathtt{e}}])$ and let S be the set of boxes which are entered by their principal door by this path. Then, we will prove that it is enough to know $(e, Q)^{\mapsto_S}$ to trace back the path which arrives to $((e, Q), [!_{\mathtt{e}}])$. To do so, we need to prove that for every intermediary step $(((e_k, P_k), T_k) \rightsquigarrow ((e_{k+1}, P_{k+1}), T_{k+1})$ we have enough information about P_{k+1} and T_{k+1} to determine e_k. This is the role of the following definition.

Definition 66. *For every context $((e, P), [T_n; \cdots; T_1]) \in Cont_G$ we define the context $((e, P), [T_n; \cdots; T_1])^{\rightarrow}$ as $((e, P'), [T'_n; \cdots; T'_1])$ with $(e, P') = (e, P)^{\rightarrow}$ and T'_i is defined by induction on i as follows:*

- *If $T_i = !_t$, and $((e, P'), [!_t; T'_{i+1}; \cdots; T'_1])^{/\rightarrow} = t'$ then $T'_i = !_{t'}$.*
- *If $T_i = ?_t$, and $((\overline{e}, P'), [!_t; {T'_{i+1}}^{\perp}; \cdots; {T'_1}^{\perp}])^{/\rightarrow} = t'$ then $T'_i = ?_{t'}$.*
- *Else, $T'_i = T_i$.*

As an intuition, if $((e, P'), T') = ((e, P), T)^{\rightarrow}$ then $((e, P'), T')$ is the "biggest" $\rightarrow$-canonical context which is a truncation of $((e, P), T)$.

Definition 67. *We also extend the relation $\blacktriangleleft$ on traces by $[T_1; \cdots; T_k] \blacktriangleleft [T'_1; \cdots; T'_k]$ iff for $1 \leq i \leq k$, we are in one of the following cases: ($T_i = !_t$, $T'_i = !_{t'}$ and $t \blacktriangleleft t'$), or ($T_i = ?_t$, $T'_i = ?_{t'}$ and $t \blacktriangleleft t'$) or $T_i = T'_i$.*

Finally we extend $\blacktriangleleft$ on contexts by $((e, P), T) \blacktriangleleft ((e, P'), T')$ iff $P \blacktriangleleft P'$ and $T \blacktriangleleft T'$.

Similarly to Lemmas 63 and 64, the $(\_)^{\rightarrow}$ mapping is idempotent: for every context C, $C^{\rightarrow}$ is $\rightarrow$-canonical. And $C^{\rightarrow} = C$ for every $\rightarrow$-canonical context C.

Lemma 68. *If $((e, P'), T') = ((e, P), T)^{\rightarrow}$ then $((e, P'), T') \blacktriangleleft ((e, P), T)$ and $((e, P'), T')$ is a $\rightarrow$-canonical context.*

Proof. By definition of $((\_, \_), \_)^{\rightarrow}$, $(e, P') = (e, P)^{\rightarrow}$. So, by Lemma 63, $P' \blacktriangleleft P$ and $(e, P') \in Can_{\rightarrow}(e)$. Let $[T_n; \cdots; T_1] = T$ and $[T'_n; \cdots; T'_1] = T'$. For every $1 \leq i \leq n$,

- If $T'_i = !_{u'}$ and $T_i = !_u$ then $u' = ((e, P'), [T_i; \cdots; T_1])^{/\rightarrow}$ so, by definition of $\rightarrow$-copy restrictions, $u' \blacktriangleleft u$ and $((e, P'), [!_{u'}]@[T'_{i+1}; \cdots; T'_1])$ is a $\rightarrow$-copy context.
- If $T'_i = ?_{u'}$ and $T_i = ?_u$ then $u' = ((\overline{e}, P'), [T_i; \cdots; T_1]^{\perp})^{/\rightarrow}$ so, by definition of $\rightarrow$-copy restrictions, $u' \blacktriangleleft u$ and $((\overline{e}, P'), [!_{u'}]@[{T'_{i+1}}; \cdots; T'_1]^{\perp})$ is a $\rightarrow$-copy context.

□

Lemma 69. *If C is a $\rightarrow$-canonical context then $C = C^{\rightarrow}$.*

Proof. Let $((e,P),[T_n;\cdots;T_1]) = C$ and $((e,P'),[T'_n;\cdots;T'_1]) = C^{\rightarrow}$. By definition of $\rightarrow$-canonical context, $(e,P) \in Can_{\rightarrow}(e)$. So, by Lemma 64, $(e,P') = (e,P)^{\rightarrow} = (e,P)$.

Let us prove by induction on i that $T'_i = T_i$. If $T_i = !_u$ then $T'_i = !_{u'}$ with $u' = ((e,P),[!_u;T'_{i-1};\cdots;T'_1])^{/\rightarrow}$. By induction hypothesis, $T'_{i-1} = T_{i-1}$,...,$T'_1 = T_1$ so $u' = ((e,P),[!_u;T_{i-1};\cdots;T_1])^{/\rightarrow}$. By definition of $\rightarrow$-canonical contexts, $((e,P),[!_u;T_{i-1};\cdots;T_1])$ is a $\rightarrow$-copy context. So by Lemma 60, $u' = u$. Thus, $T'_i = T_i$. The case $T_i = ?_u$ is treated similarly. In the other cases, $T'_i = T_i$ by definition of $((\_,\_),\_)^{\rightarrow}$. □

The following theorem is a generalization of Lemma 59 to contexts. For example, in Figure 3.13, we can notice that for every $S \subseteq B_G$ and $t \in Sig$, $((e,[\mathtt{n}(\mathtt{l}(\mathtt{e}),\mathtt{r}(\mathtt{e}))]),[!_t;⅋_r]) \mapsto_S ((a,[\mathtt{r}(\mathtt{e});\mathtt{l}(\mathtt{e})]),[!_t;⅋_r])$. So we can deduce that, if $((e,[\mathtt{n}(\mathtt{l}(\mathtt{e}),\mathtt{r}(\mathtt{e}))]),[!_t;⅋_r])^{\mapsto_S} = ((e,P),[!_u;⅋_r])$ and $((a,[\mathtt{r}(\mathtt{e});\mathtt{l}(\mathtt{e})]),[!_t;⅋_r])^{\mapsto_S} = ((a,Q),[!_v;⅋_r])$ then $u = v$.

Lemma 70. *Let $(e,P),(e,Q)$ be potential edges and U,V be lists of trace elements. Let us suppose that, for every trace element list T, $((e,P),T@U) \rightarrow ((f,Q),T@V)$ and $((\overline{f},Q),T^{\perp}@V^{\perp}) \rightarrow ((\overline{e},P),T^{\perp}@U^{\perp})$. Then, for any trace T, $((e,P),T@U)^{\rightarrow}$ and $((f,Q),T@V)^{\rightarrow}$ are of the shape $(\_,T'@U')$ and $(\_,T'@V')$ with $|T| = |T'|$.*

Proof. Let $[T_k;\cdots;T_1] = T$, $((\_,\_),[T'_k;\cdots;T'_1]@U') = ((e,P),T@U)^{\rightarrow}$ and $((\_,\_),[T''_k;\cdots;T''_1]@V') = ((f,Q),T@V)^{\rightarrow}$. We prove by induction on i that $T'_i = T''_i$.

Let us suppose that $T_i = !_t$. By definition, $T'_i = !_{t'_i}$ with $t'_i = ((e,P),[!_t;T'_{i-1};\cdots;T'_1]@U')^{/\rightarrow}$ and $T''_i = !_{t''_i}$ with $t''_i = ((f,Q),[!_t;T''_{i-1};\cdots;T''_1]@V')^{/\rightarrow}$. By induction hypothesis, we have $[T''_{i-1};\cdots;T''_1] = [T'_{i-1};\cdots;T'_1]$. By supposition, $((e,P),[!_t;T'_{i-1};\cdots;T'_1]@U') \rightarrow^* ((f,Q),[!_t;T'_{i-1};\cdots;T'_1]@V')$. Thus, as in the proof of Lemma 59 $Restr_{\rightarrow}((e,P),[!_t;T'_{i-1};\cdots;T'_1]@U')$ and $Restr_{\rightarrow}((f,Q),[!_t;T'_{i-1};\cdots;T'_1]@V')$ are equal. In particular, the maximum element of those sets is the same, so $T'_i = T''_i$.

Let us suppose that $T_i = ?_t$. By definition, $T'_i = ?_{t'_i}$ with $t'_i = ((\overline{e},P),[!_t;{T'_{i-1}}^{\perp};\cdots;{T'_1}^{\perp}]@{U'}^{\perp})^{/\rightarrow}$ and $T''_i = ?_{t''_i}$ with $t''_i = ((\overline{f},Q),[!_t;{T''_{i-1}}^{\perp};\cdots;{T''_1}^{\perp}]@{V'}^{\perp})^{/\rightarrow}$. By induction hypothesis, we have $[T''_{i-1};\cdots;T''_1] = [T'_{i-1};\cdots;T'_1]$. By supposition, $((\overline{f},Q),[!_t;{T'_{i-1}}^{\perp};\cdots;{T'_1}^{\perp}]@{V'}^{\perp}) \rightarrow^* ((\overline{e},P),[!_t;{T'_{i-1}}^{\perp};\cdots;{T'_1}^{\perp}]@{U'}^{\perp})$. Thus, as in the proof of Lemma 59, we can deduce that the sets $Restr_{\rightarrow}((\overline{e},P),[!_t;{T'_{i-1}}^{\perp};\cdots;{T'_1}^{\perp}]@{U'}^{\perp})$ and $Restr_{\rightarrow}((\overline{f},Q),[!_t;{T'_{i-1}}^{\perp};\cdots;{T'_1}^{\perp}]@{V'}^{\perp})$ are equal. In particular, the maximum element of those sets is the same, so $T'_i = T''_i$. □

For instance in Figure 3.11, for $t,u,v \in Sig$, we have:

$$((\overline{w},[\mathtt{e}]),[!_t])^{\mapsto\{A''\}} = ((\overline{w},[\mathtt{e}]),[!_{\mathtt{e}}]) \qquad ((\overline{w},[\mathtt{n}(x,y)]),[!_t])^{\mapsto\{A''\}} = ((\overline{w},[\mathtt{e}]),[!_{\mathtt{e}}])$$
$$((\overline{w},[\mathtt{r}(x)]),[!_t])^{\mapsto\{A''\}} = ((\overline{w},[\mathtt{r}(\mathtt{e})]),[!_{\mathtt{e}}]) \qquad ((c,[]),[!_t;⅋_r;!_{\mathtt{r}(u)};?_{\mathtt{r}(v)}])^{\mapsto\{A''\}} = ((c,[]),[!_{\mathtt{e}};⅋_r;!_{\mathtt{r}(\mathtt{e})};?_{\mathtt{r}(\mathtt{e})}])$$
$$((\overline{w},[\mathtt{l}(x)]),[!_t])^{\mapsto\{A''\}} = ((\overline{w},[\mathtt{l}(\mathtt{e})]),[!_{\mathtt{e}}]) \qquad ((c,[]),[!_t;⅋_r;!_{\mathtt{r}(u)};?_{\mathtt{l}(v)}])^{\mapsto\{A''\}} = ((c,[]),[!_{\mathtt{e}};⅋_r;!_{\mathtt{r}(\mathtt{e})};?_{\mathtt{l}(\mathtt{e})}])$$
$$((\overline{w},[\mathtt{r}(\mathtt{e})]),[!_t])^{\mapsto\{A'',D\}} = ((\overline{w},[\mathtt{r}(\mathtt{e})]),[!_{\mathtt{e}}]) \qquad ((\overline{w},[\mathtt{r}(\mathtt{n}(u,\mathtt{r}(v)))]),[!_t])^{\mapsto\{A'',D\}} = ((\overline{w},[\mathtt{r}(\mathtt{n}(\mathtt{e},\mathtt{r}(\mathtt{e})))]),[!_{\mathtt{e}}])$$
$$((c,[]),[!_{\mathtt{e}}?_{\mathtt{r}(t)}])^{\mapsto\{A'',D\}} = ((c,[]),[!_{\mathtt{e}}?_{\mathtt{r}(\mathtt{e})}]) \qquad ((c,[]),[!_{\mathtt{r}(\mathtt{e})};?_{\mathtt{r}(\mathtt{n}(t,\mathtt{r}(u)))}])^{\mapsto\{A'',D\}} = ((c,[]),[!_{\mathtt{r}(\mathtt{e})};?_{\mathtt{r}(\mathtt{n}(t,\mathtt{r}(u)))}])$$
$$((\sigma(B'),[t]),[!_u])^{\mapsto\varnothing} = ((\sigma(B'),[\mathtt{e}]),[!_{\mathtt{e}}]) \qquad ((\sigma(B),[t;u]),[!_v])^{\mapsto\varnothing} = ((\sigma(B),[\mathtt{e};\mathtt{e}]),[!_{\mathtt{e}}])$$

3.2.3 Elementary bound for ⇝-stratified proof-nets

The first goal of this subsection is to prove Claim 2. Let us consider a ⇝-stratified proof-ned and a potential box (B,P). We set S as the set of boxes C such that $s_{⇝}(C) < s_{⇝}(B)$ (let us recall that $s_{\_}(\_)$ is defined in Definition 27). If, there exist ⇝-paths $((\sigma(B),P),[!_t]) \rightsquigarrow^k ((e,Q),[!_u])$ and $C' \rightsquigarrow^k ((e,Q'),[!_{u'}])$

with $(e,Q)^{\mapsto_S} = (e,Q')^{\mapsto_S}$, then we have to prove that one can trace back the two paths and they will never separate.

The formalization of Claim 2 is Lemma 75. Most of the technical work will be done in Theorem 72, which we consider to be the technical core of this thesis: most of the work in this thesis drew their inspiration from this theorem and its generalizations (Theorem 104, Lemma 136 and Theorem 145), and we use them in numerous proofs.

Definition 71. *Let G be a $\rightsquigarrow$-stratified proof-net and $n \in \mathbb{N}$, we set $S_n = \{B \in B_G \mid s_{\rightsquigarrow}(B) \leq n\}$. To simplify notations, in Section 3.2.3, we will write $((e,P),T)^{/n}$ for $((e,P),T)^{/\mapsto_{S_n}}$, $(e,Q)^n$ for $(e,Q)^{\mapsto_{S_n}}$, $((e,P),T)^n$ for $((e,P),T)^{\mapsto_{S_n}}$, $Cop_n(B,P)$ for $Cop_{\mapsto_{S_n}}(B,P)$ and $Can_n(B,P)$ for $Can_{\mapsto_{S_n}}(B,P)$.*

Thus, if $s_{\rightsquigarrow}(B) = n$, the set of boxes C such that $B \rightsquigarrow C$ is included in S_{n-1}.

Theorem 72. *For any proof-net G, and $S \subset B_G$. Let C_e, C_f and C'_f be canonical contexts such that $C_e \rightsquigarrow_S C_f$ and $C_f{}^{\mapsto_S} = C'_f{}^{\mapsto_S}$, then there exists a context C'_e such that $C'_e \rightsquigarrow_S C'_f$ and $C_e{}^{\mapsto_S} = C'_e{}^{\mapsto_S}$.*

Proof. We will detail an easy step (crossing a $\parr$ node upward). Most of the other steps are quite similar. For the steps which offer some particular difficulty, we will only detail the points which differ from crossing a $\parr$ upward.

Let us suppose that $C_e = ((e,P),T.\otimes_l) \rightsquigarrow_S ((f,P),T) = C_f$ (crossing a $\parr$ upwards, such that $\overline{f}$ is not a principal edge) and $C_f{}^{\mapsto_S} = C'_f{}^{\mapsto_S}$. So C'_f is of the shape $((f,P'),T')$. We set $C'_e = ((f,P'),T'.\otimes_l)$. Let $((f,P''),T'') = C_f{}^{\mapsto_S}$, then $(f,P)^{\mapsto_S} = (f,P')^{\mapsto_S} = (f,P'')$. So, by Lemma 65, $(e,P)^{\mapsto_S} = (e,P')^{\mapsto_S} = (e,P'')$. Moreover, by Lemma 70, $C_e{}^{\mapsto_S} = ((e,P''),T''.\otimes_l)$ and $C'_e{}^{\mapsto_S} = ((e,P''),T''.\otimes_l)$ so $C_e{}^{\mapsto_S} = C'_e{}^{\mapsto_S}$.

In the case where e is the principal edge of a box B (we consider the case where we cross a *cut*) then $C_e = ((e,P),T.!_t) \rightsquigarrow_S ((f,P),T.!_t) = C_f$. So C'_f is of the shape $((f,P'),T'.!_{t'})$. We set $C'_e = ((e,P'),T'.!_{t'})$. By supposition, $C_f{}^{\mapsto_S} = C'_f{}^{\mapsto_S} = ((f,P''),T''.!_{t''})$. In particular $((f,P''),[!_t])^{/\mapsto_s} = ((f,P''),[!_{t'}])^{/\mapsto_s}$. If $B \in S$, by Lemma 70, we have $C_e{}^{\mapsto_S} = C'_e{}^{\mapsto_S} = ((e,P''),T''.!_{t''})$. Else, we have $C_e{}^{\mapsto_S} = C'_e{}^{\mapsto_S} = ((e,P''),T''.!_{\mathsf{e}})$.

In the case where $\overline{f}$ is the principal edge of a box B (we consider the case where we cross a *cut*) then $C_e = ((e,P),T.?_t) \rightsquigarrow_S ((f,P),T.?_t) = C_f$. So C'_f is of the shape $((f,P'),T'.?_{t'})$. We set $C'_e = ((e,P'),T'.?_{t'})$. By supposition, $C_f{}^{\mapsto_S} = C'_f{}^{\mapsto_S} = ((f,P''),T''.?_{t''})$. By definition of $\rightsquigarrow_S$, B is in S. So, we can notice that $C'_e \rightsquigarrow_S C'_f$ and, using Lemma 70, we have $C_e{}^{\mapsto_S} = C'_e{}^{\mapsto_S} = ((e,P''),T''.?_{t''})$.

Let us suppose that $C_e = ((e,P),T.?_t) \rightsquigarrow_S ((f,P.t),T) = C_f$ (crossing the principal door of C upwards). Then, C'_f must be of the shape $((f,P'.t'),T')$. We set $C'_e = ((e,P'),T'.?_{t'})$. The only particular point is to prove that $((\overline{e},P)^{\mapsto_S},[!_t])^{/\mapsto_S} = ((\overline{e},P')^{\mapsto},[!_{t'}])^{/\mapsto_S}$. By definition, $(f,P.t)^{\mapsto_S} = (f,Q.u)$ with $(\overline{e},P)^{\mapsto_S} = (e,Q)$ and $(\overline{e},Q),[!_t])^{/\mapsto_S} = u$. Similarly, $(f,P'.t')^{\mapsto_S} = (f,Q'.u')$ with $(\overline{e},P')^{\mapsto_S} = (\overline{e},Q')$ and $((\overline{e},Q'),[!_{t'}])^{/\mapsto_S} = u'$. We know that $C_f{}^{\mapsto_S} = C'_f{}^{\mapsto_S}$, so $(f,Q.t)^{\mapsto_S} = (f,Q'.t')^{\mapsto_S}$. Thus $u = u'$, i.e. $((\overline{e},P)^{\mapsto_S},[!_t])^{/\mapsto_S} = ((\overline{e},P')^{\mapsto_S},[!_{t'}])^{/\mapsto_S}$.

Let us suppose that $C_e = ((e,P.t),T) \rightsquigarrow_S ((f,P),T.!_t) = C_f$ (crossing the principal door of B downwards). Then C'_f must be of the shape $((f,P'),T'.!_{t'})$. We set $C'_e = ((e,P'.t'),T')$. The only particular point is to prove that $(e,P.t)^{\mapsto_S} = (e,P'.t')^{\mapsto_S}$. By definition of $(\_,\_)^{\mapsto_S}$, $(e,P.t)^{\mapsto_S} = (e,Q.u)$ with $(f,P)^{\mapsto_S} = (f,Q)$ and $((f,Q),[!_t])^{/\mapsto_S} = u$. Similarly, $(e,P'.t')^{\mapsto_S} = (e,Q'.u')$ with

$(f,P')^{\mapsto s} = (f,Q')$ and $((f,Q'),[!_{t'}])^{/\mapsto s} = u'$. By supposition,

$$\begin{aligned}((f,P),T.!_t)^{\mapsto s} &= ((f,P'),T'.!_{t'})^{\mapsto s}\\ \left((f,P)^{\mapsto s},\_@\left[!_{((f,P)^{\mapsto s},[!_t])^{/\mapsto s}}\right]\right) &= \left((f,P')^{\mapsto s},\_@\left[!_{((f,P')^{\mapsto s},[!'_t])^{/\mapsto s}}\right]\right)\\ ((f,Q),\_@[!_u]) &= ((f,Q'),\_@[!_{u'}])\\ Q.u &= Q'.u'\end{aligned}$$

Let us suppose that $C_e = ((e,P),T.!_t) \rightsquigarrow_S ((f,P.t),T) = C_f$ (crossing an auxiliary door of B upwards). Then, C'_f must be of the shape $((f,P'.t'),T')$. We set $C'_e = ((e,P'),T'.!_{t'})$. The only particular point is to prove that $((e,P)^{\mapsto s},[!_t])^{/\mapsto s} = ((e,P')^{\mapsto s},[!_{t'}])^{/\mapsto s}$. By definition of $(\_,\_)^{\mapsto s}$, $(f,P.t)^{\mapsto s} = (f,Q.u)$ with $(\sigma(B),Q) = (\sigma(B),P)^{\mapsto s}$ and $u = ((\sigma(B),Q),[!_t])^{/\mapsto s}$. Similarly, $(f,P'.t')^{\mapsto s} = (f,Q'.u')$ with $(\sigma(B),Q') = (\sigma(B),P')^{\mapsto s}$ and $u' = ((\sigma(B),Q'),[!_{t'}])^{/\mapsto s}$. We know that $(f,P.t)^{\mapsto s} = (f,P'.t')^{\mapsto s}$, so $u = u'$. By Lemma 70, $((e,P)^{\mapsto s},[!_t])^{/\mapsto s} = ((e,P')^{\mapsto s},[!_{t'}])^{/\mapsto s}$.

Let us suppose that $C_e = ((e,P.t),T) \rightsquigarrow_S ((f,P),T.?_t) = C_f$ (crossing an auxiliary door of C downwards). Then C'_f must of of the shape $((f,P'),T'.?_t)$. We set $C'_e = ((e,P'.t'),T')$. The only particular point is to prove that $(e,P.t)^{\mapsto s} = (e,P'.t')^{\mapsto s}$. By definition of $(\_,\_)^{\mapsto s}$, $(e,P.t)^{\mapsto s} = (e,Q.u)$ with $(\sigma(B),P)^{\mapsto s} = (\sigma(B),Q)$ and $((\sigma(B),Q),[!_t])^{/\mapsto s} = u$. Similarly, $(e,P'.t')^{\mapsto s} = (e,Q'.u')$ with $(\sigma(B),P')^{\mapsto s} = (\sigma(B),Q')$ and $((\sigma(B),Q'),[!_{t'}])^{/\mapsto s} = u'$. By Lemma 65, $(f,P)^{\mapsto s} = (f,Q)$ and $(f,P')^{\mapsto s} = (f,Q')$. By Lemma 70, $((\overline{f},Q),[!_t])^{/\mapsto s} = ((\sigma(B),Q),[!_t])^{/\mapsto s}$ and $((\overline{f},Q'),[!_{t'}])^{/\mapsto s} = ((\sigma(B),Q'),[!_{t'}])^{/\mapsto s}$. By supposition, we have:

$$\begin{aligned}((f,P),T.?_t)^{\mapsto s} &= ((f,P'),T'.?_{t'})^{\mapsto s}\\ \left((f,P)^{\mapsto s},\_@\left[?_{((\overline{f},P)^{\mapsto s},[!_t])^{/\mapsto s}}\right]\right) &= \left((f,P')^{\mapsto s},\_@\left[?_{((\overline{f},P')^{\mapsto s},[!'_t])^{/\mapsto s}}\right]\right)\\ \left((f,P)^{\mapsto s},\_@\left[?_{((\sigma(B),Q),[!_t])^{/\mapsto s}}\right]\right) &= \left((f,P')^{\mapsto s},\_@\left[?_{((\sigma(B),Q'),[!'_t])^{/\mapsto s}}\right]\right)\\ ((f,Q),\_@[?_u]) &= ((f,Q'),\_@[?_{u'}])\\ Q.u &= Q'.u'\end{aligned}$$

Let us suppose $C_e = ((e,P),T.?_t) \mapsto_S ((f,P),T.?_{\mathtt{1}(t)}) = C_f$ (crossing a $?C$ node downwards). Then C'_f must be of the form $((f,P'),T'.?_{u'})$. We first have to prove that u' is of the shape $\mathtt{1}(t')$. We know that $C_f{}^{\mapsto s} = C'_f{}^{\mapsto s}$ so $(f,P)^{\mapsto s} = (f,P')^{\mapsto s}$ (we will write (f,P'') for $(f,P)^{\mapsto s}$) and $((\overline{f},P''),[!_{\mathtt{1}(t)}])^{/\mapsto s} = ((\overline{f},P''),[!_{u'}])^{/\mapsto s}$ (we will write $u'' = ((\overline{f},P''),[!_{\mathtt{1}(t)}])^{/\mapsto s}$). By definition of $((\_,\_),\_)^{/\mapsto s}$, $u'' \blacktriangleleft \mathtt{1}(t)$ so either $u'' = \mathtt{e}$ or $u'' = \mathtt{1}(t'')$. In the first case, let us notice that $u'' \vartriangleleft \mathtt{1}(\mathtt{e}) \blacktriangleleft \mathtt{1}(t)$ and $((\overline{f},P''),[!_{\mathtt{1}(\mathtt{e})}])$, which would contradict the definition of $((\overline{f},P''),[!_{\mathtt{1}(t)}])^{/\mapsto s}$. So $u'' = \mathtt{1}(t'')$ and, because $u'' \blacktriangleleft u'$, u' is of the shape $\mathtt{1}(t')$.

We set $C'_e = ((e,P'),T'.?_{t'})$. The only particular point to prove is the fact that $((\overline{e},P''),[!_t])^{/\mapsto s} = ((\overline{e},P''),[!_{t'}])^{/\mapsto s}$. Let us notice that for any signature ν, we have $((\overline{f},P''),[!_{\mathtt{1}(\nu)}]) \mapsto_S ((\overline{e},P''),[!_\nu])$ so $Restr_{\mapsto_S}((\overline{f},P''),[!_{\mathtt{1}(t)}]) = \{\mathtt{e}\} \cup \{\mathtt{1}(\nu) \mid \nu \in Restr_{\mapsto_S}((\overline{e},P''),[!_t])\}$. In particular, $\mathtt{1}(t'') = ((\overline{f},P''),[!_t])^{/\mapsto s} = \mathtt{1}(((\overline{e},P''),[!_t])^{/\mapsto s})$. Similarly $\mathtt{1}(t'') = ((\overline{f},P''),[!_{t'}])^{/\mapsto s} = \mathtt{1}(((\overline{e},P''),[!_{t'}])^{/\mapsto s})$. So we can deduce that $((\overline{e},P''),[!_t])^{/\mapsto s} = ((\overline{e},P''),[!_{t'}])^{/\mapsto s} = t''$.

Let us suppose $C_e = ((e,P),T.?_{t_1}.?_{t_2}) \mapsto_S ((f,P),T.?_{\mathtt{n}(t_1,t_2)}) = C_f$ (crossing a $?N$ node downwards). Then, C'_f must be of the shape $((f,P'),T'.?_{u'}])$. We first have to prove that u' is of the shape $\mathtt{n}(t'_1,t'_2)$. We know that $C_f{}^{\mapsto s} = C'_f{}^{\mapsto s}$ so $(f,P)^{\mapsto s} = (f,P')^{\mapsto s}$ (we will write (f,P'') for $(f,P)^{\mapsto s}$) and $((\overline{f},P''),[!_{\mathtt{n}(t_1,t_2)}])^{/\mapsto s} = ((\overline{f},P''),[!_{u'}])^{/\mapsto s}$ (we will write

$u'' = ((\overline{f}, P''), [!_{\mathtt{n}(t_1,t_2)}])^{/\mapsto_S}$). By definition of $((\_,\_),\_)^{/\mapsto_S}$, $u'' \blacktriangleleft \mathtt{n}(t_1,t_2)$ so either $u'' = \mathtt{e}$ or $u'' = \mathtt{n}(t_1'', t_2'')$. In the first case, let us notice that $u'' \lhd \mathtt{n}(\mathtt{e},\mathtt{e}) \blacktriangleleft u'$ and $((\overline{f}, P''), [!_{\mathtt{n}(\mathtt{e},\mathtt{e})}])$ is a $\mapsto_S$-copy context, which would contradict the definition of $((\overline{f}, P''), [!_{\mathtt{n}(t_1,t_2)}])^{/\mapsto_S}$. So $u'' = \mathtt{n}(t_1'', t_2'')$ and, because $u'' \blacktriangleleft u'$, u' is of the shape $\mathtt{n}(t_1', t_2')$.

We set $C'_e = ((e, P'), T'.?_{t_1'}.?_{t_2'})$. The only particular points to prove are that $((\overline{e}, P''), [!_{t_2}])^{/\mapsto_S} = ((\overline{e}, P''), [!_{t_2'}])^{/\mapsto_S}$ and (if we set $t_2'' = ((\overline{e}, P''), [!_{t_2}])^{/\mapsto_S}$), $((\overline{e}, P''), [!_{t_1}; !_{t_2''}])^{/\mapsto_S} = ((\overline{e}, P''), [!_{t_1'}; !_{t_2''}])^{/\mapsto_S}$. Let us set $t_1'' = ((\overline{e}, P''), [!_{t_1}; !_{t_2''}])^{/\mapsto_S}$.

Let us observe that $\mathtt{n}(t_1'', t_2'') \in \mathit{Restr}_{\mapsto_S}(((\overline{f}, P''), [!_{\mathtt{n}(t_1,t_2)}]))$ and for every $u_2'' \sqsupseteq t_2''$, $\mathtt{p}(u_2'') \sqsupseteq \mathtt{n}(t_1'', t_2'')$. By definition of $\mathit{Restr}_{\mapsto_S}(\_)$, $((\overline{f}, P''), [!_{\mathtt{p}(u_2'')}]) \mapsto_S^* ((\_,\_), [!_{\mathtt{e}}]@\_)$. So for every $u_2'' \sqsupseteq t_2''$, $((\overline{e}, P''), [!_{u_2''}]) \mapsto_S ((\_,\_), [!_{\mathtt{e}}]@\_)$, moreover $t_2'' \blacktriangleleft t_2$ (because $\mathtt{n}(t_1'', t_2'') \blacktriangleleft \mathtt{n}(t_1, t_2)$). By Definition of $\mathit{Restr}_{\mapsto_S}(\_)$, t_2'' belongs to $\mathit{Restr}_{\mapsto_S}(((\overline{e}, P''), [!_{t_2}]))$. By definition of $((\_,\_),\_)^{/\mapsto_S}$, $t_2'' \trianglelefteq ((\overline{e}, P''), [!_{t_2}])^{/\mapsto_S}$. Let $v_2 = ((\overline{e}, P''), [!_{t_2}])^{/\mapsto_S}$, if $t_2'' \lhd v_2$, then $\mathtt{n}(t_1'', t_2'') \lhd \mathtt{n}(\mathtt{e}, v_2)$ and $\mathtt{n}(\mathtt{e}, v_2) \in \mathit{Restr}_{\mapsto_S}(((\overline{f}, P''), [!_{\mathtt{n}(t_1,t_2)}]))$ so $\mathtt{n}(t_1'', t_2'')$ is not the maximal element of $\mathit{Restr}_{\mapsto_S}(((\overline{f}, P''), [!_{\mathtt{n}(t_1,t_2)}]))$ for $\lhd$, which contradicts the definition of $\mathtt{n}(t_1'', t_2'')$. So $t_2'' \trianglelefteq v_2$ and $\neg(t_2'' \lhd v_2)$, which means that $v_2 = t_2''$. In other words: $((\overline{e}, P''), [!_{t_2}])^{/\mapsto_S} = t_2''$. We prove similarly that $((\overline{e}, P''), [!_{t_2'}])^{/\mapsto_S} = t_2''$.

For every $u_1'' \sqsupseteq t_1''$, $\mathtt{n}(u_1'', t_2'') \sqsupseteq \mathtt{n}(t_1'', t_2'')$. By definition of $\mathit{Restr}_{\mapsto_S}(\_)$, there exists a path of the shape $((\overline{f}, P''), [!_{\mathtt{n}(u_1'', t_2'')}]) \mapsto_S^* ((\_,\_), [!_{\mathtt{e}}]@\_)$. So for every $u_1'' \sqsupseteq t_1''$, $((\overline{e}, P''), [!_{u_1''}; !_{t_2''}]) \mapsto_S ((\_,\_), [!_{\mathtt{e}}]@\_)$, moreover $t_1'' \blacktriangleleft t_1$, so $t_1'' \in \mathit{Restr}_{\mapsto_S}(((\overline{e}, P''), [!_{t_1}; !_{t_2''}]))$. By definition of $((\_,\_),\_)^{/\mapsto_S}$, $t_1'' \trianglelefteq ((\overline{e}, P''), [!_{t_1}; !_{t_2''}])^{/\mapsto_S}$. Let $v_1 = ((\overline{e}, P''), [!_{t_1}; !_{t_2''}])^{/\mapsto_S}$, if $t_1'' \lhd v_1$ then $\mathtt{n}(t_1'', t_2'') \lhd \mathtt{n}(v_1, t_2'')$ and $\mathtt{n}(v_1, t_2'') \in \mathit{Restr}_{\mapsto_S}(((\overline{f}, P''), [!_{\mathtt{n}(t_1,t_2)}]))$ so $\mathtt{n}(t_1'', t_2'')$ is not the maximal element of $\mathit{Restr}_{\mapsto_S}(((\overline{f}, P''), [!_{\mathtt{n}(t_1,t_2)}]))$ for $\lhd$, which contradicts the definition of $\mathtt{n}(t_1'', t_2'')$. So $t_1'' \trianglelefteq v_1$ and $\neg(t_1'' \lhd v_1)$, which means that $t_1'' = v_1$. In other words: $((\overline{e}, P''), [!_{t_1}; !_{t_2''}])^{/\mapsto_S} = t_1''$. We prove similarly that $((\overline{e}, P''), [!_{t_1'}; !_{t_2''}])^{/\mapsto_S} = t_1''$. □

Theorem 72 allows us to trace back some $\rightsquigarrow$ paths provided that we have some information about the last context of the path. In this subsection, we will show how this implies that $\dot{\rightsquigarrow}$ satisfies the elementary stratification property (Lemma 75). But, first, we need a technical lemma.

Lemma 73. *Let us consider $\rightarrow \subseteq \mapsto$. Let us suppose that $((\sigma(B), P), [!_t]) \rightarrow^* C$, then there exists a unique context of the shape $((\sigma(B'), P'), [!_{t'}])$ such that $((\sigma(B), P), [!_t]) \rightarrow^* ((\sigma(B'), P'), [!_{t'}])(\rightsquigarrow \cap \rightarrow)^* C$*

Proof. First we prove the existence of such a context. Let us consider the $\hookrightarrow$ steps in the $\rightarrow$-path from $((\sigma(B), P), [!_t])$ to C. If there is no such step, then $((\sigma(B), P), [!_t]) \rightarrow^0 ((\sigma(B), P), [!_t])(\rightsquigarrow \cap \rightarrow)^* C$. Else we consider the last such step $((\sigma_i(B'), P'), [!_{t'}]) \hookrightarrow ((\sigma(B'), P'), [!_{t'}])$, we can observe that we have $((\sigma(B), P), [!_t]) \rightarrow^* ((\sigma(B'), P'), [!_{t'}])(\rightsquigarrow \cap \rightarrow)^* C$.

Then, we prove the unicity of such a context. If $((\sigma(B_1), P_1), [!_{t_1}]) \rightsquigarrow^* C$ and $((\sigma(B_2), P_2), [!_{t_2}]) \rightsquigarrow^* C$ then, because $\rightsquigarrow$ is injective, either $((\sigma(B_1), P_1), [!_{t_1}]) \rightsquigarrow^* ((\sigma(B_2), P_2), [!_{t_2}])$ or $((\sigma(B_2), P_2), [!_{t_2}]) \rightsquigarrow^* ((\sigma(B_1), P_1), [!_{t_1}])$. However, for any context of the shape $((\sigma(B'), P'), [!_{t'}])$, there is no context C' such that $C' \rightsquigarrow ((\sigma(B'), P'), [!_{t'}])$ so $((\sigma(B_2), P_2), [!_{t_2}]) = ((\sigma(B_1), P_1), [!_{t_1}])$. □

Lemma 74. *Let us consider $n \in \mathbb{N}$. If $((\sigma(B), P), [!_t]) \mapsto_{S_n} C_k \cdots \mapsto_{S_n} C_0$ and ${C_0}^{n-1} = {C'_0}^{n-1}$ then there exists $(C'_i)_{0 \le i \le k}$ such that $C'_k \mapsto_{S_n} \cdots \mapsto_{S_n} C'_0$ and, for $0 \le i \le k$, ${C_i}^{n-1} = {C'_i}^{n-1}$.*

Proof. We will prove (by induction on i) the existence of a context C'_i such that $C'_i \mapsto_{S_n} C'_{i-1}$ and ${C_i}^{n-1} = {C'_i}^{n-1}$. If $i = 0$, we can verify that C'_0 (given by assumption) satisfies the property. Else, by induction hypothesis we know that there exists a context C'_{i-1} such that $(C_{i-1})^{n-1} = (C'_{i-1})^{n-1}$.

If the $C_i \mapsto_S C_{i-1}$ step is a $\hookrightarrow$ step, it is of the shape $C_i = ((\overline{\sigma_j(D)}, Q), [!_u]) \hookrightarrow ((\sigma(D), Q), [!_u]) = C_{i-1}$. So C'_{i-1} is of the shape $((\sigma(D), Q'), [!_{u'}])$ with $\sigma(D), Q^{n-1} = (\sigma(D), Q')^{n-1} = (\sigma(D), Q'')$ and

$((\sigma(D), Q''), [!_u])^{/n-1} = ((\sigma(D), Q''), [!_{u'}])^{/n-1} = u''$. Let us set $C'_i = ((\overline{\sigma_j(D)}, Q'), [!_{u'}])$. By Lemma 65, $(\sigma_j(D), Q)^{n-1} = (\sigma_j(D), Q')^{n-1} = (\sigma_j(D), Q'')$. If $D \in S$, by Lemma 59, we have $((\overline{\sigma_j(D)}, Q''), [!_u])^{/\mapsto_S} = ((\overline{\sigma_j(D)}, Q''), [!_{u'}])^{/\mapsto_S} = u''$ (and in this case $C_i{}^{\mapsto_S} = C'_i{}^{\mapsto_S} = ((\overline{\sigma_j(D)}, Q''), [!_{u''}])$). Else, $D \notin S$ so we have $C_i{}^{\mapsto_S} = C'_i{}^{\mapsto_S} = ((\overline{\sigma_j(D)}, Q''), [!_{\mathtt{e}}])$.

If the $C_i \mapsto_{S_n} C_{i-1}$ step is a $\hookrightarrow$ step then, by Lemma 73, there exists a context of the shape $((\sigma(D), Q), [!_u])$ such that $((\sigma(B), P), [!_t]) \mapsto_{S_n} ((\sigma(D), Q), [!_u])(\rightsquigarrow \cap \mapsto_{S_n})^* C_{i-1}$. By definition of $\mapsto_{S_n}$, we have $D \in S_n$. So, if C_{i-1} is of the shape $((\overline{\sigma(D_i)}, Q_i), [!_v])$, then we have $D \rightsquigarrow_{S} D_i$ so $s_{\rightsquigarrow_S}(D) < s_{\rightsquigarrow_S}(D_i) \leq n$, which means that $D_i \in S_{n-1}$. Thus, we have $C_i \rightsquigarrow_{S_{n-1}} C_{i-1}$. By Theorem 72, there exists a context C'_i such that $C'_i \rightsquigarrow_{S_{n-1}} C'_{i-1}$ and $C_i{}^{n-1} = C'_i{}^{n-1}$. □

Lemma 75 (strong acyclicity)**.** *Let G be a normalizing proof-net. For every $n \in \mathbb{N}$, if $((\sigma(B), P), [!_t]) \mapsto^*_{S_n} ((e, Q), [!_u]) \mapsto^+_{S_n} ((e, Q'), [!_v])$ then $(e, Q)^{n-1} \neq (e, Q')^{n-1}$.*

Proof. We will prove it by contradiction. Let us suppose that we have $((\sigma(B), P), [!_t]) \mapsto^l_{S_n} ((e, Q), [!_u])$ and $((\sigma(B), P), [!_t]) \mapsto^{l+m}_{S_n} ((e, Q'), [!_{u'}]) = D'$, and $(e, Q)^{n-1} = (e, Q')^{n-1}$. Then, $((e, Q), [!_{u'}])^{n-1} = D'^{n-1}$. By Lemma 74, there exists a context C'_1 such that $C'_1 \mapsto^{l+m} ((e, Q), [!_{u'}])$ and $C'_1{}^{n-1} = ((\sigma(B), P), [!_t])^{n-1}$. So C'_1 is of the shape $((\sigma(B), P_1), [!_{t'_1}])$. By Lemma 53, there exists a signature t_1 such that $((\sigma(B), P_1), [!_{t_1}]) \mapsto^{l+m} ((e, Q), [!_u])$ so $((\sigma(B), P_1), [!_{t_1}]) \mapsto^{l+2m} ((e, Q'), [!_{u'}])$.

We define C_1 as the context $((\sigma(B), P_1), [!_{t_1}])$. For $k \in \mathbb{N}$, we can define by induction on k a context $C_k = ((\sigma(B), P_k), [!_{t_k}])$ such that $C_k \mapsto^{l+k\cdot m}_{S_n} D$ and $C_k \mapsto^{l+(k+1)\cdot m}_{S_n} D'$.

Thus, if $m > 0$, we define an infinite path. In particular, this path will go through infinitely many contexts of shape $((\sigma(B), P'), [!_{t'}])$. According to Theorem 20, the number of canonical potentials for an edge is finite. So there is some $(\sigma(B), P') \in Can(\vec{\mathrm{E}}_G)$ and $v, v' \in Sig$ such that $((\sigma(B), P'), [!_v]) \mapsto^+ ((\sigma(B), P'), [!_{v'}])$. This is impossible as we proved proof-nets to be acyclic (Lemma 21). This is a contradiction, so our hypothesis is wrong, $m = 0$. There is no path of the shape $((\sigma(B), P), [!_t]) \mapsto^*_{S_n} ((e, Q), [!_u]) \mapsto^+_{S_n} ((e, Q'), [!_v])$ with $(e, Q)^{n-1} = (e, Q')^{n-1}$. □

Lemma 76. *Let us consider a relation on contexts $\rightarrow \subseteq \mapsto$, and a potential box (B, P). Let us suppose that $|\{(e, Q) \mid \exists t, u \in Sig, ((\sigma(B), P), [!_t]) \rightarrow ((e, Q), [!_u])\}| \leq M$. Then, $Cop_{\rightarrow}(B, P) \leq 2^{2^{2\cdot M}}$.*

Proof. Let us consider $u \in Sig$ such that there exists $t \in Cop_{\rightarrow}(B, P)$ such that $t \sqsubseteq u$. By definition of $Cop_{\_}(\_)$ (Definition 50, in page 61), there exists a path of the shape $((\sigma(B), P), [!_t]) \rightarrow^* ((\_, \_), [!_{\mathtt{e}}])$. We will consider u as a tree. During the path beginning by $((\sigma(B), P), [!_u])$, the height of the left-most branch of u (viewed as a tree) decreases to 0 (the height of $\mathtt{e}$). The height of the left-most branch decreases only by crossing a $?C$ or $?N$ nodes upwards (which corresponds to contexts of the shape $((e, Q), [!_v])$) and during those steps it decreases by exactly 1. So the height of the left-most branch of u is inferior to the number of contexts of the shape $((e, Q), [!_v])$ through which the path goes. From the strong acyclicity lemma (Lemma 75), we can deduce that the height of the left-most branch is inferior to M.

Let t be a $\rightarrow$-copy of (B, P), then the height of t is the height of its deepest branch. Once we consider signatures as trees, a simplification u of t can be viewed as a subtree of t obtained as follows: we choose a branch of t and u is the part of t on the right of this branch, in particular this branch becomes the leftmost branch of u. So there exists a simplification u of t such that the leftmost branch of u is the deepest branch of t. So the heigth of t is equal to the heigth of the leftmost branch of u. By the preceding paragraph, the height of u is at most M so the height of t is at most M. The result is obtained by Lemma 31. □

Lemma 77. *For every $x \in \mathbb{R}$, if $x \geq 1$ then $2^x \geq x + 1$. If $x \geq 2$, then $2^x \geq 2 \cdot x$. And, if $x \geq 4$, then $2^x \geq 4 \cdot x$.*

Proof. For each of this inequality, we first check that the inequality is true for the minimal value of x. Indeed, $2^1 = 2 \geq 2 = 1 + 1$, $2^1 = 2 \geq 2 = 2 \cdot 2$, and $2^4 = 16 \geq 16 = 4 \cdot 4$. Then, we check that the derivative of the left part is higher than the derivative of the right part: if $x \geq 1$ then $\log(2) \cdot 2^x \geq \log(2) \cdot 2 \geq 1$, if $x \geq 2$ then $\log(2) \cdot 2^x \geq \log(2) \cdot 4 \geq 2$ and, if $x \geq 4$ then $\log(2) \cdot 2^x \geq \log(2) \cdot 16 \geq 4$. □

Theorem 78. *If a proof-net G normalizes and is $\rightsquigarrow$-stratified, then the length of its longest path of reduction is bounded by $2^{|\vec{E}_G|}_{3|\rightsquigarrow|}$*

Proof. By Lemmas 75 and 76, we have:

$$\max_{(B,P)\in Pot(B_G)} |Cop_n(B,P)| \leq 2^{2^{2\cdot|Can_{n-1}(\vec{E}_G)|}}$$

$$\max_{(e,P)\in Pot(\vec{E}_G)} |Can_n(e)| \leq \left(2^{2^{2\cdot|Can_{n-1}(\vec{E}_G)|}}\right)^{\partial_G}$$

$$\left|Can_n(Pot(\vec{E}_G))\right| \leq \left|\vec{E}_G\right| \left(2^{\partial_G \cdot 2^{2\cdot|Can_{n-1}(\vec{E}_G)|}}\right)$$

We define u_n as $2^{\left|\vec{E}_G\right|}_{3\cdot n}$. We will show by induction that, for every $n \in \mathbb{N}$, $\left|Can_n(Pot(\vec{E}_G))\right| \leq u_n$. For $n = 0$, we can notice that for every $e \in \vec{E}_G$, we have $|Can_0(e)| = 1$ (the only canonical potentials are lists of e) so $\left|Can_0(Pot(\vec{E}_G))\right| \leq \left|\vec{E}_G\right| \leq u_0$. If $n \geq 0$, let us notice that G has at least two edges so $\left|\vec{E}_G\right| \geq 4$. We have the following inequalities (to simplify the equations, we will write s for $\left|\vec{E}_G\right|$):

$$\left|Can_{n+1}(Pot(\vec{E}_G))\right| \leq s\left(2^{\partial_G \cdot 2^{2\cdot|Can_n(\vec{E}_G)|}}\right) \leq s\left(2^{\partial_G \cdot 2^{2\cdot u_n}}\right) \leq 2^{\frac{s}{2}}\left(2^{s\cdot 2^{2\cdot u_n}}\right)$$

$$\log\left(\left|Can_{n+1}(Pot(\vec{E}_G))\right|\right) \leq \frac{s}{2} + s \cdot 2^{2\cdot u_n} \leq (2\cdot s)\cdot 2^{2\cdot u_n} \leq 2^{s+2\cdot u_n} \leq 2^{4u_n} \leq 2^{2^{u_n}}$$

$$\left|Can_{n+1}(Pot(\vec{E}_G))\right| \leq 2^{u_n}_3 = 2^s_{3n+3} = u_{n+1}$$

Then, corollary 20 gives us the announced bound. □

Let us notice that $|\rightsquigarrow| \leq |B_G|$ and (as we argued in Section 3.1.4, page 46) it is reasonable to assume that the number of boxes does not depend on the argument of the function. So, when we will consider subsystems of LL enforcing $\rightsquigarrow$-stratification, for every typed proof-net G there exists an elementary function e_G such that for every argument H in normal form, $(G)H$ normalizes in $\leq e_G(|E_H|)$ steps. So, if a subsystem LL enforces $\rightsquigarrow$-stratification, then this subsystem is elementary time sound.

Theorem 79. *Let us suppose that there exists a box-bounded[5] subsystem[6] S such that every proof-net of G_S is $\rightsquigarrow$-stratified, then S is sound for $Elem = \left\{x \mapsto 2^x_i \;\middle|\; i \in \mathbb{N}\right\}$.*

Proof. Let us consider a proof-net G whose only conclusion's label is of the shape $A \multimap B$ with $A \in \underline{B_S}$. By definition of LL subsystems, there exist $a, b \in \mathbb{N}$ such that for every binary list l, and $H \in Enc_A(l)$, we have $\left|\vec{E}_H\right| \leq a \cdot |l| + b$. There exists $k \in \mathbb{N}$ such that, for every $x \geq 0$, $ax + b + 3 + \left|\vec{E}_G\right| \leq 2^x_k$. By definition of box-boundedness, there exists $n_A \in \mathbb{N}$ such that for every binary list l and $H \in Enc_A(l)$, there is at most n_A boxes in H. We define n as $n_A + |B_G|$.

[5] cf. Definition 37 in Section 3.1.4

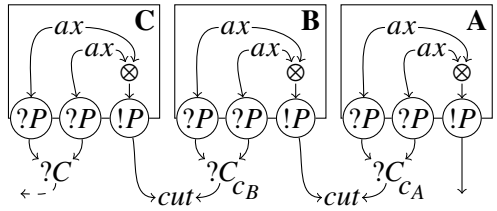
[6] cf. Definition 35 in Section 3.1.4

For every binary list l and $H \in Enc_A(l)$, $\left|\vec{E}_H\right| \leq a \cdot |l| + b$ so $\left|\vec{E}_{(G)H}\right| \leq a \cdot |l| + b + \left|\vec{E}_G\right| + 3$. We can also notice that $\left|B_{(G)H}\right| \leq |B_H| + |B_G| \leq n_A + |B_G| = n$.

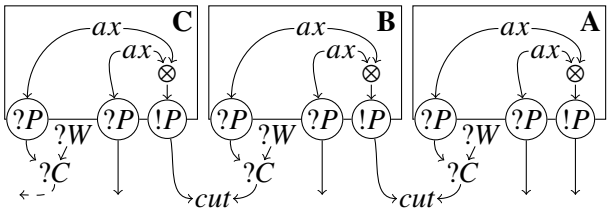
Moreover, $(G)H$ is $\rightsquigarrow\!\!\!\!\!\rightarrow$-stratified, so $S = |\ \rightsquigarrow\!\!\!\!\!\rightarrow\ |$ is lower than $\left|B_{(G)H}\right|$ so is lower than $n - k$. Thus, by Theorem 78,

$$W_{(G)H} \leq 2_{3\cdot n}^{\left|\vec{E}_{(G)H}\right|} \leq 2_{3\cdot n+k}^{|l|}$$

Let us notice that $x \mapsto 2_{3\cdot n+k}^{x} \in \mathit{Elem}$. Thus, by definition, S is sound for Elem. □



(a) This proof-net (if extended to n boxes) reduces in $O(2^n)$ reduction steps



(b) This proof-net (if extended to n boxes) reduces in $O(n)$ reduction steps.

Figure 3.14: Motivation for the definition of $\succ$ for the dependence control condition.

3.3 Polynomial time

Though $\rightsquigarrow$-stratification gives us a bound on the length of the reduction, elementary time is not considered as a reasonable bound. As stated in Section 3.1.6, the complexity class we are interested in is polynomial time. In this section, we will refine the analysis of Section 3.1.6 in order to define a more expressive characterization of *Poly*.

Figure 3.14a shows us a way for the complexity to arise, despite stratification. On this proof-net, node c_A duplicates the box B and creates contractions c_A^l and c_A^r above the premises of c_B. Then, c_B, c_A^l and c_A^r duplicate C (so the box C has 4 residues) and create contractions c_B^l, c_B^r, c_A^{ll}, c_A^{lr}, c_A^{rl} and c_A^{rr} above the premises of c_C,... In [63], this situation is called a chain of "spindles". We call "dependence control condition" any restriction on linear logic which aims to tackle this kind of spindle chains. The solution chosen by Girard [38] (presented in Section 3.1.6) was to limit the number of auxiliary doors of each $!P$-box to 1. To keep some expressivity, he introduced a new modality § with §-boxes which can have an arbitrary number of auxiliary doors.

Baillot and Mazza generalized *ELL* with L^3, a system capturing elementary time [8]. Contrary to *ELL*, L^3 allows dereliction and digging ($?D$ and $?N$ nodes). The presence of digging allows another way to create an exponential blow up, shown in Figure 3.17. Notice that in this proof-net, all the boxes have at most one auxiliary door. So, contrary to the case of *ELL* where the "one auxiliary door" condition alone ensures polynomial time, Baillot and Mazza added another restriction. They defined the L^4 proof-nets as the L^3 proof-nets without $?N$ node and with at most one auxiliary door by box. We call "nesting condition" any restriction on linear logic which aims to prevent exponential blow-ups by chains of the type of Figure 3.17. The nesting condition of *LLL* and L^4 is "no $?N$ node". L^4 proof-nets normalize in polynomial time. However, we think that having all the node labels of linear logic (with some restriction on them) in L^3 was a nice feature and it is unfortunate that the authors could not keep the digging in L^4. The nesting condition defined in Section 3.3.2 (together with our stratification and dependence control condition) enforces polynomial time normalization without forbidding the digging. In Section 5 we define *SNLL*, a subsystem of Linear Logic characterizing *Poly* which includes digging (in a constrained way).

3.3.1 Dependence control

The "one auxiliary door"condition forbids a great number of proof-nets containing boxes with more than one auxiliary door but whose complexity is still polynomial. The complexity explosion in Figure 3.14a comes from the fact that two copies of a box B fuse with the same box A. A box with several auxiliary doors is only harmful if two of its auxiliary edges are contracted as in Figure 3.14a. For instance, the proof-net of Figure 3.14b normalizes in linear time. The copies of C depend on the copies of B which depend on the

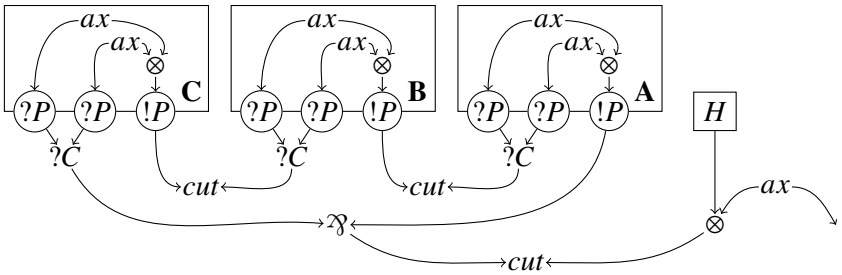


Figure 3.15: If H is in normal form, this proof-net reduces in exactly 32 *cut*-elimination steps

copies of A: $Cop(C,[]) = \{\mathtt{r(e)}\} \cup \{\mathtt{l}(t) \mid t \in Cop(B,[])\}$ and $Cop(B,[]) = \{\mathtt{r(e)}\} \cup \{\mathtt{l}(t) \mid t \in Cop(A,[])\}$. But the dependences are additives: $|Cop(C,[])| \leq 1 + |Cop(B,[])|$ and $|Cop(B,[])| \leq 1 + |Cop(A,[])|$. Contrary to the dependences in Figure 3.14a which are affine: $|Cop(C,[])| \leq 2 \cdot |Cop(B,[])|$ and $|Cop(B,[])| \leq 2 \cdot |Cop(A,[])|$.

Moreover, as we insisted on in Section 3.1.4, we are interested in the complexity of functions, not stand-alone proof-nets. We can create a proof-net which has Figure 3.14a as a subproof-net and whose complexity is polynomial (see Definition 36 for a formalization of complexity). In fact, as Figure 3.15 shows, such a proof-net can even have a constant time complexity.

As noticed in Section 3.1.6, the exponential happens when the length of the chain of affine dependence depends on the input. Here it means that the exponential blow up happens when the length of a chain of spindles depends on the input, as in Figure 3.16. If we replace the sub proof-net $H = Enc_N(3)$ (which represents 3) by a proof-net $Enc_N(n)$ representing n, the resulting proof-net (i.e. $(G)Enc_N(n)$) normalizes in more than 2^n steps of $\rightarrow_{cut}$.

That is the reason why, in the system L^{3a} [28], Dorman and Mazza replaced the "one auxiliary door" condition by a looser dependence control condition:

- Each edge is labelled with an integer, the label of an auxiliary edge must be greater or equal to the label of the principal edge of the box.
- For a given box at most one auxiliary edge can have the same label as the principal edge.

Thus, if one tries to type the proof-net of Figure 3.14a in L^{3a}, either $\sigma_0(A)$ or $\sigma_1(A)$ has a label strictly greater than the label of $\sigma(A)$. They are contracted so they must have the same label. So both auxiliary edges have a label strictly greater than the label of $\sigma(A)$. The label of $\sigma(B)$ is equal to the label of those auxiliary edges. Thus the label of $\sigma(A)$ is inferior to the label of $\sigma(B)$ which is inferior to the label of $\sigma(C)$. In general, the length of chains of spindles is bounded by the maximum label of the proof-net, which does not depend on the input. The dependence control of L^{3a} seems to give a greater expressive power than the dependence control of *LLL*. In our view, the main limitation of L^{3a} is that it uses the same labels to control dependence and to enforce stratification. This entails useless constraints on the strata corresponding to the auxiliary edges of boxes.

Our dependence control condition is closer to *MS*: in [64], Roversi and Vercelli proposed to generalize the "one auxiliary door" condition by considering a framework of logics: *MS*. *MS* is defined as a set of subsystems of *ELL* with indexes on ! and ? connectives. Roversi and Vercelli provide a characterization of the *MS* systems which are sound for *Poly*. This criterion says that a *MS* system is sound for *Poly* if and only if for every $k \in \mathbb{N}$, one of the two following condition holds:

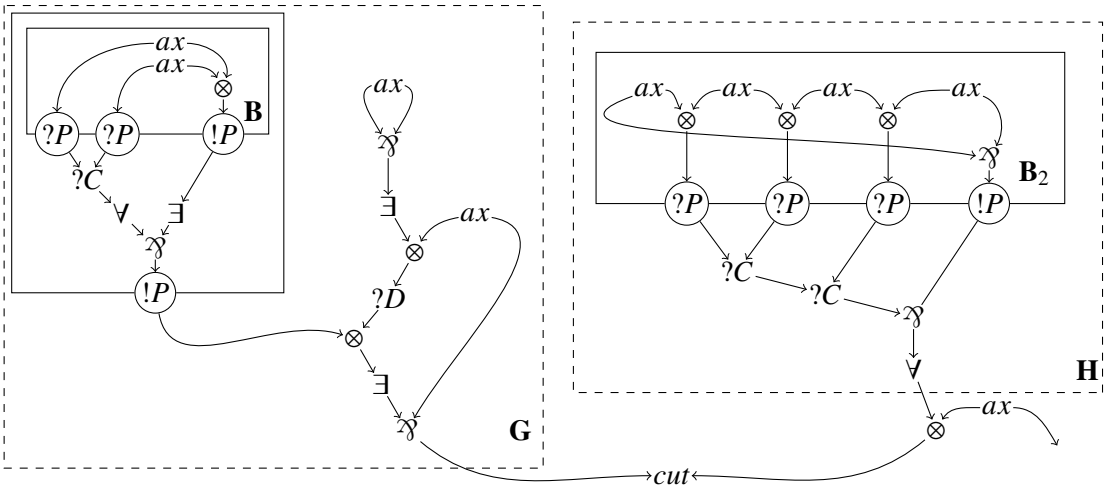


Figure 3.16: The complexity of G is not polynomial.

- If $?_iA$ and $?_jA$ can be contracted in $?_kA$, then $k \leq i$, $k \leq j$ **and at least one of those inequality is strict**. And for every box whose principal door is indexed by $!_kA$, the indexes on the ?-s of the auxiliary doors are smaller or equal to k.
- If $?_iA$ and $?_jA$ can be contracted in $?_kA$, then $k \leq i$, $k \leq j$. And for every box whose principal door is indexed by $!_kA$, the indexes on the ?-s of the auxiliary doors are smaller or equal to k **with all but (at most) one of those inequalities being strict**.

Instead of a criterion on type-systems, we propose here a criterion on proof-nets. Ours is more general: every proof-net of every *Poly* sound system of *MS* satisfies our dependence control condition. Our dependence control condition is rather close to the *MS* dependence control condition. However our stratification and nesting conditions are a lot more general than those of *MS*.

We try to have as few false negatives as possible for our criterion (proof-nets which are in *Poly* but do not satisfy the criterion) so we will only forbid proof-nets where, along the cut elimination, two (or more) residues of a box B join the same residue of B. Indeed, let us suppose that a chain of spindles appears during *cut*-elimination and that the boxes of the sequence are residues of pairwise distinct boxes of the original proof-net. Then, the length of the sequence is bounded by the number of boxes of the original proof-net. Because it is reasonable to assume that the number of boxes does not depend on the argument, the length of the sequences of spindle would be bounded by a number which does not depend on the argument. Our condition is given in the following way: we define a relation $B \succcurlyeq B'$ on boxes meaning that there exist residues B_1 and B_2 of B and residues B'_1 and B'_2 of B' such that $\sigma(B_1)$ and $\sigma(B_2)$ are cut with distinct auxiliary edge of B'_1 and B'_2. Our dependence control condition is the acyclicity of $\succcurlyeq$.

Let us observe that the relation $\succcurlyeq$ is defined by considering $\mapsto$-paths ending by a context on an (reversed) auxiliary edges of a box B' while the relation $\rightsquigarrow$ (Definition 48 in page 59) was defined by considering $\rightsquigarrow$-paths passing through the (reversed) principal edge of a box B'.

Definition 80. *We define a relation $\succcurlyeq$ on boxes by:*

$$B \succcurlyeq B' \Leftrightarrow \exists P, P'_1, P'_2 \in Pot, \exists t, u \in Sig, \exists i \neq j, \begin{cases} ((\sigma(B), P), [!_t]) \mapsto^+ ((\overline{\sigma_i(B')}, P'_1), [!_{\mathtt{e}}]) \text{ and} \\ ((\sigma(B), P), [!_u]) \mapsto^+ ((\overline{\sigma_j(B')}, P'_2), [!_{\mathtt{e}}]) \end{cases}$$

Definition 81. *A proof-net G is said* $\succ\!\!\!\rightarrow$-stratified *if* $\succ\!\!\!\rightarrow$ *is acyclic on* B_G.

For example, in Figure 3.14a, we have $B \succ\!\!\!\rightarrow A$ because $((\sigma(B),[]),[!_{\mathtt{l(e)}}]) \mapsto^2 ((\overline{\sigma_1(A)},[]),[!_{\mathtt{e}}])$ and $((\sigma(B),[]),[!_{\mathtt{r(e)}}]) \mapsto^2 ((\overline{\sigma_2(A)},[]),[!_{\mathtt{e}}])$. The proof-net of Figure 3.16 is not $\succ\!\!\!\rightarrow$-stratified because we have $((\sigma(B),[\mathtt{l(r(e))}]),[!_{\mathtt{l(e)}}]) \mapsto^{29} ((\overline{\sigma_1(B)},[\mathtt{r(e)}]),[!_{\mathtt{e}}])$ and $((\sigma(B),[\mathtt{l(r(e))}]),[!_{\mathtt{r(e)}}]) \mapsto^{29} ((\overline{\sigma_2(B)},[\mathtt{r(e)}]),[!_{\mathtt{e}}])$ and those paths imply $B \succ\!\!\!\rightarrow B$.

Lemma 82. *Let G be a $\rightsquigarrow\!\!\!\!\succ$-stratified proof-net, $s \in \mathbb{N}$ and (B,P) be a potential box with $d = s_{\succ}(B)$. There are at most $\left|Can_{s-1}(\vec{E}_G)\right|^d$ sequences $(e_i)_{1\le i\le l}$ of directed edges such that, there exists a potential sequence $(P_i)_{1\le i\le l}$, a trace sequence $(T_i)_{1\le i<l}$ and $t \in Sig$ such that:*

$$((\sigma(B),P),[!_t]) \mapsto_{S_s} ((e_1,P_1),T_1) \mapsto_{S_s} \cdots \mapsto_{S_s} ((e_{l-1},P_{l-1}),T_{l-1}) \mapsto_{S_s} ((e_l,P_l),[!_{\mathtt{e}}])$$

Proof. We prove it by induction on d. Let us suppose that $((\sigma(B),P),[!_t]) \mapsto_{S_s} ((e_1,P_1),T_1) \mapsto_{S_s} \cdots \mapsto_{S_s} ((e_{l-1},P_{l-1}),T_{l-1}) \mapsto_{S_s} ((e_l,P_l),[!_{\mathtt{e}}])$. If there exists a context in the path of the shape $((\sigma(C),Q),[!\_])$ with $s_{\succ}(C) < s_{\succ}(B)$, then we set k as the smallest index such that $((e_{k+1},P_{k+1}),T_{k+1})$ is such a context. Else, we set $k = l$.

First, let us notice that by induction hypothesis, there are at most $|Can_{s-1}(\vec{E}_G)|^{d-1}$ possibilities for $e_{k+1},\cdots,e_l$. Then, let us determine the number of possibilities for $e_1,\cdots,e_k$. There are at most $|Can_{s-1}(\vec{E}_G)|$ choices for $(e_k,P_k)^{s-1}$. Once $(e_k,P_k)^{s-1}$ is determined, we will prove by contradiction that it determines $e_1,\cdots,e_k$. Let us suppose that there exists two possible sequences: $e_1,\cdots,e_k$ and $e'_1,\cdots,e'_{k'}$, then we consider the last different edges:

$$((\sigma(B),P),[!_t]) \mapsto_{S_s} ((e_1,P_1),T_1) \mapsto_{S_s} \cdots ((e_{k-j},P_{k-j}),T_{k-j}) \cdots \quad \mapsto_{S_s} ((e_k,P_k),[!_{t_k}]) \quad \hookrightarrow ((\sigma(C),Q),[!_{t_k}])$$
$$((\sigma(B),P),[!_{t'}]) \mapsto_{S_s} ((e'_1,P'_1),T'_1) \mapsto_{S_s} \cdots ((e'_{k'-j},P'_{k'-j}),T'_{k'-j}) \cdots \mapsto_{S_s} ((e'_{k'},P'_{k'}),[!_{t'_{k'}}]) \hookrightarrow ((\sigma(C),Q'),[!_{t'_{k'}}])$$

with $(e_k,P_k)^{s-1} = (e'_{k'},P'_{k'})^{s-1}$, $e_{k-j} \neq e_{k'-j}$ and $\left[e_{1+k-j};\cdots;e_k\right] = \left[e'_{1+k'-j};\cdots;e'_{k'}\right]$. By Lemma 53, there exist signatures u,u' and sequences of traces $(U_i)_{1\le i\le k}$ and $(U'_i)_{1\le i\le k'}$ such that

$$((\sigma(B),P),[!_u]) \mapsto_{S_s} C_1 = ((e_1,P_1),U_1) \mapsto_{S_s} \cdots C_{k-j} = ((e_{k-j},P_{k-j}),U_{k-j}) \cdots \quad \mapsto_{S_s} C_k = ((e_k,P_k),[!_{\mathtt{e}}])$$
$$((\sigma(B),P),[!_{u'}]) \mapsto_{S_s} C'_1 = ((e'_1,P'_1),U'_1) \mapsto_{S_s} \cdots C'_{k'-j} = ((e'_{k'-j},P'_{k'-j}),U'_{k'-j}) \cdots \mapsto_{S_s} C'_{k'}((e'_{k'},P'_{k'}),[!_{\mathtt{e}}])$$

Let us prove by induction on i that for $0 \le i < j$, ${C_{k-i}}^{/s-1} = {C'_{k'-i}}^{/s-1}$. By supposition, we know that $(e_k,P_k)^{s-1} = (e_{k'},P'_{k'})^{s-1}$ so ${C_k}^{/s-1} = {C'_{k'}}^{/s-1}$. If $0 < i < j$ then, by induction hypothesis, we have ${C_{k+1-i}}^{/s-1} = {C_{k'+1-i}}^{/s-1}$. If the $C_{k-i} \mapsto C_{k+1-i}$ step is a $\rightsquigarrow$ step, then it is a $\rightsquigarrow_{s-1}$ step so by Theorem 72, ${C_{k-i}}^{s-1} = {C'_{k'-i}}^{s-1}$. If the $C_{k-i} \mapsto C_{k+1-i}$ step is a $\hookrightarrow$ step, because $e_{k-i} = e'_{k'-i}$, ${C_{k-i}}^{s-1} = {C'_{k'-i}}^{s-1}$.

So ${C_{k+1-j}}^{s-1} = {C'_{k'+1-j}}^{s-1}$. Let us examine the $C_{k-j} \mapsto C_{k+1-j}$ step. Every $\rightsquigarrow$ step of those paths is a $\rightsquigarrow_{S_{s-1}}$ step so, by Theorem 72, the $C_{k-j} \mapsto C_{k+1-j}$ step can not be a $\rightsquigarrow$ step. So these steps are of the shape:

$$C_{k-j} = ((\overline{\sigma_{i_1}(D)},P_{k-j}),[!_v]) \hookrightarrow ((\sigma(D),P_{k-j}),[!_v]) = C_{k+1-j}$$
$$C'_{k'-j} = ((\overline{\sigma_{i_2}(D)},P'_{k'-j}),[!_{v'}]) \hookrightarrow ((\sigma(D),P'_{k-j}),[!_{v'}]) = C'_{k'+1-j}$$

with $(C_{k+1-j})^{s-1} = (C'_{k'+1-j})^{s-1}$ and $(C_{k-j})^{s-1} \neq (C'_{k'-j})^{s-1}$. So $i_1 \neq i_2$. By definition of $\succ\!\!\!\rightarrow$, it means that $B \succ\!\!\!\rightarrow D$ and $s_{\succ}(D) < s_{\succ}(B)$. This contradicts the definition of k. So our hypothesis is false, $[e_1;\cdots;e_k] = [e'_1;\cdots;e'_{k'}]$.

Thus, there are at most $|Can_{s-1}(\vec{E}_G)|$ possibilities for $e_1,\cdots,e_k$ and at most $|Can_{s-1}(\vec{E}_G)|^{d-1}$ possibilities for $e_{k+1},\cdots,e_l$. In total, there are at most $|Can_{s-1}(\vec{E}_G)|^d$ possibilities for $e_1,\cdots,e_l$. □

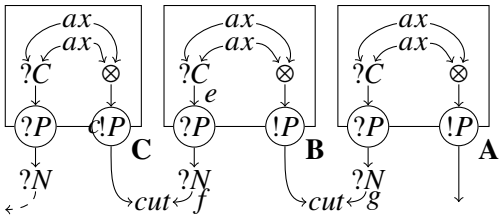


Figure 3.17: This proof-net (if extended to n boxes) reduces in $O(2^n)$ reduction steps.

3.3.2 Nesting

In this section, we will present a nesting condition (a condition preventing exponential blow-ups by chains of the type of Figure 3.17). First, we will analyse this blow-up from the point of view of context semantics. Let (B, P) be a potential box of a stratified proof-net G and let $t \in Cop_s(B, P)$. Then, there exists a path of the shape:

$$((\sigma(B), P), [!_t]) \mapsto_{S_s} ((e_1, P_1), T_1) \mapsto_{S_s} \cdots \mapsto_{S_s} ((e_{k-1}, P_{k-1}), T_{k-1}) \mapsto_{S_s} ((e_k, P_k), [!_{\mathtt{e}}])$$

In the previous subsection, we proved a bound on the number of possible sequences $e_1, \cdots, e_k$. Let us suppose that there is no context of the shape $((e_i, P_i), [!_{t_i}])$ with $\overline{e_i}$ being the conclusion of a $?N$ node. Then, there is no $\mathtt{n}(\_, \_)$ or $\mathtt{p}(\_)$ in t, t is only formed by $\mathtt{l}(\_)$, $\mathtt{r}(\_)$ and $\mathtt{e}$. One can reconstruct t by observing the list of contexts in the path, of the shape $((e_i, P_i), [!_{t_i}])$ with $\overline{e_i}$ being a premise of a contraction node.

To understand why the $?N$ nodes break this property, we can consider an example in Figure 3.17. We have $(\overline{e}, [\mathtt{n}(\mathtt{l}(\mathtt{e}), \mathtt{e})])^0 = (\overline{e}, [\mathtt{n}(\mathtt{r}(\mathtt{e}), \mathtt{e})])^0 = (\overline{e}, [\mathtt{e}])$ and

$$((\sigma(C), []), [!_{\mathtt{n}(\mathtt{e},\mathtt{n}(\mathtt{r}(\mathtt{e}),\mathtt{e}))}]) \mapsto^*_{S_1} ((\overline{e}, [\mathtt{n}(\mathtt{r}(\mathtt{e}), \mathtt{e})]), [!_{\mathtt{e}}])$$
$$((\sigma(C), []), [!_{\mathtt{n}(\mathtt{e},\mathtt{n}(\mathtt{l}(\mathtt{e}),\mathtt{e}))}]) \mapsto^*_{S_1} ((\overline{e}, [\mathtt{n}(\mathtt{l}(\mathtt{e}), \mathtt{e})]), [!_{\mathtt{e}}])$$

However $\mathtt{n}(\mathtt{e}, \mathtt{n}(\mathtt{l}(\mathtt{e}), \mathtt{e})) \neq \mathtt{n}(\mathtt{e}, \mathtt{n}(\mathtt{r}(\mathtt{e}), \mathtt{e}))$. If we follow the paths backwards we see that the crucial step is $((\overline{f}, []), [!_{\mathtt{n}(\mathtt{e},\mathtt{n}(\mathtt{r}(\mathtt{e}),\mathtt{e}))}]) \mapsto_1 ((\overline{\sigma_1(B)}, [], [!_{\mathtt{e}}; !_{\mathtt{n}(\mathtt{r}(\mathtt{e}),\mathtt{e})}])$ where a difference on the second trace element (which comes from a box B, with $s_{\rightsquigarrow}(B) = 1$) becomes a difference on the first trace element, which will correspond to the copy. The paths corresponding to $\mathtt{n}(\mathtt{e}, \mathtt{n}(\mathtt{l}(\mathtt{e}), \mathtt{e}))$ and $\mathtt{n}(\mathtt{e}, \mathtt{n}(\mathtt{r}(\mathtt{e}), \mathtt{e}))$ may be the same, but their simplifications are different and have different paths.

In fact, for every $u \in Cop_1(B, [])$, $\mathtt{n}(\mathtt{e}, u)$, $\mathtt{n}(\mathtt{l}(\mathtt{e}), u)$ and $\mathtt{n}(\mathtt{r}(\mathtt{e}), u)$ are in $Cop_1(C, [])$. So $|Cop_1(C, [])|$ depends affinely on $|Cop_1(B, [])|$. Similarly, $|Cop_1(B, [])|$ depends affinely on $|Cop_1(A, [])|$. We will define a relation $\prec\!\!\!\triangleleft$ on boxes capturing this dependence. For instance, we will have $C \prec\!\!\!\triangleleft B$ and $B \prec\!\!\!\triangleleft A$. If we extend this sequence to n boxes, the leftmost box (in the direction of Figure 3.17) has more than 3^{n-1} copies. As we noticed in Section 3.3.1 for the relation $\succ\!\!\!\!\rightarrow$ and Figure 3.15, if the length of $\prec\!\!\!\triangleleft$ does not depend on the argument, then those dependencies do not prevent the proof-net from normalizing in polynomial time. Because we can reasonably suppose that the number of boxes does not depend on the argument we will only require the relation $\prec\!\!\!\triangleleft$ to be acyclic.

Definition 83. *Let B and C be boxes of G, then*

$$B \prec\!\!\!\triangleleft C \Leftrightarrow \exists P, Q \in Pot, \exists t \in Cop(B, P), \exists v \sqsupset t, ((\sigma(B), P), [!_v]) \mapsto^* ((\sigma(C), Q), [!_{\mathtt{e}}])$$

A proof-net G is said $\prec\!\!\!\triangleleft$-stratified if $\prec\!\!\!\triangleleft$-is acyclic on B_G.

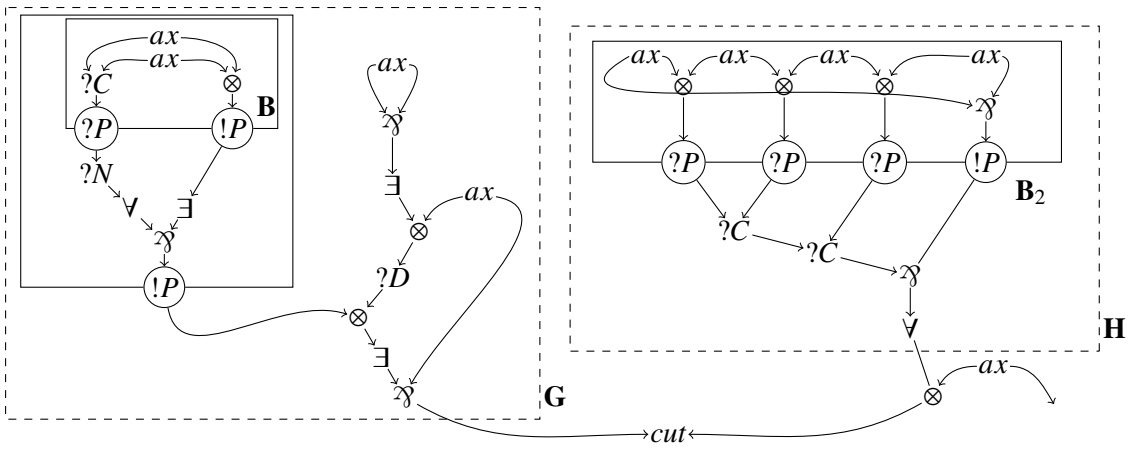


Figure 3.18: The complexity of G is not polynomial.

For example, in Figure 3.17, we have $B \prec\!\!\!\triangleleft A$ because $\mathtt{p(e)}$ is a strict simplification of $\mathtt{n(r(e),e)}$ and $((\sigma(B),[]),[!_{\mathtt{p(e)}}]) \mapsto^3 ((\sigma(A),[]),[!_{\mathtt{e}}])$. Similarly, in Figure 3.18, we have $((\sigma(B),[\mathtt{l(l(e))}]),[!_{\mathtt{p(e)}}]) \mapsto^3 ((\sigma(B),[\mathtt{l(r(e))}]),[!_{\mathtt{e}}])$ so $B \prec\!\!\!\triangleleft B$ and the proof-net of Figure 3.18 is not $\prec\!\!\!\triangleleft$-stratified.

To prove that $\prec\!\!\!\triangleleft$-stratification (together with $\rightsquigarrow$-stratification and $\succ\!\!\!\rightarrow$-stratification) implies polynomial time, we will need some technical lemmas to handle simplifications of copies.

In the following, we consider a $\rightsquigarrow$-stratified, $\prec\!\!\!\triangleleft$-stratified, $\succ\!\!\!\rightarrow$-stratified proof-net G. Let $s, n \in \mathbb{N}$, we set $T_{s,n} = \{B \in B_G \mid (s_{\rightsquigarrow}(B), s_{\prec\!\!\!\triangleleft}(B)) \leq_{lex} (s,n)\}$ with $\leq_{lex}$ referring to the usual strict lexicographic order: $(a,b) \leq_{lex} (a',b')$ iff $a < a'$ or $(a \leq a'$ and $b \leq b')$.

Lemma 84. *Let $\rightarrow \subseteq \rightarrowtail$ be two relations on contexts included in $\mapsto$. Let $(B,P) \in Can_{\rightarrowtail}(B_G)$ and $t, t' \in Cop_{\rightarrowtail}(\sigma(B),P)$ such that there exist sequences of contexts $(C_i)_{1\leq i\leq n}$ and $(C'_i)_{1\leq i\leq n}$ such that*

- $((\sigma(B),P),[!_t]) = C_1 \rightarrowtail C_2 \cdots C_n = ((e,Q),[!_{\mathtt{e}}])$, $((\sigma(B),P),[!_{t'}]) = C'_1 \rightarrowtail C'_2 \cdots C'_n = ((e,Q'),[!_{\mathtt{e}}])$ *and for every $1 \leq i \leq n$, $C_i^{\rightarrow} = C_i'^{\rightarrow}$,*
- *If $u \sqsupset t$, $((\sigma(B),P),[!_u]) \rightarrowtail^k ((\overline{\sigma_j(C)},Q),[!_v])$, the edges of those paths are the edges of $C_1, \cdots, C_k$ and $C_k = ((\overline{\sigma_j(C)},Q),V@[!_v])$ with $V \in Tra$ then $((\sigma(C),Q),[!_v])^{/\rightarrow} = v$.*
- *If $u' \sqsupset t'$, $((\sigma(B),P),[!_{u'}]) \rightarrowtail^k ((\overline{\sigma_j(C)},Q'),[!_{v'}])$, the edges of those paths are the edges of $C'_1, \cdots, C'_k$ and $C'_k = ((\overline{\sigma_j(C)},Q'),V'@[!_{v'}])$ with $V' \in Tra$, then $((\sigma(C),Q'),[!_{v'}])^{/\rightarrow} = v'$.*

Then, we have $t = t'$.

Proof. We will prove, by induction on $s_{\sqsubset}(u)$, that for every $u \in Sig$ $(u \sqsupseteq t) \Rightarrow (u \sqsupseteq t')$. This entails the result because, in particular, it gives us that $t \sqsupseteq t'$ so $t = t'$ (because t is a copy, it is standard so we can not have $t \sqsupset t'$).

We consider $u \sqsupseteq t$ and we suppose that for every signature $w \sqsupset u$, $w \sqsupset t'$. Let us consider the maximal signature w (for the order $\sqsupseteq$) such that $u \sqsubset w$. By induction hypothesis, $t' \sqsubset w$. Let us set u' as the minimal signature (for the order $\sqsupseteq$) such that $t' \sqsubseteq u' \sqsubset w$. We will show that $u = u'$ (thus we will have $u \sqsupseteq t'$). We can notice that the only differences between u and u' are on their leftmost branches. For $i \in \mathbb{N}$, we define $D_i = ((d_i,P_i),[!_{t_i}]@T_i)$ and $D'_i = ((d'_i,P'_i),T'_i)$ as the contexts such that $((\sigma(B),P),[!_u]) \rightarrowtail^i D_i$ and $((\sigma(B),P),[!_{u'}]) \rightarrowtail^i D'_i$.

By definition of $Cop_{\rightarrowtail}(\_)$, there exists $m \in \mathbb{N}$ such that D_m is of the shape $((f,R),[!_{\mathtt{e}}])$ (we choose the highest such m). Let us consider the lowest $k \in \mathbb{N}$ such that B_k is not of the shape $((d_i,P_i),[!\_]@T_i)$, this step (if it exists) must cross a $?N$ node upwards. We also consider the lowest $l \in \mathbb{N}$ such that the edge of C_l is different from the edge of D_l. If l exists then, by definition, k exists and $k \leq l$.

- If such a l does not exist, because the edges of $(C_i)_{1\leq i\leq m}$ and $(C'_i)_{1\leq i\leq m}$ are the same, the edges of $(D_i)_{1\leq i\leq m}$ and $(D'_i)_{1\leq i\leq m}$ are the same. In particular, D'_m is of the shape $((f,R),[!_{v'}])$. This gives us that $u \blacktriangleleft u'$, but we want $u = u'$. To prove $u = u'$, we need to prove that $v' = \mathtt{e}$.

  Either $n = m$ and in this case $D'_m = ((e,Q'),[!_{\mathtt{e}}])$. Or C_m is of the shape $((f,R),V.!_{\mathtt{e}})$ with $V \in Tra$. Because m is maximal, $((f,R),[!_{\mathtt{e}}]) \not\mapsto$ and $C_m = ((f,R),V.!_{\mathtt{e}}) \mapsto$. So $\overline{f}$ is the conclusion of a $?D$ node. Because, $C'_m \rightarrowtail C'_{m+1} \rightarrowtail^* C'_n$, we have $v' = \mathtt{e}$.

- Else, let us consider the paths $C_k \mapsto^* C_l$ and $D_k \mapsto^* D_l$. By Lemma 8, C_{l-1} and D_{l-1} are of the shape $((\overline{\sigma_j(C)},R),V.!_v)$ and $((\overline{\sigma_j(C)},R),[!_v])$ (the $\rightsquigarrow$ steps preserve the invariant "C_i is of the shape $((d_i,P_i),[!\_]@T_i)$"). Because ${C_{l-1}}^{\rightarrow} = {C'_{l-1}}^{\rightarrow}$ (by assumption of the lemma), C'_{l-1} is of the shape $((\overline{\sigma_j(C)},R'),V'.!_{v'})$ and the edges of those paths are the same. Then, because u differs only from u' on its leftmost branch, we can prove by induction on j that for $1 \leq j \leq l$, D_i and D'_i are of the shape $((f_i,Q_i),[!_{v_i}]@V_i)$ and $((f_i,Q_i),[!_{v'_i}]@V_i)$. In particular, D'_l is of the shape $((\overline{\sigma_j(C)},R),[!_{v'}])$. Because $C_l{}^{/\rightarrow} = C'_l{}^{/\rightarrow}$, we have $((\sigma_j(C),R),[!_v])^{/\rightarrow} = ((\sigma_j(C),R),[!_{v'}])^{/\rightarrow}$. Thus, because of the assumptions of the lemma, $v = ((\sigma_j(C),R),[!_v])^{/\rightarrow} = ((\sigma_j(C),R),[!_{v'}])^{/\rightarrow} = v'$. So $u = u'$.

□

Lemma 85. *For $s,n \in \mathbb{N} - \{0\}$ and every $(B,P) \in Can(B_G)$,*

$$\left|Cop_{\mapsto_{T_{s,n}}}(B,P)\right| \leq \left|Can_{s-1}(\vec{\mathrm{E}}_G)\right|^{|\succ\!\!\succ|} \cdot \left|\vec{\mathrm{E}}_G\right| \cdot \left(\max_{(C,Q)\in Pot(B_G)} \left|Cop_{\mapsto_{T_{s,n-1}}}(C,Q)\right|\right)^{\partial_G}$$

Proof. If $s_{\prec\!\!\prec}(B,P) > n$, then $Cop_{\mapsto_{T_{s,n}}}(B,P) = \{\mathtt{e}\}$ so the lemma stands. Else (if $s_{\prec\!\!\prec}(B,P) \leq n$), let us consider $t,t' \in Cop_{T_{s,n}}(B,P)$. By definition, there exists paths of the shape:

$$\begin{array}{llllll}((\sigma(B),P),[!_t]) \mapsto_{T_{s,n}} C_k = ((e_k,P_k),T_k) & \mapsto_{T_{s,n}} \cdots & \mapsto_{T_{s,n}} C_1 = ((e_1,P_1),T_1) & \mapsto_{T_{s,n}} ((e,Q),[!_{\mathtt{e}}])\\ ((\sigma(B),P),[!_{t'}]) \mapsto_{T_{s,n}} C'_{k'} = ((e'_{k'},P'_{k'}),T'_{k'}) \mapsto_{T_{s,n}} \cdots & \mapsto_{T_{s,n}} C'_1 = ((e'_1,P'_1),T'_1) & \mapsto_{T_{s,n}} ((e',Q'),[!_{\mathtt{e}}])\end{array}$$

Let us suppose that $[e_n;\cdots;e_1] = [e'_{n'};\cdots;e'_1]$. By Lemma 82, there are at most $\left|Can_{s-1}(\vec{\mathrm{E}}_G)\right|^{|\succ\!\!\succ|}$ possible choices for $[e_n;\cdots;e_1]$.

If $(e,Q)^{\mapsto_{T_{s,n-1}}} = (e',Q')^{\mapsto_{T_{s,n-1}}}$, then by Theorem 72, for $1 \leq i \leq k$, $C_i^{\mapsto_{T_{s,n-1}}} = C'_i{}^{\mapsto_{T_{s,n-1}}}$. Let us suppose that there exist $u \sqsupset t$, such that $((\sigma(B),P),[!_u]) \mapsto^l_{T_{s,n}} ((\overline{\sigma_j(C)},R),[!_v])$, the edges of those paths are the edges of $C_k,\cdots,C_{k-l}$ and $C_{k-l} = ((\overline{\sigma_j(C)},Q),V@[!_v])$ with $V \in Tra$ then there exists $t'' \blacktriangleleft t$ and $u'' \sqsupseteq t''$ such that $((\sigma(B),P),[!_{u''}]) \mapsto_{T_{s,n}} ((\overline{\sigma_j(C)},R),[!_{\mathtt{e}}])$. By definition of $\prec\!\!\prec$, $B \prec\!\!\prec C$ so $s_{\prec\!\!\prec}(C) \leq n-1$ and $((\sigma(C),R),[!_v])^{/\mapsto_{T_{s,n-1}}} = v$. We can prove a similar result for the path $C'_k,\cdots,C'_{k-l}$. And thus, by Lemma 84, $t = t'$.

So we proved that, if we choose $[e_n;\cdots;e_1]$ and $(e,Q)^{\mapsto_{T_{s,n-1}}}$ then t is uniquely determined. Thus,

$$\left|Cop_{\mapsto_{T_{s,n}}}(B,P)\right| \leq \left|Can_{s-1}(\vec{\mathrm{E}}_G)\right|^{|\succ\!\!\succ|} \cdot \left|Can_{\mapsto_{T_{s,n-1}}}(\vec{\mathrm{E}}_G)\right|^{\partial_G}$$

$$\left|Cop_{\mapsto_{T_{s,n}}}(B,P)\right| \leq \left|Can_{s-1}(\vec{\mathrm{E}}_G)\right|^{|\succ\!\!\succ|} \cdot \left|\vec{\mathrm{E}}_G\right| \cdot \left(\max_{(C,Q)\in Pot(B_G)} \left|Cop_{\mapsto_{T_{s,n-1}}}(C,Q)\right|\right)^{\partial_G}$$

□

Theorem 86. *Let $x = \left|\vec{E}_G\right|$, $S = |\rightsquigarrow|$, $D = |\succ\!\!\!\rightarrow|$, $N = |\prec|$, and $\partial = \partial_G$, then:*

$$\max_{(B,P)\in Pot(B_G)} |Cop(B,P)| \leq x^{D^S\cdot\partial^{N\cdot S}}$$

Proof. For $s,n \in \mathbb{N}$, we set $u_{0,n} = u_{s,0} = 1$ and $u_{s,n} = u^N_{s-1,N}\cdot x\cdot u^{\partial}_{s,n-1}$. Then, thanks to Lemma 85, we can prove by induction on (s,n) that $u_{s,n} \leq \max_{(B,P)\in Pot(B_G)} Cop_{\mapsto_{T_{s,n}}}(B,P)$. Let us prove by induction on n that for every $s,n\in\mathbb{N}$, $u_{s,n} \leq (x\cdot(u_{s-1,N})^D)^{\partial^{2\cdot n}}$. For $n=0$, we have $u_{s,n} = 1 \leq x\cdot u^D_{s-1,N}$, and if $n\geq 0$,

$$u_{s,n+1} = u^D_{s-1,N}\cdot x\cdot u^{\partial}_{s,n} \leq \left(x\cdot(u_{s-1,N})^D\right)\cdot\left(\left(x\cdot(u_{s-1,N})^D\right)^{\partial^{2\cdot n}}\right)^{\partial}$$

$$u_{s,n+1} \leq \left(x\cdot u^D_{s-1,N}\right)\cdot(x\cdot u^D_{s-1,N})^{\partial^{1+2\cdot n}} \leq \left(x\cdot u^D_{s-1,N}\right)^{1+\partial^{1+2\cdot n}} \leq (x\cdot u^D_{s-1,N})^{\partial^{2+2\cdot n}}$$

Then, let us set $M = D\cdot\partial^N$, we prove by induction on s that for every $s\in\mathbb{N}$, $u_{s,N}\leq x^{M^{2\cdot s}}$. For $s=0$, we have $u_{0,N} = 1 \leq x$. And if $n\geq 0$,

$$\begin{aligned} u_{s+1,N} &\leq x^{\partial^N}\cdot(u_{s,N})^M \leq x^{\partial^N}\cdot\left(x^{M^{2\cdot s}}\right)^M \\ &\leq x^{\partial^N}\cdot x^{M^{1+2\cdot s}} \leq x^{\partial^N + M^{1+2\cdot s}} \leq x^{M+M^{1+2\cdot s}} \\ u_{s+1,N} &\leq x^{M^{2+2\cdot s}} \end{aligned}$$

Finally, let us notice that $\mapsto_{T_{S,N}} = \mapsto$, so $Cop_{\mapsto_{T_{S,N}}}(B,P) = Cop(B,P)$. □

Corollary 87. *Let us consider a $\rightsquigarrow$-stratified, $\prec$-stratified, $\succ\!\!\!\rightarrow$-stratified proof-net G. Let $x = \left|\vec{E}_G\right|$, $S = |\rightsquigarrow|$, $D = |\succ\!\!\!\rightarrow|$, $N = |\prec|$, and $\partial = \partial_G$, then:*

$$W_G \leq x^{1+D^S\cdot\partial^{1+N\cdot S}}$$

Proof. By Theorem 86, we have

$$W_G = \left|Can(\vec{E}_G)\right| \leq \left|\vec{E}_G\right|\cdot\max_{(B,P)\in Pot(B_G)}|Cop(B,P)| \leq x\cdot\left(x^{D^S\cdot\partial^{N\cdot S}}\right)^{\partial}$$

$$W_G \leq x^{1+D^S\cdot\partial^{1+N\cdot S}}$$

□

The degree of the polynomial in the bound only depends on $|\rightsquigarrow|$, $|\succ\!\!\!\rightarrow|$, $|\prec|$, and ∂_G. Those four parameters are bounded by the number of boxes. So a stratified proof-net controlling dependence normalizes in a time bounded by a polynomial on the size of the proof-net, the polynomial depending only on the number of boxes of the proof-net.

Theorem 88. *Let us suppose that there exists a box-bounded[7] subsystem[8] S such that every proof-net of G_S is $\rightsquigarrow$-stratified, $\succ\!\!\!\rightarrow$-stratified and $\prec$-stratified, then S is sound for Poly.*

[7] cf. Definition 37 in Section 3.1.4

[8] cf. Definition 35 in Section 3.1.4

Proof. Let us consider a proof-net G whose only conclusion's label is of the shape $A \multimap B$ with $A \in \underline{B}_S$. By definition of *LL* subsystems, there exist $a, b \in \mathbb{N}$ such that for every binary list l, $\left|\vec{\mathrm{E}}_{Enc_A(l)}\right| \leq a \cdot |l| + b$. By definition of box-boundedness, there exists $n_A \in \mathbb{N}$ such that for every binary list l, there is at most n_A boxes in $Enc_A(l)$. We define n as $n_A + |B_G|$. Let us set

$$P = \left(a \cdot X + b + \left|\vec{\mathrm{E}}_G\right| + 3\right)^{1+n^{1+n+n^2}}$$

P is a polynomial so $P \in Poly$. And, for every binary list l, $\left|\vec{\mathrm{E}}_{Enc_A(l)}\right| \leq a \cdot |l| + b$ so $\left|\vec{\mathrm{E}}_{(G)Enc_A(l)}\right| \leq a \cdot |l| + b + \left|\vec{\mathrm{E}}_G\right| + 3$. We can also notice that $\left|B_{(G)Enc_A(l)}\right| \leq \left|B_{Enc_A(l)}\right| + |B_G| \leq n_A + |B_G| = n$.

Moreover, $(G)Enc_A(l)$ is $\rightsquigarrow$-stratified, $\succ\!\!\!\rightarrow$-stratified and $\prec\!\!\!\vartriangleleft$-stratified so $S = |\rightsquigarrow|$, $D = |\succ\!\!\!\rightarrow|$, $N = |\prec\!\!\!\vartriangleleft|$, and $\partial = \partial_G$ are lower than $\left|B_{(G)Enc_A(l)}\right|$ so are lower than n. Thus, by Corollary 87,

$$W_{(G)Enc_A(l)} \leq \left|\vec{\mathrm{E}}_{(G)Enc_A(l)}\right|^{1+D^S \cdot \partial^{1+N \cdot S}} \leq \left(a \cdot |l| + b + \left|\vec{\mathrm{E}}_G\right| + 3\right)^{1+n^n \cdot n^{1+n \cdot n}}$$

$$W_{(G)Enc_A(l)} \leq P(|l|)$$

Thus, by definition, S is sound for *Poly*. □

We defined intuitively the notions of "stratification condition", "dependence control condition" and "nesting condition" by giving examples of the kind of proof-nets those conditions should forbid. A more interesting way to view those conditions is to observe how they are used in the proofs leading to Theorem 87.

- Stratification allows us to trace back the $\rightsquigarrow$ paths. Thus, if $C \rightsquigarrow^* D$, we need a bounded amount of information on D to know all the edges of the path.
- Dependence control allows us to trace back the $\hookrightarrow$ steps. So, in presence of a stratification condition, if $C \mapsto^* D$, we need a bounded amount of information to know all the edges of the path.
- Nesting allows us to bound the strict simplifications of copies. If we know all the edges of the path $((\sigma(B), P), [!_t]) \mapsto^* D$, we need a bounded amount of information on D to know t.

Let us notice that these criteria are rather independent from one another. With these principles in mind, one can try to relax any of the three criteria. This is exactly what we do in Section 3.5: we try to push those principles to their limits. Before this, we prove lemmas in Section 3.4 which are used throughout this thesis.

3.4 Definition and acyclicity of $\nrightarrow$

In the following we will prove two generalizations of Lemma 21. Those generalizations are used in the Definition of $\rho_{G\to H}(\_)$ (Lemma 38), in Section 3.5.2 and Chapter 4.

Formally, Lemma 21 (Section 2.3) states that for every normalizing proof-net G, there is no path of the shape $((e,P),[!_t]) \mapsto^+ ((e,P),[!_u])$. So Lemma 21 proves that a path starting from the principal door of (B,P) can not arrive at an auxiliary door of (B,P). With Lemma 91, we prove that such a path can not get inside (B,P) by an auxiliary door. So, intuitively a path can not go from a door of (B,P) to the inside of (B,P). The acyclicity of $\nrightarrow$ (a relation defined in Definition 92) intuitively means that a path can not go from the inside of (B,P) to a door of (B,P).

3.4.1 From (B,P) to the inside of (B,P)

Along a sequence $G \to^*_{cut} H$ of reductions, a box B may have several residues: there might exist $B'_1 \neq B'_2$ such that $\pi_{G\to H}((\sigma(B'_1),P'_1),T'_1) = ((\sigma(B),P_1),T_1)$ and $\pi_{G\to H}((\sigma(B'_2),P'_2),T'_2) = ((\sigma(B),P_2),T_2)$. However, (B,P,t) has only one residue and, if the reductions does not involve the principal door of B, (B,P) has only one residue. This intuition is formalized by Lemma 89.

Lemma 89. *Let us consider a reduction $G \to_{cut} H$ and let us suppose that $\pi_{G\to H}((\sigma(B'_1),P'_1),T'_1.!_{t'_1}) = ((\sigma(B_1),P_1),T_1.!_{t_1})$ and $\pi_{G\to H}((\sigma(B'_2),P'_2),T'_2.!_{t'_2}) = ((\sigma(B_2),P_2),T_2.!_{t_2})$.*

- *If $(B_1,P_1,t_1) = (B_2,P_2,t_2)$ then $(B'_1,P'_1,t'_1) = (B'_2,P'_2,t'_2)$.*
- *If $(B_1,P_1) = (B_2,P_2)$ and the reduction does not involve $\sigma(B_1)$, then $(B'_1,P'_1) = (B'_2,P'_2)$.*

Proof. By observation of the definition of $\pi_{G\to H}(\_)$, (Definition 12, page 26). To better understand the premises of the lemma, let us suppose that $(B_1,P_1) = (B_2,P_2) = (B,P)$ and let us consider a $!P/?N$ step involving $\sigma(B)$. Either t_1 is of the shape $\mathtt{n}(t'_1,t''_1)$ and B'_1 is the inner residue of B, or t_1 is of the shape $\mathtt{p}(t'_1)$ and B'_1 is the outer residue of B. This is why, to deduce that $(B'_1,P'_1) = (B'_2,P'_2)$ (or even, to deduce that $B'_1 = B'_2$) we need to know that the top-connectives of t_1 and t_2 are the same. The case of a $!P/?C$ step involving $\sigma(B)$ is similar.

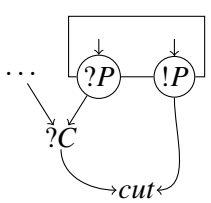
However, if the reduction step does not involve $\sigma(B)$, one does not need any information on t_1 and t_2 to deduce that $(B'_1,P'_1) = (B'_2,P'_2)$. For instance, let us suppose that the reduction step is a $!P/?C$ step involving $\sigma(C)$ and B is immediately included in C.

- If $P = Q.\mathtt{l}(u)$, then $(B'_1,P'_1) = (B'_2,P'_2) = (B_l,Q.u)$ with B_l the residue of B in the left residue of C.
- If $P = Q.\mathtt{r}(u)$, then $(B'_1,P'_1) = (B'_2,P'_2) = (B_r,Q.u)$ with B_r the residue of B in the right residue of C.
- The other cases are impossible because we supposed that $((\sigma(B),P),T_1.!_{t_1})$ and $((\sigma(B),P),T_2.!_{t_2})$ are in the image of $\pi_{G\to H}(\_)$.

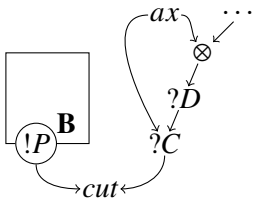
□

Lemmas 90 and 91 are based on the preceding observation (and Lemma 89). Intuitively, it is enough to consider the cases where the only *cut* node is the head of the box B considered. Figures 3.19a and 3.19b give an insight on the reason why the paths considered in Lemmas 90 and 91 are impossible.

Lemma 90. *There is no path of the shape $((\sigma(B),P),[!_t]) \mapsto^* ((\overline{\sigma_i(B)},P),T)$ with $(B,P) \in Can(B)$.*



(a) Intuition behind Lemma 90: $((\sigma(B),[]),[!_{\mathtt{e}}]) \rightsquigarrow^* ((\overline{\sigma_1(B)},[]),[!_{\mathtt{e}}])$ but this is not a valid proof-net.



(b) Intuition behind Lemma 91: $((\sigma(B),[]),[!_{\mathtt{l(e)}}]) \rightsquigarrow^* ((\overline{\sigma(B)},[]),\_.?_{\mathtt{r(e)}})$ and $\mathtt{l(e)} \neq \mathtt{r(e)}$.

Figure 3.19: Intuitions underlying the results of Section 3.4.1.

Proof. We prove it by contradiction. Let us suppose that there exists a path of the shape $((\sigma(B),P),[!_t]) \mapsto^* ((\overline{\sigma_i(B)},P),T)$. We will prove that it leads to a contradiction.

We first prove it in the case where every *cut* node is either the head of $\sigma(B)$ or a $!P/?W$ cut. In this case, let us suppose that $((\sigma(B),P),[!_t]) \mapsto^* ((\overline{\sigma_i(B)},P),T)$. Let us consider the last step of the path of the shape $((e,Q),U) \mapsto ((\overline{f},Q),U)$ with e,f premises of a *cut* node. Because $((\sigma(B),P),[!_t]) \mapsto^* ((e,Q),U)$, e is not the conclusion of a $?W$ node. Because $((\overline{f},Q),U) \mapsto^* ((\overline{\sigma_i(B)},P),T)$, f is not the conclusion of a $?W$ node. And because the path $((\overline{f},Q),U) \mapsto^* ((\overline{\sigma_i(B)},P),T)$ does not cross any *cut* node, $f \neq \sigma(B)$ (else all the edges of this path would be inside B) so $e = \sigma(B)$. This violates Theorem 1 of [37]: indeed there exists a switching containing all the edges of the path $((e,Q),U) \mapsto^* ((\overline{\sigma_i(B)},R),T)$ so there is a cyclic switching. Our hypothesis is false, there is no path of the shape $((\sigma(B),P),[!_t]) \mapsto^* ((\overline{\sigma_i(B)},P),T)$.

Else, we can reduce any *cut* node except the $!P/?W$ cuts and the ones whose premise is $\sigma(B)$. Let us write $G \rightarrow_{cut} H$ this reduction step. There exist $(B',P') \in Can(B_H)$ such that $\pi_{G\to H}((\sigma(B'),P'),[!_t]) = ((\sigma(B),P),[!_t])$ and $\pi_{G\to H}((\overline{\sigma_i(B')},P'),T) = ((\overline{\sigma_i(B)},P),T)$ (Lemma 89). By Lemma 13, we deduce that $((\sigma(B'),P'),[!_t]) \mapsto^* ((\overline{\sigma_i(B')},P'),T)$. We can keep on reducing until we obtain a proof-net whose *cut* nodes are either the head of $\sigma(B')$ or $!P/?W$ cuts. This is a contradiction. □

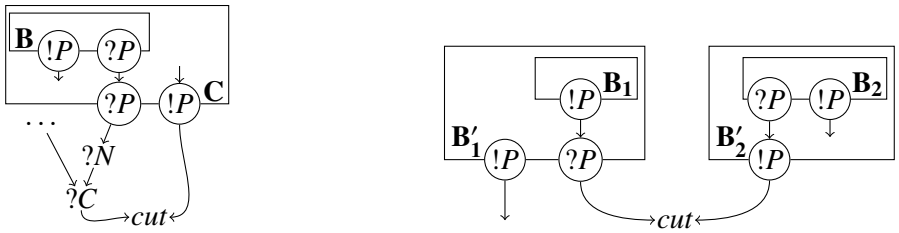
Lemma 91. *If* $((\sigma(B),P),[!_t]) \mapsto^* ((\overline{\sigma(B)},P),U.?_u)$*, then* $u \not\sqsubseteq t$.

Proof. We prove the lemma by contradiction. Let suppose that $((\sigma(B),P),[!_t]) \mapsto^* ((\overline{\sigma(B)},P),U.?_u)$ with $u \sqsubseteq t$. We can reduce every *cut* in the path except the $!P/?W$ cuts. Let us write $G \rightarrow_{cut} H$ this reduction step. Observing the definition of $\pi_{G\to H}(\_)$, one can verify that we are in one of the following cases:

- Either there exists a potential box (B',P') and $t',u' \in Sig$ such that $u' \sqsubseteq t'$, $\pi_{G\to H}((\sigma(B'),P'),[!_{t'}]) = ((\sigma(B),P),[!_t])$ and $\pi_{G\to H}((\overline{\sigma(B')},P'),U.?_{u'}) = ((\overline{\sigma(B)},P),U.?_u)$. In this case, by Lemma 13, we have $((\sigma(B'),P'),[!_{t'}]) \mapsto^* ((\overline{\sigma(B')},P'),U.?_{u'})$.

- Or $u = \mathtt{n}(u_1,u_2)$, $t = \mathtt{p}(t_2)$ with $u_2 \sqsubseteq t_2$, and the step from G to H is the reduction of the box B with a $?N$ node. In this case there exist boxes B_i and B_e (we use the same notations as in Figure 2.5) of H such that $\pi_{G\to H}((\sigma(B_e),P),[!_{t_2}]) = ((\sigma(B),P),[!_t])$ and $\pi_{G\to H}((\overline{\sigma(B_i)},P.u_2),U.?_{u_1}) = ((\sigma(B),P),U.?_{\mathtt{n}(u_1,u_2)})$. By Lemma 13, we have $((\sigma(B_e),P),[!_{t_2}]) \mapsto^* ((\overline{\sigma(B_i)},P.u_2),U.?_{u_1})$. Because there is only one choice for the last $\mapsto$ step, $((\sigma(B_e),P),[!_{t_2}]) \mapsto^* ((\overline{\sigma(B_e)},P),U.?_{u_1}.?_{u_2})$.

We can keep on reducing until we obtain a proof-net whose *cut* nodes are $!P/?W$ cuts. In this case, we know that the path crosses a *cut* node because $\sigma(B)$ is downward and $\overline{\sigma(B)}$ is upwards but in this case, we have: $((\sigma(B),P),[!_t]) \mapsto^+ ((e,Q),V) \mapsto ((\overline{f},Q),V) \mapsto^+ ((\overline{\sigma(B)},P),U.?_u)$ with either e or f being the conclusion of a $?W$ node, which is impossible. □



(a) Intuition behind the irreflexiveness of $\mapsto$. (b) Some intution behind Lemma 98.

Figure 3.20: Proof-nets providing the intuitions underlying Section 3.4.2.

3.4.2 From the inside of (B, P) to (B, P)

We use the notation $BoxSig_G$ for the set $\{(B, P, t) \mid (B, P) \in Pot(B_G), t \in Sig,$ and t is quasi-standard$\}$. Similarly we define $CanCop_G$ as the set $\{(B, P, t) \mid (B, P) \in Can(B_G), u \in Cop(B, P)$ and $t \sqsubseteq u\}$.

In the following we will need to prove the acyclicity of a relation $\mapsto$. Intuitively, $(B, P, t) \mapsto (C, Q, u)$ means that G reduces to a proof-net G' where the residue of (B, P, t) is inside the residue of (C, Q, u). Let us recall that we can write "the residue" instead of "a residue" because of Lemma 89.

First, we define a relation $\subseteq$ on $BoxSig_G$, based on $\sqsubseteq$. $(B, P, t) \subseteq (C, Q, u)$ means that the residues of (B, P, t) are inside the residues of (C, Q, u). Then, we define a relation $\smile$ on $BoxSig_G$, $(B, P, t) \smile (C, Q, u)$ means that G reduces to a proof-net G' where the residues of (B, P, t) and (C, Q, u) have fused (they were the two boxes of a $!P/?P$ step).

Definition 92. *Let $(B, P, t), (C, Q, u) \in BoxSig_G$, we write $(B, P, t) \subseteq (C, Q, u)$ if either ($(B, P) = (C, Q)$ and $t \sqsubseteq u$) or ($B \subset C$ and there exists $v \sqsubseteq u$ such that $P = Q.v@\_$). As in the signature case, we write $(B, P, t) \subset (C, Q, u)$ if $(B, P, t) \subseteq (C, Q, u)$ and $(B, P, t) \neq (C, Q, u)$.*

Let $(B, P, t), (C, Q, u) \in BoxSig_G$, we write $(B, P, t) \smile (C, Q, u)$ if $((\sigma(B), P), [!_t]) \mapsto^* ((\sigma(C), Q), [!_u])$.

We write $\mapsto$ as a shortcut for $\subseteq \smile$: let $(B, P, t) \neq (C, Q, u) \in BoxSig_G$, we write $(B, P, t) \mapsto (C, Q, u)$ if there exists $(D, R, v) \in BoxSig_G$ such that $(B, P, t) \subseteq (D, R, v)$ and $(D, R, v) \smile (C, Q, u)$.

Similarly, we write $(B, P, t) \hookleftarrow (C, Q, u)$ if there exists $(D, R, v) \in BoxSig_G$ such that $(B, P, t) \subseteq (D, R, v)$ and $(C, Q, u) \smile (D, R, v)$.

It is rather easy to prove that $\mapsto$ is irreflexive. Let us suppose that $(B, P, t) \subseteq (C, Q, v) \smile (B, P, u)$. Either $(B, P) = (C, Q)$ and $t \sqsubseteq v$, which contradicts Lemma 21. Or $B \subset C$ and there exists $v' \sqsubseteq v$ such that $P = Q.v'@\_$. Let us consider the path $((\sigma(C), Q), [!_v]) \mapsto^* ((\sigma(B), P), [!_u])$, this path enters (C, Q, v'). So either $((\sigma(C), Q), [!_v]) \mapsto^+ ((\overline{\sigma_i(C)}, Q), V.!_{v'})$ (which contradicts Lemma 90, as in Figure 3.20a) or $((\sigma(C), Q), [!_v]) \mapsto^+ ((\overline{\sigma(C)}, Q), V.?_{v'})$ (which contradicts Lemma 91).

The problem is to prove that there is no cycle of length ≥ 2. An idea would be to prove that $\mapsto$ is transitive: let us suppose that $(B, P, t) \mapsto (B', P', t') \mapsto (B'', P'', t'')$, then by definition of $\mapsto$ we have $(B, P, t) \subseteq (D, Q, u) \smile (B', P', t') \subseteq (D', Q', u') \smile (B'', P'', t'')$. To prove that $(B, P, t) \mapsto (B'', P'', t'')$, we need to prove that $\smile$ and $\subseteq$ commute. Then we would have $(B, P, t) \subseteq\subseteq \smile \smile (B'', P'', t'')$ so $(B, P, t) \subseteq \smile (B'', P'', t'')$ because $\subseteq$ and $\smile$ are transitive, as proven by the following lemma.

Lemma 93. *$\sqsubseteq$ and $\smile$ are orders on $BoxSig_G$*

Proof. Let $(B, P, t) \in BoxSig_G$, we can notice that $t \sqsubseteq t$ so $(B, P, t) \sqsubseteq (B, P, t)$. We can also notice $((\sigma(B), P), [!_t]) \mapsto^0 ((\sigma(B), P), [!_t])$ so $(B, P, t) \smile (B, P, t)$.

If $(B,P,t) \sqsubseteq (C,Q,u)$ and $(C,Q,u) \sqsubseteq (B,P,t)$. Then $B \subseteq C$ and $C \subseteq B$ so $B = C$. Thus $P = Q$, $t \sqsubseteq u$ and $u \sqsubseteq t$. Because $\sqsubseteq$ is an order, $t = u$.

If $(B,P,t) \smile (C,Q,u)$ and $(C,Q,u) \smile (B,P,t)$ then there exists $k,l \in \mathbb{N}$ such that $((\sigma(B),P),[!_t]) \mapsto^k ((\sigma(C),Q),[!_u])$ and $((\sigma(C),Q),[!_u]) \mapsto^l ((\sigma(B),P),[!_t])$. So $((\sigma(B),P),[!_t]) \mapsto^{k+l} ((\sigma(B),P),[!_t])$. By Lemma 21, $k + l = 0$. So $(B,P,t) = (C,Q,u)$.

Let us suppose that $(B,P,t) \sqsubseteq (C,Q,u) \sqsubseteq (D,R,v)$. If we are in the case $(B,P) = (C,Q)$ and $t \sqsubseteq u$ then: if $(C,Q) = (D,R)$ and $u \sqsubseteq v$ then $(B,P) = (D,R)$ and $t \sqsubseteq v$ so $(B,P,t) \sqsubseteq (D,R,v)$. And if $C \subset D$ and $Q = R.v'@\_$ with $v' \sqsubseteq v$ then $B \subset D$ and $P = Q = R.v'@\_$ with $v' \sqsubseteq v$ so $(B,P,t) \sqsubseteq (D,R,v)$. If we are in the case $B \subset C$ and $P = Q.u'@\_$ with $u' \sqsubseteq u$ then: if $(C,Q) = (D,R)$ and $u \sqsubseteq v$ then $B \subset C = D$ and $P = Q.u'@\_$ with $u' \sqsubseteq v$ so $(B,P,t) \sqsubseteq (D,R,v)$. And if $C \subset D$ and $Q = R.v'@\_$ with $v' \sqsubseteq v$ then $B \subset D$ and $P = R.v'@\_.u'@\_$ so $(B,P,t) \sqsubseteq (D,R,v)$.

Let us suppose that $(B,P,t) \smile (C,Q,u) \smile (D,R,v)$, then we have $((\sigma(B),P),[!_t]) \mapsto^* ((\sigma(C),Q),[!_u]) \mapsto^* ((\sigma(D),R),[!_v])$ so $((\sigma(B),P),[!_t]) \mapsto^* ((\sigma(D),R),[!_v])$ and $(B,P,t) \smile (D,R,v)$. □

So we want to prove that $\smile$ and $\subseteq$ commute. More precisely, in the case where $(D,Q,u) \smile (B',P',t') \subseteq (D',Q',u')$, we would like to prove that $(D,Q,u) \subseteq \smile(D',Q',u')$. The idea is to consider the path $((\sigma(D),Q),[!_u]) \mapsto ((e_k,P_k),T_k) \cdots ((e_1,P_1),T_1) \mapsto ((\sigma(B'),P'),[!_{t'}])$, and to prove by induction on i that for every $1 \leq i \leq k$, there exists a context $((f_i,Q_i),U_i)$ such that $((e_i,P_i),T_i) \subseteq ((f_i,Q_i),U_i)$ and $((f_i,Q_i),U_i) \mapsto^* ((\sigma(D'),Q'),[!_{u'}])$. With $\subseteq$ the following relation defined on contexts, corresponding to the relation $\subseteq$ on $BoxSig_G$.

Definition 94. *Let C and C' be contexts, we write $C \subseteq C'$ iff we are in one of the following situations:*

1. *$C = ((e,P.t@Q),T)$ and $C' = ((\sigma(B),P),[!_{t'}])$ with $e \in B$ and $t \sqsubseteq t'$*
2. *$C = ((e,P),T@[!_t]@U)$ and $C' = ((e,P),[!_{t'}]@U)$ with $t \sqsubseteq t'$*
3. *$C = ((e,P),T@[?_t]@U)$ and $C' = ((\overline{e},P),[!_{t'}]@U^\perp)$ with $t \sqsubseteq t'$*
4. *There exists a ?D node n, a potential Q and a trace T such that $C \mapsto^+ ((concl_n,Q),T.?_{\mathtt{e}})$ and $C' = ((\overline{concl_n},Q),[!_{\mathtt{e}}])$.*
5. *There exists a ?D node n, a potential Q and a trace T such that $C(\mapsto^*)(\leftarrow^+)((\overline{concl_n},Q),T.!_{\mathtt{e}})$, $C' = ((\overline{concl_n},Q),[!_{\mathtt{e}}])$ and the contexts of the $\mapsto^*$ path are distinct from the contexts of the $\leftarrow^+$ path.*

We define the relation $\nrightarrow$ on contexts as a shortcut for $\subset (\mapsto \cup \leftarrow)^*$.

The two first situations of Definition 94 correspond to the $\subseteq$ relation on $BoxSig_G$, as shown by the following lemma.

Lemma 95. *For every (B,P,t) and (B',P',t') in $BoxSig_G$, $(B,P,t) \subseteq (B',P',t')$ if and only if $((\sigma(B),P),[!_t]) \subseteq ((\sigma(B'),P'),[!_{t'}])$.*

Proof. Let us suppose that $(B,P,t) \subseteq (B',P',t')$. By definition of $\subseteq$ on $BoxSig_G$,

- Either $(B,P) = (B',P')$ and $t \sqsubseteq t'$. In this case, $((\sigma(B),P),[!_t]) \subseteq ((\sigma(B'),P'),[!_{t'}])$ by situation 2 of Definition 94.
- Or $B \subset B'$ and there exists $u' \sqsubseteq t'$ such that $P = P'.u'@\_$. Then, $((\sigma(B),P),[!_t]) \subseteq ((\sigma(B'),P'),[!_{t'}])$ by situation 1 of Definition 94.

Let us suppose that $((\sigma(B), P), [!_t]) \subseteq ((\sigma(B'), P'), [!_{t'}])$. Then,

- If we are in situation 1 of Definition 94, then $\sigma(B) \in B'$ and $P = P'.u'@\_$ with $u' \sqsubseteq t'$. Then, by definition of $\subseteq$ on $BoxSig_G$, $(B, P, t) \subseteq (B', P', t')$.
- If we are in situation 2 of Definition 94, so $(B, P) = (B', P')$ and $t \sqsubseteq t'$. Then, by definition of $\subseteq$ on $BoxSig_G$, $(B, P, t) \subseteq (B', P', t')$.
- Situation 3 is impossible because there is no $?\_$ trace element in the trace of $((\sigma(B), P), [!_t])$.
- Situations 4 and 5 are impossible because $\overline{\sigma(B')}$ can not be the conclusion of a $?D$ node.

□

The situations 3, 4 and 5 of Definition 94 are necessary to prove the commutation between the relations $\subseteq$ and $(\mapsto \cup \mapsfrom)$ (Lemma 98). Let us notice that it is weaker than what we announced previously: if $C \mapsto^* C'$ and $C' \subseteq D'$, then there might not exist a context D such that $C \subseteq D \mapsto^* D'$. For example, let us consider the proof-net of Figure 3.20b: we have $C = ((\sigma(B_1), [\mathtt{e}]), [!_\mathtt{e}]) \mapsto^* ((\sigma(B_2), [\mathtt{e}]), [!_\mathtt{e}]) = C'$ and $C' \subseteq ((\sigma(B'_2), []), [!_\mathtt{e}]) = D'$. The natural choice for D would be $((\sigma(B'_1), []), [!_\mathtt{e}])$. However, we do not have $D \mapsto^* D'$, only $D \mapsfrom^2 D'$.

We will often use the relation $(\mapsto \cup \mapsfrom)^*$. Let us notice that this relation is not equal to $\mapsto^* \cup \mapsfrom^*$. Indeed, if B is a box with at least 2 auxiliary doors, then $((\overline{\sigma_1(B)}, P), [!_t]) \mapsto ((\sigma(B), P), [!_t]) \mapsfrom ((\overline{\sigma_2(B)}, P), [!_t])$ but we have neither $((\overline{\sigma_1(B)}, P), [!_t]) \mapsto^* ((\overline{\sigma_2(B)}, P), [!_t])$ nor $((\overline{\sigma_2(B)}, P), [!_t]) \mapsto^* ((\overline{\sigma_1(B)}, P), [!_t])$. The following Lemma gives a characerization of $(\mapsto \cup \mapsfrom)^*$.

Lemma 96. *If $C(\mapsto \cup \mapsfrom)^*C'$, there exists a context D such that $C \mapsto^* D$ and $C' \mapsto^* D$.*

Proof. We prove it by induction on the length of the $(\mapsto \cup \mapsfrom)^*$ between C and C'. If $C = C'$ then we can set $D = C$. Else, there exists a context C'' such that $C(\mapsto \cup \mapsfrom)^*C''(\mapsto \cup \mapsfrom)C'$. By induction hypothesis, there exists a context D'' such that $C \mapsto^* D''$ and $C'' \mapsto^* D''$. If $C'' \mapsfrom C'$ then we can set $D = D''$ (we can verify that $C \mapsto^* D$ and $C' \mapsto^+ D$). If $C'' \mapsto C'$ and $C'' \mapsto^+ D''$ then we can set $D = D''$ (because $\mapsto$ is deterministic, we can verify that $C \mapsto^* D$ and $C'' \mapsto^* D$). Finally, if $C'' \mapsto C'$ and $C'' = D''$, then we set $D = C'$ (we can verify that $C \mapsto^+ D$ and $C' \mapsto^0 D$). □

Lemma 97. *Let $C, D, D' \in Cont_G$. If $C(\mapsto \cup \mapsfrom)D \subseteq D'$, then there exists C' such that $C \subseteq C'(\mapsto \cup \mapsfrom)^*D'$.*

Proof. First let us study some of the easy cases.

- Let us suppose that $C = ((c, P.t@Q), T) \mapsto ((d, P.t@Q), T) = D$ (crossing a *cut* node) and $D' = ((\sigma(B), P), [!_{t'}])$ with $d \in B$ and $t \sqsubseteq t'$. We can notice that $c \in B$ so we set $C' = D'$ (we can notice that $C' \mapsto^0 D'$).
- Let us suppose that $C = ((c, P), T@[!_t]@U) \mapsfrom ((d, P), T@[!_t]@U.\parr_l) = D$ (crossing a $\otimes$ node upwards) and $D' = ((d, P), [!_{t'}]@U.\parr_l)$ with $t \sqsubseteq t'$. Then we set $C' = ((c, P), [!_{t'}]@U)$ (we can notice that $C' \mapsfrom D'$).
- Let us suppose that $C = ((c, P), T@[?_t]@U) \mapsto ((d, P), T@[?_t]@U.\forall) = D$ (crossing a $\forall$ node downwards) and $D' = ((\overline{d}, P), [!_{t'}]@U^\perp.\exists)$ with $t \sqsubseteq t'$. Then we set $C' = ((\overline{c}, P), [!_{t'}]@U^\perp)$ (we can notice that $C' \mapsfrom D'$).

- Let us suppose that $C = ((c,P),T) \mapsto ((d,P),T) = D$ (crossing an ax node) and there exists a $?D$ node n such that $D' = ((\overline{concl_n},Q),[!_{\mathtt{e}}])$ and $D \mapsto^+ ((concl_n,Q),U.?_{\mathtt{e}})$. Then, we have $D \mapsto^+ ((concl_n,Q),U.?_{\mathtt{e}})$. So we set $C' = D'$.

- Let us suppose that $C = (\underline{(c,P)},T) \mapsto ((d,P),T) = D$ (crossing an ax node) and there exists a $?D$ node n such that $D' = ((\overline{concl_n},Q),[!_{\mathtt{e}}])$ and $D(\mapsto^*)(\leftarrow^+)((concl_n,Q),U.?_{\mathtt{e}})$. In this case, we have $C(\mapsto^+)(\leftarrow^+)((concl_n,Q),U.?_{\mathtt{e}})$. So we set $C' = D'$.

Now we will study the interesting cases. For the most part, they are the cases where the "situation" we will use to prove that $C \subseteq C'$, is not the same as the one proving that $D \subseteq D'$ (cf Definition 94).

- If $D \subseteq D'$ is in the situation 1 of Definition 94.
  - If $C = ((\sigma(B),P),T.!_t) \leftarrow ((d,P.t),T) = D$ and $D' = ((\sigma(B),P),[!_{t'}])$ with $t \sqsubseteq t'$, then we set $C' = D'$. One can verify that $C \subseteq C'$ (situation 2).
  - If $C = ((\overline{\sigma(B)},P),T.?_t) \mapsto ((d,P.t),T) = D$ and $D' = ((\sigma(B),P),[!_{t'}])$ with $t \sqsubseteq t'$, then we set $C' = D'$. One can verify that $C \subseteq C'$ (situation 3).
  - If $C = (\underline{(\sigma_i(B),P)},T.?_t) \leftarrow ((d,P.t),T) = D$ and $D' = ((\sigma(B),P),[!_{t'}])$ with $t \sqsubseteq t'$, then we set $C' = ((\overline{\sigma_i(B)},P),[!_{t'}])$. One can verify that $C \subseteq C'$ (situation 3) and $C' \mapsto D'$.
  - If $C = (\underline{(\overline{\sigma_i(B)},P)},T.!_t) \mapsto ((d,P.t),T) = D$ and $D' = ((\sigma(B),P),[!_{t'}])$ with $t \sqsubseteq t'$, then we set $C' = ((\overline{\sigma_i(B)},P),[!_t])$. One can verify that $D \subseteq D'$ (situation 2) and $C' \mapsto D'$.

- If $D \subseteq D'$ is in the situation 2 of Definition 94.
  - If $C = ((c,P),T@[!_u;!_v]) \leftarrow ((d,P),T@[!_{\mathtt{n}(u,v)}]) = D$ (crossing a $?N$ node upwards) and $D' = ((d,P),[!_{\mathtt{p}(v')}])$ with $v \sqsubseteq v'$, then we set $C' = ((c,P),[!_{v'}])$. One can verify that $C \subseteq C'$ (situation 2) and $C' \leftarrow D'$.
  - If $C = ((c,P),T@[!_{\mathtt{n}(u,v)}]) \mapsto ((d,P),T@[!_u;!_v]) = D$ (crossing a $?N$ node upwards) and $D' = ((d,P),[!_{v'}])$ with $v \sqsubseteq v'$, then we set $C' = ((c,P),[!_{\mathtt{p}(v')}])$. One can verify that $C \subseteq C'$ (situation 2) and $C' \mapsto D'$.
  - If $C = ((c,P),T@[!_{\mathtt{n}(u,v)}]) \mapsto ((d,P),T@[!_u;!_v]) = D$ (crossing a $?N$ node upwards) and $D' = ((d,P),[!_{u'};!_v])$ with $u \sqsubseteq u'$, then we set $C' = ((c,P),[!_{\mathtt{n}(u',v)}])$. One can verify that $C \subseteq C'$ (situation 2) and $C' \mapsto D'$.
  - If $C = ((c,P.t),T) \leftarrow ((\overline{\sigma_i(B)},P),T@[!_t]) = D$ (crossing an auxiliary door upwards) and $D' = ((\overline{\sigma_i(B)},P),[!_{t'}])$ with $t \sqsubseteq t'$, then we set $C' = ((\sigma(B),P),[!_{t'}])$. One can verify that $C \subseteq C'$ (situation 1) and $D' \mapsto C'$.
  - If $C = ((c,P),U) \leftarrow ((\overline{concl_n},P),U.!_{\mathtt{e}}) = D$ (crossing a $?D$ node upwards) and let us suppose that $D' = ((\overline{concl_n},P),[!_{\mathtt{e}}])$, then we set $C' = D'$. We can verify that $C(\mapsto^0) \leftarrow ((\overline{concl_n},P),U.!_{\mathtt{e}})$ so $C \subseteq C'$ (situation 5).

- If $D \subseteq D'$ is in the situation 3 of Definition 94.
  - If $C = (\underline{(c,P)},T@[?_{\mathtt{n}(u,v)}]) \leftarrow ((d,P),T@[?_u;?_v]) = D$ (crossing a $?N$ node downwards) and $D' = ((\overline{d},P),[!_{v'}])$ with $v \sqsubseteq v'$, then we set $C' = ((\overline{c},P),[!_{\mathtt{p}(v')}])$. One can verify that $C \subseteq C'$ (situation 3) and $C' \mapsto D'$.

– If $C = ((c, P), T@[?_u; ?_v]) \mapsto ((d, P), T@[?_{\mathtt{n}(u,v)}]) = D$ (crossing a $?N$ node downwards) and $D' = ((\overline{d}, P), [!_{\mathtt{p}(v')}])$ with $v \sqsubseteq v'$, then we set $C' = ((\overline{c}, P), [!_{v'}])$. One can verify that $C \subseteq C'$ (situation 3) and $C' \hookleftarrow D'$.

– If $C = ((c, P.t), T) \hookleftarrow ((\overline{\sigma(B)}, P), T@[?_t]) = D$ (crossing a principal door upwards) and $D' = ((\sigma(B), P), [!_{t'}])$ with $t \sqsubseteq t'$, then we set $C' = D'$. One can verify that $C \subseteq C'$ (situation 1).

– If $C = ((c, P.t), T) \mapsto ((\sigma_i(B), P), T@[?_t]) = D$ (crossing an auxiliary door downwards) and $D' = ((\overline{\sigma_i(B)}, P), [!_{t'}])$ with $t \sqsubseteq t'$, then we set $C' = ((\sigma(B), P), [!_{t'}])$. One can verify that $C \subseteq C'$ (situation 1) and $C' \hookleftarrow D'$.

– If $C = ((c, P), U) \mapsto ((concl_n, P), U.?_{\mathtt{e}}) = D$, crossing a $?D$ node downwards, and $D' = ((\overline{concl_n}, P), [!_{\mathtt{e}}])$, then we set $C' = D'$. One can verify that $C \mapsto ((concl_n, P), U.?_{\mathtt{e}})$ so $C \subseteq C'$ (situation 4).

- Let us suppose that $D \subseteq D'$ is in the situation 4 of Definition 94. There exists a $?D$ node n, a potential Q and a trace T such that $D \mapsto^+ ((concl_n, Q), T.?_{\mathtt{e}})$ and $D' = ((\overline{concl_n}, Q), [!_{\mathtt{e}}])$. If $C \mapsto D$, then $C \mapsto^+ ((concl_n, Q), T.?_{\mathtt{e}})$ so we can set $C' = D'$ and we have $C \subseteq C'$ (situation 4). If $C \hookleftarrow D$, then $C \mapsto^* ((concl_n, Q), T.?_{\mathtt{e}})$.

  – Either $C \mapsto^+ ((concl_n, Q), T.?_{\mathtt{e}})$ then $C \subseteq D'$ (situation 4) so we can set $C' = D'$.

  – Or $C = ((concl_n, Q), T.?_{\mathtt{e}})$ so $C \subseteq D'$ (situation 3) so we can set $C' = D'$.

- Let us suppose that $D \subseteq D'$ is in the situation 5 of Definition 94. There exists a $?D$ node n, a potential Q and a trace T such that $D(\mapsto^*)(\hookleftarrow^+)((\overline{concl_n}, Q), T.!_{\mathtt{e}})$, $D' = ((\overline{concl_n}, Q), [!_{\mathtt{e}}])$ and the contexts of the $\mapsto^*$ path are distinct from the contexts of the $\hookleftarrow^+$ path.

  – Let us first suppose that $C \mapsto D$ and $D \mapsto^0 (\hookleftarrow^+)((\overline{concl_n}, Q), T.!_{\mathtt{e}})$, then there exists a context E such that $D \hookleftarrow E \hookleftarrow^* ((\overline{concl_n}, Q), T.!_{\mathtt{e}})$.

    * Either $C \neq E$ and in this case $C \mapsto (\hookleftarrow^+)((\overline{concl_n}, Q), T.!_{\mathtt{e}})$ and the contexts of the $\mapsto^*$ path are distinct from the contexts of the $\hookleftarrow^+$ path so $C \subseteq D'$ (situation 5) and we can set $C' = D'$.

    * Or $C = E$ and $C(\hookleftarrow^*)((\overline{concl_n}, Q), T.!_{\mathtt{e}})$. In this case, either $C(\hookleftarrow^+)((\overline{concl_n}, Q), T.!_{\mathtt{e}})$ and $C \subseteq D'$ (situation 5) or $C = ((\overline{concl_n}, Q), T.!_{\mathtt{e}})$ and $C \subseteq D'$ (situation 2). In both cases, we can set $C' = D'$.

  – In the other cases, we stay in situation 5.

□

Corollary 98. *Let $C, D, D' \in Cont_G$. If $C(\mapsto \cup \hookleftarrow)^* D \subseteq D'$, there exists C' such that $C \subseteq C'(\mapsto \cup \hookleftarrow)^* D'$.*

Proof. Simple induction on the length of the $(\mapsto \cup \hookleftarrow)^*$ path, using Lemma 97. □

Lemma 97 is the main technical lemma of this section. We will use it to prove that the relation $\nrightarrow$ is acyclic on contexts of any proof-net G. We will consider a $\nrightarrow$ cycle and will prove that it leads to a contradiction. If G is in normal form, then the $\mapsto$ paths of the situations 4 and 5 of Definition 94 can not cross a *cut* node. Thus, it is easier to find a contradiction when G is in normal form. In the general case, we will reduce the proof-net as much as possible while preserving the $\nrightarrow$ cycle, thanks to the following lemma.

Lemma 99. *If $G \rightarrow_{cut} H$, then $C \subseteq D$ iff $\pi_{G\rightarrow H}(C) \subseteq \pi_{G\rightarrow H}(D)$.*

Proof. Simple observations of the $\pi_{G\rightarrow H}(\_)$ mapping, using Lemma 13.

For instance, let us suppose that the *cut*-elimination step from G to H is a $!P/?C$ step duplicating the box B. We consider an edge $c \in B$. We write c_l and B_l the left residues of c and B. We then consider the contexts $C = ((c_l, P.t@Q), T)$ and $D = ((\sigma(B_l), P), [!_{t'}])$ of H with $t \sqsubseteq t'$ (by situation 1 of Definition 94, we have $C \subseteq D$). Let us notice that $\pi_{G\rightarrow H}(C) = ((c, P.\mathtt{1}(t)@Q), T)$ and $\pi_{G\rightarrow H}(D) = ((\sigma(B), P), [!_{\mathtt{1}(t')}])$. By definition of $\sqsubseteq$, because $t \subseteq t'$, we have $\mathtt{1}(t) \sqsubseteq \mathtt{1}(t')$ so $\pi_{G\rightarrow H}(C) \subseteq \pi_{G\rightarrow H}(D)$ (situation 1). □

Lemma 100. *There is no $\nrightarrow$ sequence of the shape $((e, P), [!_t]) \nrightarrow^+ ((e, P), [!_u])$.*

Proof. We prove the lemma by contradiction. Let us suppose that $((e, P), [!_t]) \nrightarrow^* ((e, P), [!_u])$. Then, by Corollary 98, there exists a path of the shape $((e, P), [!_t]) \subseteq C_1 \subseteq \cdots \subseteq C_k(\mapsto \cup \leftarrow)^*((e, P), [!_u])$. By Lemma 99, we can suppose that every *cut* node of the proof-net has a premise which is either e, the conclusion of a $?W$ node, or the edge of a context C_i (with $1 \leq i \leq k$).

We can build the proof-net in such a way that in the step creating the edge of C_i, the edge of C_{i-1} is already created. If the edges of every C_i was e, then it would violate Lemma 21. Else, we are in one of the following cases:

- If there is some C_i of the shape $((\sigma(B), Q), [!_v])$, then e is included in strictly more boxes than C_k: $C_k = ((e_k, P_k), [!_{t_k}])$ and $P = P_k.u_k@\_$ with $u_k \sqsubseteq t_k$. This would contradict Lemma 90.
- Or for every $1 \leq i \leq k$, C_i is of the shape $((\overline{concl_{n_i}}, Q_i), [!_{\mathtt{e}}])$ (with n_i a $?D$ node). For $1 \leq i < k$, because $C_i \not\mapsto$, there exists $V_i \in Tra$ such that, $((\overline{concl_{n_i}}, Q_i), [!_{\mathtt{e}}])(\mapsto^0) \leftarrow^+ ((\overline{concl_{n_{i+1}}}, Q_{i+1}), V_i.!_{\mathtt{e}})$. One can verify, that this violates the correctness criterion of proof-nets [37] (it is similar to the proof of Lemma 90).

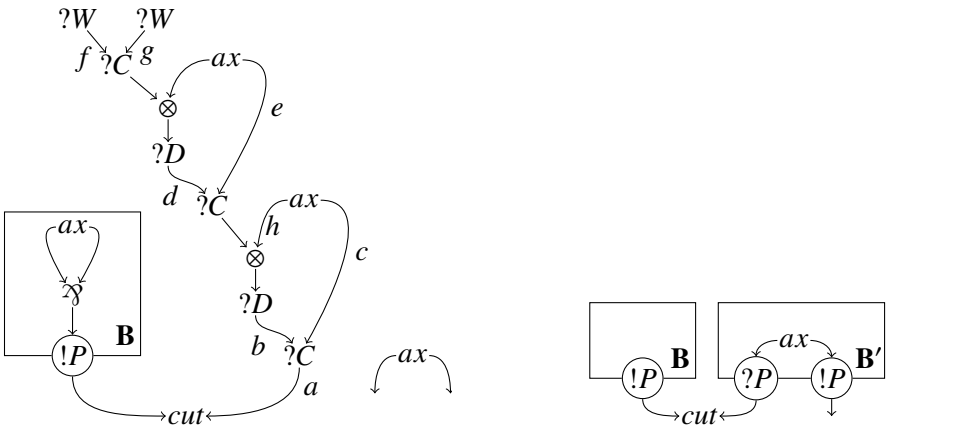
□

Lemma 101. *There is no sequence of the shape $(B, P, t) \nrightarrow^+ (B, P, u)$ or $(B, P, t) \nleftrightarrow^+ (B, P, u)$.*

Proof. Let us suppose that $(B, P, t) \nrightarrow^* (B, P, u)$. By definition of $\nrightarrow$ on $BoxSig_G$, there exists a sequence of elements of $BoxSig_G$ of the shape: $((\sigma(B), P), [!_t]) = ((\sigma(B_0), P_0), [!_{t_0}]) \subseteq ((\sigma(C_0), Q_0), [!_{u_0}]) \mapsto^* ((\sigma(B_1), P_1), [!_{t_1}]) \cdots \mapsto^* ((\sigma(B_k), P_k), [!_{t_k}]) \subseteq ((\sigma(C_k), Q_k), [!_{u_k}]) = ((\sigma(B), P), [!_u])$.

Either, for every $0 \leq i \leq k$ $(B_i, P_i, t_i) = (C_i, Q_i, u_i)$, in this case $((\sigma(B), P), [!_t]) \mapsto^+ ((\sigma(B), P), [!_u])$ which contradicts Lemma 90. Or there exists $0 \leq i \leq k$ such that $((\sigma(B_i), P_i), [!_{t_i}]) \subset ((\sigma(C_i), Q_i), [!_{u_i}])$, and in this case we can notice that there exists $u_i \in Sig$ such that $((\sigma(B_i), P_i), [!_{t_i}]) \nrightarrow^+ ((\sigma(B_i), P_i), [!_{u_i}])$ which contradicts Lemma 100.

Similarly, if $(B, P, t) \nleftrightarrow^* (B, P, u)$ either $((\sigma(B), P), [!_u]) \mapsto^+ ((\sigma(B), P), [!_t])$ (which contradicts Lemma 90) or there exists $(B_i, P_i, t_i) \in BoxSig_G$ and $u_i \in Sig$ such that $((\sigma(B_i), P_i), [!_{t_i}]) \nrightarrow^+ ((\sigma(B_i), P_i), [!_{u_i}])$ (which contradicts Lemma 100). □



(a) This proof-net is not $\rightsquigarrow\!\!\!\!\!\cdot\!\!>$-stratified. (b) Proof-nets can be made $\mathcal{R}_1$-stratified.

Figure 3.21: Motivating examples for the definition of $\rightsquigarrow\!\!\!\!\!\cdot\!\!>$.

3.5 More expressive polynomial time characterization

In Section 3.3, we defined three relations $\rightsquigarrow$, $\succ$ and $\prec$ such that the acyclicity of $\rightsquigarrow$ is a stratification condition, the acyclicity of $\succ$ is a dependence control and the acyclicity of $\prec$ is a nesting condition. If every proof-net of a subsystem satisfies the three conditions, then this subsystem is sound for *Poly*.

In order to define more expressive subsystems sound for *Poly*, we want to define smaller relations whose acyclicity still are stratification/dependence control/nesting conditions. We improve the stratification condition (which becomes entangled with dependence control) in Section 3.5.1. The nesting condition is improved in Section 3.5.2. Those results are used to define the subsystem $SwLL$ of Linear Logic, but they are not used in Chapter 4 and Section 5.1.1.

3.5.1 Improved stratification condition

3.5.1.1 Motivating examples

To understand why $\rightsquigarrow$ is too large and produces too much false negatives, we will observe several examples where $((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(C)},Q),[!_{t_1}]@U.?_u)$ so $(B,P)\rightsquigarrow(C,Q)$ but this pair does not seem necessary because the copies of (B,P) do not depend on the copies of (C,Q).

Whenever the $\rightsquigarrow$-path leaves (C,Q,u) Let us observe the proof-net of Figure 3.21a. The proof-net is not $\rightsquigarrow$-stratified because $((\sigma(B),[]),[!_{\mathtt{r(e)}}]) \rightsquigarrow^* ((\overline{\sigma(B)},[]),[!_{\mathtt{e}};\otimes_r;?_{\mathtt{l(e)}}])$ so $(B,[])\rightsquigarrow(B,[])$ and $B \rightsquigarrow B$. However, one can still prove a result similar to Lemma 74. Lemma 74 states that, whenever $((\sigma(B),P),[!_t]) \mapsto_{S_n} C_k \cdots \mapsto_{S_n} C_0$ and ${C_0}^{n-1} = {C'_0}^{n-1}$, then there exists $(C'_i)_{0\leq i\leq k}$ such that $C'_k \mapsto_{S_n} \cdots \mapsto_{S_n} C'_0$ and, for $0 \leq i \leq k$, ${C_i}^{n-1} = {C'_i}^{n-1}$. This lemma is essential to prove the elementary and polynomial bounds because it allows us to bound the number of sequences of edges $e_k,\cdots,e_0$ such that there is a path of the shape $((\sigma(B),P),[!_t]) \mapsto_{S_n} ((e_k,\_),\_) \cdots \mapsto_{S_n} ((e_0,\_),[!\_])$.

On the proof-net of Figure 3.21a, we associate to every $t \in Cop(B,[])$, a potential edge (e_t,Q_t) such that $((\sigma(B),[]),[!_t]) \mapsto^* ((e_t,Q_t),[!_{\mathtt{e}}])$. We define $(e_{\mathtt{e}},Q_{\mathtt{e}}) = (\sigma(B),[])$, $(e_{\mathtt{l(e)}},Q_{\mathtt{l(e)}}) = (b,[])$, $(e_{\mathtt{r(e)}},Q_{\mathtt{r(e)}}) = (c,[])$, $(e_{\mathtt{r(l(e))}},Q_{\mathtt{r(l(e))}}) = (d,[])$, $(e_{\mathtt{r(r(e))}},Q_{\mathtt{r(r(e))}}) = (e,[])$, $(e_{\mathtt{r(r(l(e)))}},Q_{\mathtt{r(r(l(e)))}}) = (f,[])$, and finally $(e_{\mathtt{r(r(r(e)))}},Q_{\mathtt{r(r(r(e)))}}) = (g,[])$. We can prove by induction on i that, if $C_i \rightsquigarrow^i ((e_t,Q_t),[!_{\mathtt{e}}])$ and $C'_i \rightsquigarrow^i$

$((e', Q'), [!_{\mathtt{e}}])$ with $(e_t, Q_t)^{\varnothing} = (e', Q')^{\varnothing}$ then[9] $C_i{}^{\varnothing} = C'_i{}^{\varnothing}$. Most of the steps can be proved using Theorem 72. The only interesting step is whenever C_{i-1} is of the shape $((\overline{\sigma(B)}, []), T.?_u)$. For instance:

$$((\sigma(B), []), [!_{\mathtt{r(1(e))}}]) \rightsquigarrow^6 C_{11} = ((a, []), [!_{\mathtt{1(e)}}; \otimes_r; ?_{\mathtt{1(e)}}]) \rightsquigarrow C_{10} = ((\overline{\sigma(B)}, []), [!_{\mathtt{1(e)}}; \otimes_r; ?_{\mathtt{1(e)}}])$$
$$\rightsquigarrow^6 C_4 = ((\overline{a}, []), [!_{\mathtt{1(e)}}; \parr_l; !_{\mathtt{1(e)}}]) \rightsquigarrow^4 ((e_{\mathtt{r(1(e))}}, Q_{\mathtt{r(1(e))}}), [!_{\mathtt{e}}]) = ((\overline{d}, []), [!_{\mathtt{e}}])$$

$$C'_{11} = ((a, []), [!_v; \otimes_r; ?_u]) \rightsquigarrow C'_{10} = ((\overline{\sigma(B)}, []), [!_v; \otimes_r; ?_u])$$
$$\rightsquigarrow^6 C'_4 = ((\overline{a}, []), [!_y; \parr_l; !_x]) \rightsquigarrow^4 ((\overline{d}, []), [!_{\mathtt{e}}]) = ((\overline{d}, []), [!_{\mathtt{e}}])$$

To prove that $C_{11}{}^{\varnothing} = C'_{11}{}^{\varnothing}$, we need to prove that $((\overline{a}, []), [!_{\mathtt{1(e)}}])^{/\varnothing} = ((\overline{a}, []), [!_u])^{/\varnothing}$. We can observe that, by the shape of the paths u must be equal to x. Moreover, we know that $C_4{}^{\varnothing} = C'_4{}^{\varnothing}$ so $((\overline{a}, []), [!_{\mathtt{1(e)}}])^{/\varnothing} = ((\overline{a}, []), [!_x])^{/\varnothing} = ((\overline{a}, []), [!_u])^{/\varnothing}$. Thus $C_{11}{}^{\varnothing} = C'_{11}{}^{\varnothing}$.

Observing the example of Figure 3.21a, a first try to generalize $\rightsquigarrow$ would be the relation $\mathcal{R}_1$ on potential boxes defined by $(B, P)\mathcal{R}_1(C, Q)$ if and only if there exists $t \in Sig$, $T \in Tra$ and $e \in C$ such that $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q), T) \rightsquigarrow^* ((e_t, Q_t), [!_{\mathtt{e}}])$ with $Q_t = Q@R$, and all the edges in the second part of the path are in C.

The proof-net of Figure 3.21a would indeed be $\mathcal{R}_1$-stratified. However, the acyclicity of $\mathcal{R}_1$ does not entail an elementary bound (it does not satisfy the elementary stratification property). Indeed (as one can observe in Figure 3.21b) one can transform any proof-net G in a $\mathcal{R}_1$-stratified proof-net G' which reduces to G. The transformation is to add, for every box B of G, another box B' containing only an axiom and linked to B by its auxiliary door. We can observe that:

- There is no potential box C and potential P such that $(B, P)\mathcal{R}_1(C, Q)$.
- There is no potential box C and potential P' such that $(C, Q)\mathcal{R}_1(B', P')$: no path stays in (B', P') because every path leaves (B', P') by its auxiliary door.

So G' is $\mathcal{R}_1$-stratified. The problem is that, if the $\rightsquigarrow$-path beginning by $((\sigma(C), Q), [!_t])$ enters many different copies of (B, P), then the number of copies of (C, Q) depends on the number of copies of (B, P) which depends itself on the number of copies of (B', P). So $|Cop(C, Q)|$ depends on $|Cop(B', P)|$ even if $\neg(\ (C, Q)\mathcal{R}_1(B', P)\ )$.

Observing these examples, it seems that whenever there exists a path of the shape $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q), U.?_u) \rightsquigarrow^* ((e_t, Q_t), [!_{\mathtt{e}}])$, then we should have $(B, P)\mathcal{R}(C, Q)$. Unless the path has left the box (C, Q) by its principal door. The following definition precises what we mean by "leaving a potential box by its principal door" and Theorem 104, which is a strong version of Theorem 72, proves that the intuition we gave is correct.

Definition 102. *A path of the shape* $((\overline{\sigma(B)}, P), T.?_t) \mapsto^k C$, *is said to* leave (B, P, t) *if there exist* $j < k$, $e_1, \cdots, e_j \in \vec{\mathrm{E}}_G$ *and* $T' \in Tra$ *such that*

$$((\overline{\sigma(B)}, P), T.?_t) \quad \mapsto ((e_1, \_), \_) \cdots \mapsto ((e_j, \_), \_) = ((\sigma(B), P), T'.!_t)$$
$$\forall U.?_u \in Tra, \quad ((\overline{\sigma(B)}, Q), U.?_u) \quad \mapsto ((e_1, \_), \_) \cdots \mapsto ((e_j, \_), \_) = ((\sigma(B), R), V) \Rightarrow (R = Q \ \wedge \ V = \_.!_u)$$

Otherwise the path is said to stay in (B, P, t).

A path $C \mapsto^* D$ *is said to* definitely enter $(B, P, t) \in BoxSig_G$ *if there exist* $T.?_t \in Tra$ *such that* $C \mapsto^* ((\overline{\sigma(B)}, P), T.?_t) \mapsto^+ D$ *and* $((\overline{\sigma(B)}, P), T.?_t) \mapsto^* D$ *stays in* (B, P, t).

[9] Let us notice that, for any potential box (e, P), $(e, P)^{\varnothing} = (e, [\mathtt{e}; \cdots; \mathtt{e}])$. Thus $(e_t, Q_t)^{\varnothing} = (e', Q')^{\varnothing}$ means $e_t = e'$

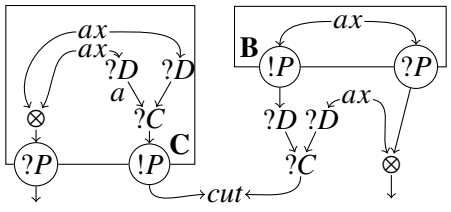


Figure 3.22: $(B,[])\dot{\rightsquigarrow}(C,[])$ but this pair is unnecessary.

In the following, we suppose defined a mapping from $BoxSig_G$ to contexts such that: for every $(B,P,t) \in BoxSig_G$, $C_{B,P,t}$ is of the shape $((\_,\_),[!\_])$ and $((\sigma(B),P),[!_t]) \rightsquigarrow^* C_{B,P,t}$.

Definition 103. *We define a relation $\mathcal{R}_1$ on B_G by: $B\mathcal{R}_1 D$ if there exists a path $((\sigma(B),P),[!_t]) \rightsquigarrow^* C_{B,P,t}$ which enters (D,Q,v) definitely.*

Theorem 104. *Let us consider a subset S of B_G and a path of the shape $C_k \rightsquigarrow \cdots \rightsquigarrow C_1 \rightsquigarrow C_0$ such that, if $C_k \rightsquigarrow^* C_0$ definitely enters (B,P,t) then $B \in S$. For every context C'_0 such that ${C_0}^{\mapsto s} = (C'_0)^{\mapsto s}$, there exists a path of the shape $C'_k \rightsquigarrow \cdots \rightsquigarrow C'_0$ and for $0 \leq i \leq k$, ${C_i}^{\mapsto s} = {C'_i}^{\mapsto s}$.*

Proof. We define C'_i by induction on i. C'_0 is already defined by assumption. Let us consider $1 \leq i \leq k$ and let us suppose that we defined $C'_{i-1},\cdots,C'_0$ such that for $0 \leq j < i$, ${C_j}^{\mapsto s} = {C'_j}^{\mapsto s}$. We know that $C_i \rightsquigarrow C_{i-1}$. If $C_i \rightsquigarrow_S C_{i-1}$ then, by Lemma 72, there exists a context C'_i such that $C'_i \rightsquigarrow C'_{i-1}$ and ${C_i}^{\mapsto s} = {C'_i}^{\mapsto s}$.

Else, C_{i-1} is of the shape $((\overline{\sigma(B)},P),T.?_t)$ with $B \notin S$. Because the path $C_k \rightsquigarrow^* C_1$ does not enter (B,P,t) definitely, the path $C_{i-1} \rightsquigarrow^* C_0$ leaves (B,P,t). So there exists $0 < j < i-1$, $T' \in Tra$ and a sequence of edges $e_j \cdots e_{i-2}$ such that $C_{i-1} \rightsquigarrow C_{i-2} = ((e_{i-2},\_),\_)\cdots C_j = ((\sigma(B),P),T'.!_t)$ and for every trace $U.?_u$, the existence of a path of the shape $((\overline{\sigma(B)},Q),U.?_u) \rightsquigarrow ((e_{i-2},\_),\_)\cdots \rightsquigarrow ((\sigma(B),R),V)$ implies that $R = Q$ and V is of the shape $\_.!_u$.

Let us notice that ${C_{i-1}}^{\mapsto s} = {C'_{i-1}}^{\mapsto s}$ so C'_{i-1} is of the shape $((\overline{\sigma(B)},Q),U.?_u)$ with $((\sigma(B),P),[!_t])^{\mapsto s} = ((\sigma(B),Q),[!_u])^{\mapsto s}$ (we write $((\sigma(B),R),[!_{\mathtt{e}}])$ for $((\sigma(B),P),[!_t])^{\mapsto s}$). Let us notice that for every $j \leq j' < i-1$, ${C_{j'}}^{\mapsto s} = {C'_{j'}}^{\mapsto s}$ so the edge of $C'_{j'}$ is the same as the edge of $C_{j'}$ (which is $e_{j'}$). Thus, C'_j is of the shape $((\sigma(B),Q),U'.!_u)$. Because $j-1 \geq 0$, ${C_{j-1}}^{\mapsto s} = {C'_{j-1}}^{\mapsto s}$. We will assume that $C_j \mapsto ((e,P),T'.!_t)$, crossing a *cut* node (the other cases are similar). Thus, we have $C'_{j-1} = ((e,Q),U'.!_u)$. Because ${C_{j-1}}^{\mapsto s} = {C'_{j-1}}^{\mapsto s}$, we have $((e,R),[!_t])^{/\mapsto s} = ((e,R),[!_u])^{/\mapsto s}$. Let us notice that C_i and C'_i must be of the shape $((\overline{e},P),T.?_t)$ and $((\overline{e},Q),U.?_u)$. So ${C_i}^{\mapsto s} = {C'_i}^{\mapsto s}$. □

Then, if $\mathcal{R}_1$ is acyclic on a proof-net G, one may deduce an elementary bound on W_G, the elementary function depending only on $|\mathcal{R}_1|$. The proof is done from Theorem 104 exactly as the proof of Theorem 78 from Theorem 72. However the acyclicity of $\mathcal{R}_1$ is not general enough: there are many proof-nets which can easily be proved to normalize in elementary time, but for which $\mathcal{R}_1$ is cyclic.

When t has not been used Let us consider the proof-net of Figure 3.22. Let us observe the $\rightsquigarrow$-path $((\sigma(B),[]),[!_{\mathtt{l(e)}}]) \rightsquigarrow^3 ((\overline{\sigma(C)},[]),[!_{\mathtt{l(e)}};?_{\mathtt{l(e)}}]) \rightsquigarrow^2 ((\overline{a},[\mathtt{l(e)}]),[!_{\mathtt{e}}])$, we have $(B,[])\dot{\rightsquigarrow}(C,[])$. However, one can still prove a result similar to Theorem 104, if we did not have this pair. Let us suppose that $((\sigma(B),[]),[!_{t'}]) \rightsquigarrow^3 ((\overline{\sigma(C)},[]),[!_{t'_1};?_{u'}]) \rightsquigarrow ((\overline{a},[u']),[!_{\mathtt{e}}])$ then, even if $(C,[])$ has three copies ($\mathtt{e}$, $\mathtt{l(e)}$ and $\mathtt{r(e)}$), the only possibility for u' is $\mathtt{l(e)}$.

Definition 105. *A context C is said* used *if there exists a path $((\sigma(B),P),[!_t]) \mapsto^* ((e,Q),[!_{t_1}]@T)$ with $t \neq t_1$ and $C^{\varnothing} = ((e,Q),[!_{t_1}]@T)^{\varnothing}$.*

Definition 106. *We define a relation $\mathcal{R}_2$ by $B\mathcal{R}_2 D$ if there exists a path of the shape $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(D)}, Q), [!_{t_1}]@U.?_u) \rightsquigarrow^* C_{B,P,t}$, $((\overline{\sigma(D)}, Q), [!_{t_1}]@U.?_u)$ is used and the path $((\overline{\sigma(D)}, Q), [!_{t_1}]@U.?_u) \rightsquigarrow^* C_{B,P,t}$ stays in (D, Q, u).*

Theorem 107. *Let G be a normalizing proof-net and S a subset of B_G. Let us suppose that ${C_0}^{\mapsto_S} = {C'_0}^{\mapsto_S}$, there exist paths of the shape*

$$\begin{array}{llll} ((\sigma(B), P), [!_t]) = & C_l & \mapsto^{l-k} ((\sigma(B'), Q), [!_u]) & = C_k \rightsquigarrow^k C_0 \\ ((\sigma(B), P), [!_{t'}]) = & C'_{l'} & \mapsto^{l'-k'} ((\sigma(B''), Q'), [!_{u'}]) & = C'_{k'} \rightsquigarrow^{k'} C'_0 \end{array}$$

And for every used context $((\overline{\sigma(D)}, R), V.?_v)$ in the $\rightsquigarrow$ path from C_k to C_0 we are in one of the following situations: D is in S, $((\overline{\sigma(D)}, R), V.?_v)$ is not used, or the path $((\overline{\sigma(D)}, R), V.?_v) \rightsquigarrow C_0$ leaves (D, R, v).

Then either ($k = k'$ and for every $0 \leq i \leq k$, ${C_i}^{\mapsto_S} = {C'_i}^{\mapsto_S}$) or ($C_0$ and C'_0 are not used).

Proof. For $1 \leq i \leq l$, we set $((e_i, P_i), [!_{t_i}]@T_i) = C_i$. For $1 \leq i \leq l'$, we set $((e'_i, P'_i), [!_{t'_i}]) = C'_i$. Let us consider the smallest $0 \leq j \leq \min(k, k')$ such that ${C_j}^{\mapsto_S} \neq {C'_j}^{\mapsto_S}$.

If such a j does not exist, then ${C_{\min(k,k')}}^{\mapsto_S} = {C'_{\min(k,k')}}^{\mapsto_S}$. So $C_{\min(k,k')}$ and $C'_{\min(k,k')}$ are both of the shape $((\sigma(\_), \_), [!\_])$. Because contexts of this shape have no antecedent by $\rightsquigarrow$, $k = k'$.

Else, let us notice that $j > 0$ so ${C_{j+1}}^{\mapsto_S} = {C'_{j+1}}^{\mapsto_S}$. By Theorem 104, the $\rightsquigarrow$ steps $C_j \rightsquigarrow C_{j+1}$ and $C'_j \rightsquigarrow C'_{j+1}$ are of the shape:

$$\begin{array}{llll} ((e_j, P_j), [!_{t_j}]@T_j) & \rightsquigarrow ((\overline{\sigma(D)}, R), V.?_v) = & ((e_{j+1}, P_{j+1}), [!_{t_{j+1}}]@T_{j+1}) \\ ((e'_j, P'_j), [!_{t'_j}]@T'_j) & \rightsquigarrow ((\overline{\sigma(D)}, R'), V'.?_{v'}) = & ((e_{j+1}, P'_{j+1}), [!_{t'_{j+1}}]@T'_{j+1}) \end{array}$$

with D a box which is not in S and such that $C_{j+1} \rightsquigarrow^* C_0$ stays in (D, R, v).

So, by assumption on the path from C_k to C_0, the context $((\overline{\sigma(D)}, R), V.?_v)$ is not used. We can notice that $((\overline{\sigma(D)}, R), V.?_v)^{\varnothing} = ((\overline{\sigma(D)}, R'), V'.?_{v'})^{\varnothing}$. Thus, $((\overline{\sigma(D)}, R'), V'.?_{v'})$ is not used either. For $1 \leq i \leq l - j$ and $1 \leq i' \leq l' - j$, the contexts C_{l-i} and $C'_{l'-i'}$ are not used. We can prove by a straightforward induction on i that, for $1 \leq i \leq \max(l, l') - j$, we have $((e_{l-i}, P_{l-i}), T_{l-i}) = ((e'_{l'-i}, P'_{l'-i}), T'_{l'-i})$, $t_i = u = t$ and $t'_i = u' = t'$ (indeed, crossing a $?C$ or $?N$ node upwards with a trace of the shape $[!\_]$ would "use" the context).

- Let us suppose that $l < l'$. We will prove by contradiction that C_0 and C'_0 are not used. Let us suppose that one of them is used. Let us consider the highest index j' such that either $C_{j'}$ or $C'_{j'}$ is used. Because C_j and C'_j are not used, we have $j' < j$. We can notice that ${C_{j'}}^{\mapsto_S} = {C'_{j'}}^{\mapsto_S}$ so they are both used. Let us consider $C_{j'} = C_{l-(l-j')}$ so, as we proved above, the only difference between $C_{j'}$ and $C'_{l'-(l-j')}$ is the signature on their leftmost trace element. So, by definition of used contexts, $C'_{l'-(l-j')}$ is used. Because $l < l'$, $l' - l + j' > j'$ which contradicts our hypothesis of maximality of j'.
- If we suppose that $l > l'$, we can prove similarly that C_0 and C'_0 are not used.

We can now suppose that $l = l'$. In this case, $C_{k'} = C_{l-(l'-k')}$, so $C_{k'}$ is of the shape $((\sigma(B''), Q'), [!\_])$. Such a context has no antecedent for $\rightsquigarrow$ so $k \leq k'$. We prove similarly that $k' \leq k$ so $k = k'$. And for every $0 \leq i \leq k$, $C_i = C_{l-(l'-i)}$ so ${C_i}^{\mapsto_S} = {C'_i}^{\mapsto_S}$. □

Thanks to this Lemma, we could prove that the acyclicity of $\mathcal{R}_2$ entails an elementary bound. And the acyclicity of $\mathcal{R}_2$, $\succ$ and $\prec$ entail a polynomial bound. However, we have other improvements to define. Soft Linear Logic (SLL [47]) is defined as the set of proof-nets without $?N$ nodes, and where the premises

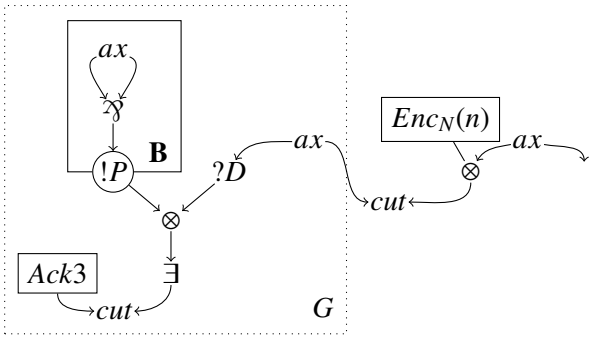


Figure 3.23: The weight of $(G)Enc_N(n)$ is linear in n.

of $?C$ nodes are either conclusions of $?C$ nodes or conclusions of $?D$ nodes. For instance, we can notice that the proof-net of Figure 3.22 is a proof-net of SLL. In our understanding, the stratification of SLL comes from Theorem 107: if $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q), [!_{t_1}]@U.?_u)$ then this path has not crossed any $?C$ node or $?N$ node, so $t = t_1$. Thus, in SLL proof-nets, $\rightsquigarrow$ and $\mathcal{R}_1$ may be cyclic, but $\mathcal{R}_2$ is always equal the empty relation $\emptyset$ (which is obviously acyclic).

Whenever the $\mathcal{R}_2$ cycle does not depend on the argument Let us name $Ack3$ the proof-net of Figure 3.12. We have $Ack3 = (Ack)Enc_N(3)$ with Ack a proof-net representing the ackermann function ack (we recall that, for $n \in \mathbb{N}$, $Enc_N(n)$ is defined in Section 3.1.4 with $Enc_N(3)$ presented in Figure 3.4). The proof-net $Ack3$ strongly normalizes to $Enc_N(ack(3))$. So W_{Ack3} is finite. Let us notice that, in the proof-net $(G)Enc_N(n)$ of Figure 3.23, there are exactly $2(ack(3)) - 1$ copies of $(B, [])$, and the only copy of the box inside $Enc_N(n)$ is $\mathtt{e}$. Thus,

$$W_{(G)Enc_N(N)} \leq W_{Ack3} + 3 \cdot (2 \cdot ack(3) - 1) + (4n + 18) \cdot 1 \leq (8 \cdot ack(3)) \cdot n + (36 \cdot ack(3))$$

So we have a bound on $W_{(G)Enc_N(n)}$ which is linear in n, even though $(G)Enc_N(N)$ is not $\mathcal{R}_2$-stratified. In terms of context semantics, $\mathcal{R}_2$ is cyclic on $(G)Enc_N(N)$. But, for every box B in a $\mathcal{R}_2$ cycle and (B, P) in $Pot(B_G)$, the number of copies of (B, P) is bounded by a number which does not depend on n.

We could define for every $k \in \mathbb{N}$, an elementary function e_k and prove that, for every proof-net G and subset S of B_G such that $\mathcal{R}_2$ is acyclic on $B_G - S$, W_G is bounded by $e_{|(\mathcal{R}_2)_{/B_G - S}|}(|\mathrm{E}_G|, M)$ with $M = \max_{(B,P) \in Pot(S)} |Cop(B, P)|$. Thus, if a proof-net G satisfies "for every box B in a $\mathcal{R}_2$ cycle and (B, P) in $Pot(B_G)$, the number of copies of (B, P) is bounded by a number which does not depend on the argument" then G normalizes in elementary time.

Whenever the $\mathcal{R}_2$ dependence is additive We will make a final improvement. Let us consider the proof-net of Figure 3.24. Its sub-proof-net G normalizes in elementary time. To guide intuition, G is an encoding of $t = \lambda m.((m)(\lambda n.((n)Succ_\lambda)1))2$ with $Succ_\lambda = \lambda n.\lambda f.\lambda x.((n)f)(f)x$ (the λ-term $Succ_\lambda$ corresponds to the proof-net $Succ$ defined in Figure 3.4c). One can observe that $\lambda n.((n)Succ_\lambda)1$ is equivalent to $\lambda n.(Succ_\lambda)n$, so t is equivalent to $\lambda m.(Succ_\lambda)(Succ_\lambda)m$. However, the proof-net does not satisfy the above criterion for elementary time. Indeed, if the proof-net G is applied to $Enc_N(k)$ with $k \geq 2$, then $|Cop(B, \mathtt{r(e)})| = k + 2$ and $B\mathcal{R}_2B$. For instance, we have:

$$((\sigma(B), [\mathtt{l(r(e))}]), [!_{\mathtt{l(r(e))}}]) \rightsquigarrow^* ((\overline{\sigma(B)}, [\mathtt{l(l(e))}]), [!_{\mathtt{r(e)}}; \otimes_l; \otimes_r; ?_{\mathtt{l(e)}}]) \rightsquigarrow^4 ((\overline{\sigma_2(C)}, [\mathtt{l(l(e))}; \mathtt{l(e)}]), [!_{\mathtt{e}}])$$

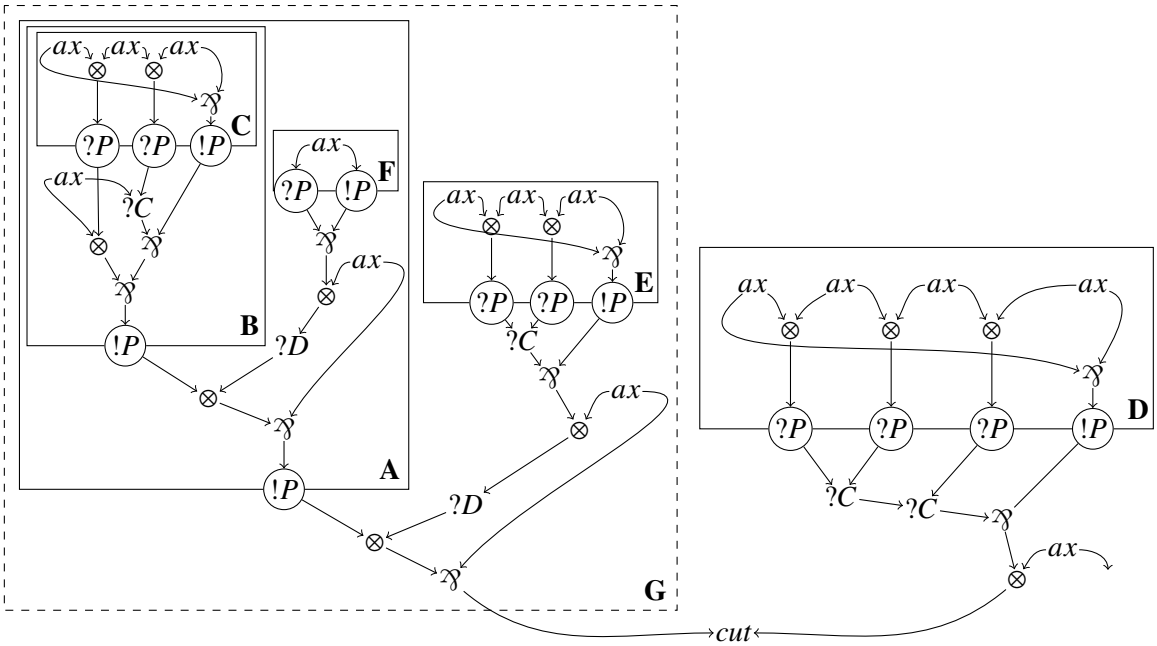


Figure 3.24: $\mathcal{R}_2$ is acyclic on this proof-net but it normalizes in linear time

The reason why (in the general case) the cycles of $\mathcal{R}_2$ can entail a non-elementary complexity is because, if $(B,P)\mathcal{R}_2(C,Q)$, the number of copies of (B,P) may depend non-additively on the number of copies of (C,Q). However, in the case of Figure 3.24, the dependence of $Cop(B,[\mathtt{l}(\mathtt{r}(\mathtt{e}))])$ on $Cop(B,[\mathtt{l}(\mathtt{l}(\mathtt{e}))])$ is additive: for every copy u of $(B,[\mathtt{l}(\mathtt{l}(\mathtt{e}))])$, there is only one maximal copy t of $(B,[\mathtt{l}(\mathtt{r}(\mathtt{e}))])$ such that there is a path of the shape

$$((\sigma(B),[\mathtt{l}(\mathtt{r}(\mathtt{e}))]),[!_t]) \rightsquigarrow^* ((\overline{\sigma(B)},[\mathtt{l}(\mathtt{l}(\mathtt{e}))]),[!_\mathtt{e};\otimes_l;\otimes_r ?_u]) \rightsquigarrow^4 ((\overline{\sigma_2(C)},[\mathtt{l}(\mathtt{l}(\mathtt{e}));u]),[!_\mathtt{e}]) = C_{B,[\mathtt{l}(\mathtt{r}(\mathtt{e}))],t}$$

Intuitively in this section we split the dependencies of $\mathcal{R}_2$ between the additive dependencies (a relation $\boxdot\!\!\rightarrow$) and the non-additive ones (a stratum $\mathfrak{s}(B)$ assigned to each box B). To be more precise: if the path starting from $((\sigma(B),P),[!_t])$ definitely enters (D,Q,u) we will have either $\mathfrak{s}(B) > \mathfrak{s}(D)$ (which intuitively corresponds to $B \rightsquigarrow\!\!> D$) or $(B,P,t) \boxdot\!\!\rightarrow (D,Q,u)$.

Let us recall that we defined (in Definition 92, page 85) the relation $\smile$ on $BoxSig_G$ by $(B,P,t) \smile (D,Q,u)$ if and only if there exists a path of the shape $((\sigma(B),P),[!_t]) \rightsquigarrow^*\!\!\hookrightarrow ((\sigma(D),Q),[!_u])$. By Lemma 53, if $(B,P,t) \smile^+ (D,Q,u)$, then for every $u' \in Cop(D,Q)$ there exists (at least one) $t' \in Cop(B,P)$ such that $((\sigma(B),P),[!_{t'}]) \mapsto^+ ((\sigma(D),Q),[!_{u'}])$. So, whenever $(B,P,t) \smile (D,Q,u)$, $Cop(B,P)$ depends on $Cop(D,Q)$. We can have $B \smile^+ B$ (with $\smile$ defined by $B \smile D$ iff $\exists P,Q \in Pot, \exists t,u \in Sig, (B,P,t) \smile (D,Q,u)$) in a proof-net G satisfying the conditions of Section 3.3: if this dependence is additive such a cycle does not lead to a non-polynomial complexity.

However, if there exist distinct paths $(B,P,t) = (B_1,P_1,t_1) \smile (B_2,P_2,t_2) \cdots (B_k,P_k,t_k) = (D,Q,u)$ and $(B,P,t') = (B'_1,P'_1,t'_1) \smile (B'_2,P'_2,t'_2) \cdots (B'_{k'},P'_{k'},t'_{k'}) = (D,Q',u')$ then we have $B \succ\!\!\rightarrow D$ (and $Cop(B,P)$ may depend non-additively on $Cop(D,Q)$). Similarly, if $(B,P) \smile (D,Q)$ and $(B,P,t) \boxdot\!\!\rightarrow (D,Q,u)$, then $Cop(B,P)$ may depend non-additively on $Cop(D,Q)$. For each box B, we will define an index $\mathfrak{d}(B)$ such that $\mathfrak{d}(B) > \mathfrak{d}(D)$ whenever the copies of B depend non-additively on the copies of D (intuitively, it corresponds to $B \succ\!\!\rightarrow D$).

3.5.1.2 Definition of weak stratifications

Definition 108. *A* weak stratification *is a tuple* $(S, \boxdot\!\!\rightarrow, \mathfrak{s}(\_), \mathfrak{d}(\_))$ *with* S *a subset of* B_G, $\boxdot\!\!\rightarrow$ *an acyclic relation on* $BoxSig_G$, *and* $\mathfrak{s}(\_)$ *and* $\mathfrak{d}(\_)$ *mappings from* B_G *to* $\mathbb{N}$, *satisfying the following conditions.*

For every $B \in S$, *we have* $\mathfrak{s}(B) = 0$. *And for every* $B \in B_G - S$, *we have* $\mathfrak{s}(B) \geq 1$. *For* $s \in \mathbb{N}$, *we define* S_s *as the set* $\{B \in B_G \mid \mathfrak{s}(B) \leq s\}$ *(in particular* $S_0 = S$*).*

For every $B \in B_G - S$ *and* $(B, P, t) \in CanCop_G$, *there exists a context* $C_{B,P,t} = ((e, Q), [!_u])$ *such that*

- $((\sigma(B), P), [!_t]) \mapsto^* C_{B,P,t}$. *For every* $((\sigma(D), R), [!_v])$ *in the path we have* $\mathfrak{d}(B) = \mathfrak{d}(D)$ *and* $\mathfrak{s}(B) \geq \mathfrak{s}(D)$. *If* $\overline{e}$ *is not an auxiliary edge then* $u = \mathtt{e}$ *and there is no path of the shape* $C_{B,P,t} \rightsquigarrow^+ ((\overline{\sigma(\_)}, \_), \_)$. *If* $\boxdot\!\!\rightarrow (B, P, t)$, *then the path* $((\sigma(B), P), [!_t]) \mapsto^* C_{B,P,t}$ *is a* $\rightsquigarrow$*-path.*

- *Let us suppose that* $((\sigma(B), P), [!_t]) \mapsto^* ((\overline{\sigma(D)}, R), V.?_v) \mapsto^* C_{B,P,t}$ *and the second part of the path does not leave* (D, R, v) *then we are in one of the following situations:*

  - $\mathfrak{s}(B) > \mathfrak{s}(D)$
  - $\mathfrak{s}(B) = \mathfrak{s}(D)$, $(B, P, t) \boxdot\!\!\rightarrow (D, R, v)$ *and the first part of the path is a* $\rightsquigarrow$*-path (it is to say that* $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(D)}, R), V.?_v)$*).*
  - $((\overline{\sigma(D)}, R), V.?_v)$ *is not used and* $\not\exists \boxdot\!\!\rightarrow (B, P, t)$.

- *Let* $(B, P, t), (B', P', t') \in CanCop_G$. *If* $(B, P, t) \boxdot\!\!\rightarrow (D, R, v)$, $(B', P', t') \boxdot\!\!\rightarrow (D, R', v')$ *then there exist paths of the shape (notice that the edges are the same in the second part of the path):*

$$\begin{array}{lll} ((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(D)}, R), V.?_v) & \rightsquigarrow ((e_1, \_), \_) \cdots & \rightsquigarrow ((e_k, \_), \_) = C_{B,P,t} \\ ((\sigma(B'), P'), [!_{t'}]) \rightsquigarrow^* ((\overline{\sigma(D)}, R'), V'.?_{v'}) & \rightsquigarrow ((e_1, \_), \_) \cdots & \rightsquigarrow ((e_k, \_), \_) = C_{B',P',t'} \end{array}$$

  and the paths do not enter boxes. Moreover, if $\overline{e_k}$ *is an auxiliary edge of a box* C, *we have* $\mathfrak{d}(B) > \mathfrak{d}(C)$ *and* $\mathfrak{s}(B) \leq \mathfrak{s}(C)$.

Let us consider a box $B \in B_G$. *Either* $\mathfrak{d}(B) = 0$ *and* $((\sigma(B), P), [!_t]) \mapsto^* ((e, Q), [!_u])$ *implies that every box containing* C *is in* S. *Or* $\mathfrak{d}(B) \geq 1$ *and we have the following conditions:*

- *If* $(B, P, t) \smile (C, Q, u)$ *or* $(B, P, t) \boxdot\!\!\rightarrow (C, Q, u)$ *then* $\mathfrak{d}(B) \geq \mathfrak{d}(C)$.

- *For* $s \in \mathbb{N}$, *if* $((\sigma(B), P), [!_t]) \mapsto^*_{S_s} ((\overline{\sigma_i(C)}, Q), [!_u])$ *and* $((\sigma(B), P), [!_{t'}]) \mapsto^*_{S_s} ((\overline{\sigma_{i'}(C)}, Q'), [!_{u'}])$ *with* $(C, Q)^{\mapsto s_s} = (C, Q')^{\mapsto s_s}$ *then either* $i = i'$ *or* $\mathfrak{d}(B) > \mathfrak{d}(C)$.

- *Let us suppose that* $(B, P, t) \smile^* (C, Q, u)$, $(B, P, t') \smile^* (D, R, v) \boxdot\!\!\rightarrow^+ (C, Q', u')$ *and* $s = \mathfrak{s}(D)$ *then we have* $(C, Q)^{\mapsto s_{s-1}} \neq (C, Q')^{\mapsto s_{s-1}}$.

The following lemma shows that the notion of weak stratification is, indeed, a generalization of $\rightsquigarrow$-stratification and $\succ\!\!\rightarrow$-stratification.

Lemma 109. *If* $\rightsquigarrow$ *and* $\succ\!\!\rightarrow$ *are acyclic, we can define a weak stratification on* G *by* $S = \emptyset$, $\boxdot\!\!\rightarrow = \emptyset$, $\mathfrak{s}(B) = s_{\rightsquigarrow}(B)$ *and* $\mathfrak{d}(B) = s_{\succ\!\!\rightarrow}(B)$.

Proof. Because $S = \emptyset$, every box is in $B_G - S$. By definition of $s_{\_}(\_)$, $s_{\rightsquigarrow}(B) \geq 1$. And, because $\rightsquigarrow$ is supposed acyclic on the finite set B_G, $s_{\rightsquigarrow}(B) \in \mathbb{N}$.

For every $(B, P, t) \in CanCop_G$, we set $C_{B,P,t}$ as the last context of the path $((\sigma(B), P), [!_t]) \rightsquigarrow^* \not\rightsquigarrow$ of the shape $((e, Q), [!_u])$. Let us suppose that $\overline{e}$ is not an auxiliary door then, there is no path of the shape

$C_{B,P,t} \mapsto^+ ((\_,\_),[!\_])$ (otherwise it would contradict the definition of $C_{B,P,t}$). In particular $u = \mathtt{e}$ (because, by supposition, $t \in Cop(B,P)$) and there is no path of the shape $C_{B,P,t} \mapsto^+ ((\overline{\sigma\_(\_)},\_),[!\_])$.

Let us suppose that $((\sigma(B),P),[!_t]) \mapsto^* ((\overline{\sigma(D)},R),V.?_v) \mapsto^* C_{B,P,t}$. Then this path is a $\rightsquigarrow$ path so $B \rightsquigarrow D$ and $\mathfrak{s}(B) > \mathfrak{s}(D)$.

For every box B we have $\mathfrak{d}(B) = s_{\succ}(B)$ which is in $\mathbb{N}$ (because $\succ$ is supposed to be acyclic) and ≥ 1 (by definition of $s\_(\_)$). If $(B,P,t) \smile (C,Q,u)$ then $s_{\succ}(B) \geq s_{\succ}(C)$. If $((\sigma(B),P),[!_t]) \mapsto^*_{S_s} ((\overline{\sigma_i(C)},Q),[!_u])$ and $((\sigma(B),P),[!_{t'}]) \mapsto^*_{S_s} ((\overline{\sigma_{i'}(C)},Q'),[!_{u'}])$ then we have $B \succ C$ so $\mathfrak{d}(B) = s_{\succ}(B) > s_{\succ}(C) = \mathfrak{d}(C)$.

Let us consider $(B,P,t) \smile^* (C,Q,u), (B,P,t') \smile^* (D,R,v) \boxdot\!\!\mapsto^+ (C,Q',u')$ with $(C,Q)^{\mapsto_{S_s}} = (C,Q')^{\mapsto_{S_s}}$. Because we supposed that $\boxdot\!\!\mapsto = \varnothing$, we have $((\sigma(B),P),[!_t]) = ((\sigma(C),Q),[!_u])$ and $((\sigma(B),P),[!_{t'}]) \mapsto^*_{S_s} ((\sigma(C),Q'),[!_{u'}])$. Thus, $((\sigma(C),Q),[!_u]) \mapsto^+_{S_s} ((\sigma(C),Q'),[!_{u'}])$. It contradicts Lemma 75. □

Defining a weak stratification in the motivating examples We can verify that every proof-nets used as motivating examples of this section have a weak stratification. For the proof-net of Figure 3.21a, we can set $S = \varnothing$, $\mathfrak{s}(B) = 1$, $\mathfrak{d}(B) = 1$ and $\boxdot\!\!\mapsto = \varnothing$. The contexts $C_{B,[],\_}$ are defined as we wrote in the introduction. For every t of the shape $\mathtt{r}(\_)$ the path $((\sigma(B),[]),[!_t]) \rightsquigarrow^* C_{B,[],t}$ enters the box $(B,[],\_)$ but leaves the box by its principal door before reaching $C_{B,[],t}$.

For the proof-net of Figure 3.22, we set $S = \varnothing$, $\mathfrak{s}(B) = \mathfrak{s}(C) = 1$, $\mathfrak{d}(B) = \mathfrak{d}(C) = 1$ and $\boxdot\!\!\mapsto = \varnothing$. We can set $C_{B,[],t}$ (resp. $C_{C,[],t}$) as the last context of the $\rightsquigarrow$ path beginning by $((\sigma(B),[]),[!_t])$ (resp. $((\sigma(C),[]),[!_t])$) this context is on the conclusion of a $?C$ node if $t = \mathtt{e}$, of a $?D$ node in the other cases. The only interesting cases to consider are the paths of the shape $((\sigma(B),[]),[!_t]) \rightsquigarrow^3 ((\overline{\sigma(C)},[]),[!_t;?_{\mathtt{l}(\mathtt{e})}]) \rightsquigarrow^* C_{B,[],t}$. These paths enter $(C,[],\mathtt{l}(\mathtt{e}))$ definitely, but $((\overline{\sigma(C)},[]),[!_t;?_{\mathtt{l}(\mathtt{e})}])$ is not used so we need neither $\mathfrak{s}(B) > \mathfrak{s}(D)$ nor $(B,[],t) \boxdot\!\!\mapsto (C,[],\mathtt{l}(\mathtt{e}))$.

For the proof-net of Figure 3.23, we set $S = B_G$. So for every $B \in B_G$, we set $\mathfrak{s}(B) = 0$ and $\mathfrak{d}(B) = 0$. If we name C the box in $Enc_N(n)$, we set $\mathfrak{s}(B) = 1$ and $\mathfrak{d}(B) = 1$. Finally we set $\boxdot\!\!\mapsto = \varnothing$.

Definining a weak stratification in Figure 3.24 The most interesting proof-net is the proof-net of Figure 3.24, because it is the only one which needs to use $\boxdot\!\!\mapsto$. We set $S = \varnothing$, $\mathfrak{s}(A) = \mathfrak{s}(D) = \mathfrak{s}(E) = 1$, $\mathfrak{s}(B) = \mathfrak{s}(C) = \mathfrak{s}(F) = 2$. We set $\mathfrak{d}(C) = \mathfrak{d}(D) = \mathfrak{d}(E) = \mathfrak{d}(F) = 1$, $\mathfrak{d}(B) = \mathfrak{d}(A) = 2$.

The only copy of $(D,[])$ is $\mathtt{e}$ and we can set $B_{D,[],\mathtt{e}} = ((\sigma(D),[]),[!_\mathtt{e}])$. The maximal copies of $(A,[])$ are $\mathtt{l}(\mathtt{l}(\mathtt{e}))$, $\mathtt{l}(\mathtt{r}(\mathtt{e}))$ and $\mathtt{r}(\mathtt{e})$. We can set $B_{A,[],t}$ as the last context of the $\rightsquigarrow$-paths beginning by $((\sigma(A),[]),[!_t])$: $B_{A,[],\mathtt{l}(\mathtt{l}(\mathtt{e}))} = ((\overline{\sigma_1(D)},[]),[!_\mathtt{e}])$, $B_{A,[],\mathtt{l}(\mathtt{r}(\mathtt{e}))} = ((\overline{\sigma_2(D)},[]),[!_\mathtt{e}])$ and $B_{A,[],\mathtt{r}(\mathtt{e})} = ((\overline{\sigma_3(D)},[]),[!_\mathtt{e}])$. Then, let us notice that the only copy of $(E,[])$ is $\mathtt{e}$. We could set $B_{E,[],\mathtt{e}}$ as the last context $((\overline{d_A},[\mathtt{l}(\mathtt{l}(\mathtt{e}))]),[!_\mathtt{e}])$ (with d_A the conclusion of the $?D$ node in A) of the $\rightsquigarrow$-path beginning by $((\sigma(E),[]),[!_\mathtt{e}])$. Let us notice that the path from $((\sigma(E),[]),[!_\mathtt{e}])$ to $((\overline{\sigma(A)},[]),[!_\mathtt{e};\parr_r;\otimes_l;?_{\mathtt{l}(\mathtt{l}(\mathtt{e}))}])$ does not cross any $?C$ or $?N$ node. But this context is nonetheless used because we have $((\sigma(B),[\mathtt{l}(\mathtt{l}(\mathtt{e}))]),[!_{\mathtt{l}(\mathtt{e})}]) \mapsto^{45} ((\overline{\sigma(A)},[]),[!_\mathtt{e};\parr_r;\otimes_l;?_{\mathtt{l}(\mathtt{l}(\mathtt{e}))}])$, so if we set $B_{E,[],\mathtt{e}} = ((\overline{d_A},[\mathtt{l}(\mathtt{l}(\mathtt{e}))]),[!_\mathtt{e}])$ we would need to set $\mathfrak{s}(E) \geq 2$. This is why we make the other choice: $B_{E,[],\mathtt{e}} = ((\sigma(E),[]),[!_\mathtt{e}])$.

Things get more complicated for the box B. To keep the description of the weak stratification relatively short we only deal with maximal canonical potentials and copies. We can observe that the maximal copies of $(B,[\mathtt{l}(\mathtt{l}(\mathtt{e}))])$ are $\mathtt{l}(\mathtt{e})$ and $\mathtt{r}(\mathtt{e})$ because of the following paths:

$$((\sigma(B),[\mathtt{l}(\mathtt{l}(\mathtt{e}))]),[!_{\mathtt{l}(t)}]) \rightsquigarrow^{23} ((\overline{\sigma_1(E)},[]),[!_t])$$
$$((\sigma(B),[\mathtt{l}(\mathtt{l}(\mathtt{e}))]),[!_{\mathtt{r}(t)}]) \rightsquigarrow^{23} ((\overline{\sigma_2(E)},[]),[!_t])$$

We set $C_{B,[\mathtt{l}(\mathtt{l}(\mathtt{e}))],\mathtt{l}(\mathtt{e})} = ((\overline{\sigma_1(E)},[]),[!_\mathtt{e}])$, $C_{B,[\mathtt{l}(\mathtt{l}(\mathtt{e}))],\mathtt{r}(\mathtt{e})} = ((\overline{\sigma_2(E)},[]),[!_\mathtt{e}])$. Those paths do not enter any box by its principal door.

To find the copies of $(B, [\mathtt{l}(\mathtt{r}(\mathtt{e}))])$, let us first observe the following path:

$$((\sigma(B), [\mathtt{l}(\mathtt{r}(\mathtt{e}))]), [!_t]) \rightsquigarrow^{50} ((\overline{\sigma(E)}, []), [!_t; \otimes_l; \otimes_r; ?_{\mathtt{e}}]) \rightsquigarrow^{29} ((\overline{\sigma(B)}, [\mathtt{l}(\mathtt{l}(\mathtt{e}))]), [!_t; \otimes_l; \otimes_r; ?_{\mathtt{r}(\mathtt{e})}])$$

If $t = \mathtt{r}(\mathtt{e})$, then the path chooses the right premise of the $?C$ node in B and the $\rightsquigarrow$ path ends on the second auxiliary edge of C. We set $C_{B,[\mathtt{l}(\mathtt{r}(\mathtt{e}))],\mathtt{r}(\mathtt{e})} = ((\overline{\sigma_2(C)}, [\mathtt{l}(\mathtt{l}(\mathtt{e})); \mathtt{r}(\mathtt{e})]), [!_{\mathtt{e}}])$. We can observe that this path enters definitely the boxes $(E, [], \mathtt{e})$, $(A, [], \mathtt{l}(\mathtt{l}(\mathtt{e})))$ and $(B, [\mathtt{l}(\mathtt{l}(\mathtt{e}))], \mathtt{r}(\mathtt{e}))$ without crossing $?C$ or $?N$ nodes but the corresponding contexts are nonetheless used. We have $\mathfrak{s}(B) = 2 > 1 = \mathfrak{s}(E) = \mathfrak{s}(A)$ so those are not problematic. On the contrary, $\mathfrak{s}(B) = \mathfrak{s}(B)$ so we have to set $(B, [\mathtt{l}(\mathtt{r}(\mathtt{e}))], \mathtt{r}(\mathtt{e})) \boxdot\!\!\rightarrow (B, [\mathtt{l}(\mathtt{l}(\mathtt{e}))], \mathtt{r}(\mathtt{e}))$. In the path described above, if we have $t = \mathtt{r}(\_)$ the $\rightsquigarrow$ path finishes in 4 steps, but if $t = \mathtt{l}(\_)$ the $\rightsquigarrow$ paths leaves $(B, [\mathtt{l}(\mathtt{l}(\mathtt{e}))], \mathtt{r}(\mathtt{e}))$ by its principal door. Let us observe the following path:

$$((\overline{\sigma(B)}, [\mathtt{l}(\mathtt{r}(\mathtt{e}))]), [\mathtt{l}(t)]) \rightsquigarrow^{79} ((\overline{\sigma(B)}, [\mathtt{l}(\mathtt{l}(\mathtt{e}))]), [!_{\mathtt{l}(t)}; \otimes_l; \otimes_r; ?_{\mathtt{r}(\mathtt{e})}]) \rightsquigarrow^{61} ((\overline{\sigma(B)}, [\mathtt{l}(\mathtt{l}(\mathtt{e}))]), [!_t; \otimes_l; \otimes_r; ?_{\mathtt{l}(\mathtt{e})}])$$

If $t = \mathtt{r}(\mathtt{e})$ then the path chooses the right premise of the $?C$ node in B and the $\rightsquigarrow$ path ends on the second auxiliary edge of C. Thus, similarly to the case $(B, [\mathtt{l}(\mathtt{r}(\mathtt{e}))], \mathtt{r}(\mathtt{e}))$, we define $C_{B,[\mathtt{l}(\mathtt{r}(\mathtt{e}))],\mathtt{l}(\mathtt{r}(\mathtt{e}))}$ as the context $((\overline{\sigma_2(C)}, [\mathtt{l}(\mathtt{l}(\mathtt{e})); \mathtt{l}(\mathtt{e})]), [!_{\mathtt{e}}])$ and we set $(B, [\mathtt{l}(\mathtt{r}(\mathtt{e}))], \mathtt{l}(\mathtt{r}(\mathtt{e}))) \boxdot\!\!\rightarrow (B, [\mathtt{l}(\mathtt{l}(\mathtt{e}))], \mathtt{l}(\mathtt{e}))$.

Finally, to find the last maximal copy of $(B, [\mathtt{l}(\mathtt{r}(\mathtt{e}))])$, let us observe the following path:

$$\begin{aligned}((\overline{\sigma(B)}, [\mathtt{l}(\mathtt{r}(\mathtt{e}))]), [\mathtt{l}(\mathtt{l}(t))]) &\rightsquigarrow^{79} ((\overline{\sigma(B)}, [\mathtt{l}(\mathtt{l}(\mathtt{e}))]), [!_{\mathtt{l}(\mathtt{l}(t))}; \otimes_l; \otimes_r; ?_{\mathtt{r}(\mathtt{e})}]) \\ &\rightsquigarrow^{61} ((\overline{\sigma(B)}, [\mathtt{l}(\mathtt{l}(\mathtt{e}))]), [!_{\mathtt{l}(t)}; \otimes_l; \otimes_r; ?_{\mathtt{l}(\mathtt{e})}]) \\ &\rightsquigarrow^{63} ((\overline{\sigma_1(F)}, [\mathtt{l}(\mathtt{l}(\mathtt{e}))]), [!_t])\end{aligned}$$

We can observe that this path enters both $(B, [\mathtt{l}(\mathtt{l}(\mathtt{e}))], \mathtt{l}(\mathtt{e}))$ and $(B, [\mathtt{l}(\mathtt{l}(\mathtt{e}))], \mathtt{r}(\mathtt{e}))$ but not definitely. Thus, this path does not require any $\boxdot\!\!\rightarrow$ pair. We define $C_{B,[\mathtt{l}(\mathtt{r}(\mathtt{e}))],\mathtt{l}(\mathtt{l}(\mathtt{e}))} = ((\overline{\sigma_1(F)}, [\mathtt{l}(\mathtt{l}(\mathtt{e}))]), [!_{\mathtt{e}}])$. There is no other copy of $(B, [\mathtt{l}(\mathtt{r}(\mathtt{e}))])$ because the only copy of $(F, [\mathtt{l}(\mathtt{l}(\mathtt{e}))])$ is $\mathtt{e}$ as can be observed with this path.

$$\begin{aligned}((\sigma(F), [\mathtt{l}(\mathtt{l}(\mathtt{e}))]), [!_{\mathtt{e}}]) &\rightsquigarrow^{58} ((\overline{\sigma_1(C)}, [\mathtt{l}(\mathtt{l}(\mathtt{e})); \mathtt{l}(\mathtt{e})]), [!_{\mathtt{e}}]) \\ &\hookrightarrow\!\rightsquigarrow^{37} ((\overline{\sigma_1(C)}, [\mathtt{l}(\mathtt{l}(\mathtt{e})); \mathtt{r}(\mathtt{e})]), [!_{\mathtt{e}}]) \\ &\hookrightarrow\!\rightsquigarrow^{82} ((\overline{d_A}, [\mathtt{l}(\mathtt{r}(\mathtt{e}))]), [!_{\mathtt{e}}]) \not\mapsto\end{aligned}$$

Because $\mathfrak{d}(F) = \mathfrak{d}(C) = 1$, we can set $C_{F,[\mathtt{l}(\mathtt{l}(\mathtt{e}))],\mathtt{e}} = C_{C,[\mathtt{l}(\mathtt{l}(\mathtt{e}));\mathtt{l}(\mathtt{e})],\mathtt{e}} = C_{C,[\mathtt{l}(\mathtt{l}(\mathtt{e}));\mathtt{r}(\mathtt{e})],\mathtt{e}} = ((\overline{d_A}, [\mathtt{l}(\mathtt{r}(\mathtt{e}))]), [!_{\mathtt{e}}])$.

Rather than detailing the (very long) paths corresponding to the copies of $(B, [\mathtt{r}(\mathtt{e})])$, we will describe the general case $(G)Enc_N(n)$ for $n \in \mathbb{N}$ (in this case D has n auxiliary doors). Every copy of (B, P) is of the shape $\mathtt{l}(\mathtt{l}(\cdots \mathtt{l}(\mathtt{r}(\mathtt{e}))))$ or $\mathtt{l}(\mathtt{l}(\cdots \mathtt{l}(\mathtt{e})))$. We write $\mathtt{l}^i(t)$ for the signature obtained by i $\mathtt{l}(\_)$ constructions on t. Intuitively, there is an order on the copies of potential boxes in this proof-net. We have $\mathtt{r}(\mathtt{e}) < \mathtt{l}(\mathtt{r}(\mathtt{e})) < \mathtt{l}(\mathtt{l}(\mathtt{r}(\mathtt{e}))) \cdots \mathtt{l}^{i-1}(\mathtt{r}(t)) < \mathtt{l}^i(t)$. Then, for $0 \leq i < |Cop(B, P)|$, we write (B, P, i) for (B, P, t_i) with t_i the i-th copy of (B, P) for the order described above. For instance, in Figure 3.24 (where we have $n = 3$) $(C, [], 0)$, $(C, [], 1)$ and $(C, [], 2)$ represent respectively $(C, [], \mathtt{r}(\mathtt{e}))$, $(C, [], \mathtt{l}(\mathtt{r}(\mathtt{e})))$ and $(C, [], \mathtt{l}(\mathtt{l}(\mathtt{e})))$. Thus $(B, [2], 0)$ and $(B, [2], 1)$ represent respectively $(C, [\mathtt{l}(\mathtt{l}(\mathtt{e}))], \mathtt{r}(\mathtt{e}))$ and $(C, [\mathtt{l}(\mathtt{l}(\mathtt{e}))], \mathtt{l}(\mathtt{e}))$.

Then, for every $j < |Cop(B, [i+1])|$, the path beginning by $((\sigma(B), [i]), [!_{\mathtt{l}^j(\mathtt{r}(\mathtt{e}))}])$ enters $(B, [i+1], 0)$ by its principal door, uses a $\mathtt{l}(\_)$ on the leftmost trace element and leaves $(B, [i+1], 0)$ by its principal door. Then, it enters $(B, [i+1], 1)$, uses a $\mathtt{l}(\_)$ and leaves $(B, [i+1], 0)$,... Eventually, the path enters $(B, [i+1], j)$ with the context $((\overline{\sigma(B)}, [i+1]), [!_{\mathtt{r}(\mathtt{e})}; \otimes_l; \otimes_r; ?_j])$. Thus, we use the $\mathtt{r}(\_)$ on the leftmost trace element and the path ends with $((\overline{\sigma_2(C)}, [i+1; j]), [!_{\mathtt{e}}])$. Thus, we set $C_{B,[i],\mathtt{l}^j(\mathtt{r}(\mathtt{e}))} = ((\overline{\sigma_2(C)}, [i+1; j]), [!_{\mathtt{e}}])$ and $(B, [i], \mathtt{l}^j(\mathtt{r}(\mathtt{e}))) \boxdot\!\!\rightarrow (B, [i+1], j)$. Thus, for every $0 \leq j < |Cop(B, [i+1])|$, $\mathtt{l}^j(\mathtt{r}(\mathtt{e}))$ is a copy of $(B, [i])$. There are $|Cop(B, [i+1])|$ such copies of $(B, [i])$.

Then, let us notice that the path beginning by $((\sigma(B),[i]),[!_{1^k(\mathtt{e})}])$ (with $k = |Cop(B,[i+1])|$) enters every $(B,[i+1],j)$ for $0 \leq j < k$, uses a $1(\_)$ each time, and leaves them. Then, the path ends on $((\overline{d_A},[i+1]),[!_{\mathtt{e}}])$. So we set $C_{B,[i],1^k(\mathtt{e})} = ((\overline{d_A},[i+1]),[!_{\mathtt{e}}])$. In total, we can notice that $(B,[i])$ has $1 + Cop(B,[i+1])$ copies, one of which does not need a $\boxdot\!\!\mapsto$ pair to satisfy the criteria of weak stratification.

3.5.1.3 Tracing back paths

We now consider a proof-net G and a weak stratification $(S, \boxdot\!\!\mapsto, \mathfrak{s}(), \mathfrak{d}())$ of G. We write $\mathfrak{s}_G$ and $\mathfrak{d}_G$ for $\max_{B\in B_G} \mathfrak{s}(G)$ and $\max_{B\in B_G} \mathfrak{d}(G)$. Similarly to the sets S_s of Section 3.2.3, for every $s \in \mathbb{N}$, we define in this section S_s as the set $\{B \in B_G \mid \mathfrak{s}(B) \leq s\}$. To simplify notations, in this section, we will write $((e,P),T)^{/s}$ for $((e,P),T)^{/\mapsto_{S_s}}$, $(e,Q)^s$ for $(e,Q)^{\mapsto_{S_s}}$, $((e,P),T)^s$ for $((e,P),T)^{\mapsto_{S_s}}$, $Cop_s(B,P)$ for $Cop_{\mapsto_{S_s}}(B,P)$ and $Can_s(B,P)$ for $Can_{\mapsto_{S_s}}(B,P)$. We write M for $\max_{(B,P)\in Pot(S)} |Cop(B,P)|$ and $\mathfrak{M}$ for $\left(|\,\mathrm{E}_G\,| \cdot M^{\partial_G}\right)^{|\,\mathrm{E}_G\,|\cdot M^{\partial_G}}$.

The goal of Section 3.5.1.3 is to prove a theorem corresponding to Lemma 82. For every canonical box (B,P), we prove a bound on the number of sequences $e_1,\cdots,e_l$ of edges such that there exists a path of the shape $((\sigma(B),P),[!_t]) \rightsquigarrow ((e_1,P_1),T_1) \rightsquigarrow ((e_2,P_2),T_2)\cdots((e_l,P_l),[!_{\mathtt{e}}])$. To prove this bound, if $s = \mathfrak{s}(B)$, we fix $(e_l,P_l)^{s-1}$ and $(B_i,P_i)^{s-1}$ for every $((\overline{\sigma\_(B_i)},P_{i-1}),[!\_]) \hookrightarrow ((\sigma(B_i),P_i),[!\_])$ step where $\mathfrak{d}(\_)$ strictly decreases (it is to say $\mathfrak{d}(B_i)$ is strictly smaller than the $\mathfrak{d}(\_)$ of the previous $\hookrightarrow$ step). Then, when those restricted potentials are fixed, we prove that it determines uniquely the sequence of edges. This result is formalized by Lemma 111. First, we need a technical lemma to handle the $\boxdot\!\!\mapsto$ relation.

Lemma 110. *Let us suppose that $\mathfrak{s}(B_k) = s > 0$, ${C_{B_0,P_0,t_0}}^{s-1} = {C_{B'_0,P'_0,t'_0}}^{s-1}$, C_{B_k,P_k,t_k} is a used context, the paths from $((\sigma(B_k),P_k),[!_{t_k}])$ to C_{B_k,P_k,t_k} and from $((\sigma(B'_{k'}),P'_{k'}),[!_{t'_{k'}}])$ to $C_{B'_{k'},P'_{k'},t'_{k'}}$ are $\rightsquigarrow$-paths, and there exist sequences of the shape:*

$$(B,P,t)\,\smile^*(B_k,P_k,t_k) \quad \boxdot\!\!\mapsto \cdots \boxdot\!\!\mapsto (B_1,P_1,t_1)\boxdot\!\!\mapsto (B_0,P_0,t_0)\not\boxdot\!\!\mapsto$$
$$(B,P,t')\,\smile^*(B'_{k'},P'_{k'},t'_{k'})\boxdot\!\!\mapsto \cdots \boxdot\!\!\mapsto (B'_1,P'_1,t'_1)\boxdot\!\!\mapsto (B'_0,P'_0,t'_0)\not\boxdot\!\!\mapsto$$

Then $k = k'$, the paths $((\sigma(B_k),P_k),[!_{t_k}]) \rightsquigarrow^* C_{B_k,P_k,t_k}$ and $((\sigma(B'_k),P'_k),[!_{t'_k}]) \rightsquigarrow^* C_{B'_k,P'_k,t'_k}$ have the same length and their contexts are pairwise equivalent for $\mapsto_{s-1}$: if we have $C_{k,i} \rightsquigarrow^i C_{B_k,P_k,t_k}$ and $C'_{k,i} \rightsquigarrow^i C'_{B'_k,P'_k,t'_k}$ then ${C_{k,i}}^{s-1} = {C'_{k,i}}^{s-1}$.

Proof. For $1 \leq j \leq k$, we write $C_{j,i}$ as the context (if it exists) such that $C_{j,i} \rightsquigarrow^i C_{B_j,P_j,t_j}$. For $1 \leq j \leq k'$, we write $C'_{j,i}$ as the context (if it exists) such that $C'_{j,i} \rightsquigarrow^i C_{B'_j,P'_j,t'_j}$. We will prove by induction on (j,i) that $C_{j,i}$ is defined if and only if $C'_{j,i}$ is defined, and ${C_{j,i}}^{s-1} = {C'_{j,i}}^{s-1}$.

First, let us notice that for $0 \leq j \leq k$, the path $((\sigma(B_j),P_j),[!_{t_j}]) \mapsto^* C_{B_j,P_j,t_j}$ is a $\rightsquigarrow$-path. Indeed, either $j = k$ and in this case the property is an assumption of the lemma. Else $(B_{j+1},P_{j+1},t_{j+1}) \boxdot\!\!\mapsto (B_j,P_j,t_j)$ so the property follows from the definition of weak stratifications.

We can also notice that, because $(B_k,P_k,t_k) \boxdot\!\!\mapsto (B_{k-1},P_{k-1},t_{k-1}) \boxdot\!\!\mapsto \cdots \boxdot\!\!\mapsto (B_0,P_0,t_0)$ we have $\mathfrak{s}(B_0) = \mathfrak{s}(B_1) = \cdots = \mathfrak{s}(B_k) = s$ (by definition of weak stratifications).

We already know by assumption that ${C_{B_0,P_0,t_0}}^{s-1} = {C_{B'_0,P'_0,t'_0}}^{s-1}$. So ${C_{0,0}}^{s-1} = {C'_{0,0}}^{s-1}$. Let us consider a context of the shape $((\overline{\sigma(D)},R),V.?_v)$ in the path $((\sigma(B_0),P_0),[!_{t_0}]) \mapsto^* C_{B_0,P_0,t_0}$ such that $((\overline{\sigma(D)},R),V.?_v) \mapsto^* C_{B_0,P_0,t_0}$ stays in (D,R,v). By definition of weak stratification we are in one of the following situations:

- Either $(B_0,P_0,t_0) \boxdot\!\!\mapsto (D,R,v)$, but this is impossible because we supposed (B_0,P_0,t_0) to be maximal for $\boxdot\!\!\mapsto$. So this case does not happen.

- Or $((\overline{\sigma(D)}, R), V.?_v)$ is not used (which implies that C_{B_0,P_0,t_0} is not used so $k \neq 0$) and $\not\emptyset\!\!\!\mapsto (B_0, P_0, t_0)$ (which implies $k = 0$). This is a contradiction so this case does not happen.

- Because we have shown that the two first cases are impossible, we are in the last possible case: it is to say that $\mathfrak{s}(D) < \mathfrak{s}(B_0) = s$. By Lemma 107, $C_{0,i}$ is defined iff $C'_{0,i}$ is defined and ${C_{0,i}}^{s-1} = {C'_{0,i}}^{s-1}$.

If $0 < j \leq \min(k, k')$, we know that $(B_j, P_j, t_j) \boxdot\!\!\!\mapsto (B_{j-1}, P_{j-1}, t_{j-1})$ and $(B'_j, P'_j, t'_j) \boxdot\!\!\!\mapsto (B'_{j-1}, P'_{j-1}, t'_{j-1})$. So, by definition of weak stratifications, we have paths of the shape:

$$((\sigma(B_j), P_j), [!_{t_j}]) \rightsquigarrow^* D_j \rightsquigarrow ((\overline{\sigma(B_{j-1})}, P_{j-1}), [!_]@V_j.?_{t_{j-1}}) \rightsquigarrow ((e_1, \_), \_) \cdots ((e_k, \_), \_) = C_{B_j,P_j,t_j}$$
$$((\sigma(B'_j), P'_j), [!_{t'_j}]) \rightsquigarrow^* D'_j \rightsquigarrow ((\overline{\sigma(B'_{j-1})}, P'_{j-1}), [!_]@V'_j.?_{t'_{j-1}}) \rightsquigarrow ((e_1, \_), \_) \cdots ((e_k, \_), \_) = C_{B'_j,P'_j,t'_j}$$

Let us notice that D_j and D'_j are of the shape $((e, P), [!_]@V_j.?_{t_{j-1}}@W_j)$ and $((e, P'), [!_]@V'_j.?_{t'_{j-1}}@W'_j)$ with $C_{j-1,1} = ((\overline{e}, P), [!_{t_{j-1}}]@\overline{W_j})$ and $C'_{j-1,1} = ((\overline{e}, P'), [!_{t'_{j-1}}]@\overline{W'_j})$. By induction hypothesis, ${C_{j-1,1}}^{s-1} = {C'_{j-1,1}}^{s-1}$. Because none of the $\overline{e_-}$ is a principal edge or auxiliary edge of a box and C_{B_j,P_j,t_j}, $C_{B'_j,P'_j,t'_j}$ are of the shape $((\_, \_), [!_])$, V_j and V'_j do not contain any $?_-$ trace element and their $!_-$ trace elements are equal. Finally, by definition of weak stratifications, ${C_{B_j,P_j,t_j}}^{s-1} = {C_{B'_j,P'_j,t'_j}}^{s-1} = \mathtt{e}$. Thus, ${D_j}^{s-1} = {D'_j}^{s-1}$. Let us consider a context of the shape $((\overline{\sigma(D)}, R), V.?_v)$ in the $\rightsquigarrow$-path from $((\sigma(B_j), P_j), [!_{t_j}])$ to C_j such that $((\overline{\sigma(D)}, R), V.?_v) \rightsquigarrow^* C_j$ stays in (D, R, v). By definition of weak stratification we are in one of the following situations:

- Either $(B_j, P_j, t_j) \boxdot\!\!\!\mapsto (D, R, v)$, but this is impossible because the path $((\overline{\sigma(D)}, R), V.?_v) \rightsquigarrow^* C_{B_j,P_j,t_j}$ enters B_{j-1} by its principal door.

- Or $((\overline{\sigma(D)}, R), V.?_v)$ is not used (which implies that C_{B_j,P_j,t_j} is not used so $k \neq j$) and $\not\emptyset\!\!\!\mapsto (B_j, P_j, t_j)$ (which implies $k = j$).

- Because we have shown that the two first cases are impossible, we are in the last possible case: $\mathfrak{s}(D) < \mathfrak{s}(B_j) = s$. By Lemma 107, $C_{j,i}$ is defined iff $C'_{j,i}$ is defined and ${C_{j,i}}^{s-1} = {C'_{j,i}}^{s-1}$.

Thus, $C_{\min(k,k'),i}$ is defined iff $C'_{\min(k,k'),i}$ is defined and ${C_{\min(k,k'),i}}^{s-1} = {C'_{\min(k,k'),i}}^{s-1}$. Let us suppose that $k < k'$, then we have $(B, P, t) \smile^* (B_k, P_k, t_k)$ and $(B, P, t') \smile^* (B'_{k'}, P'_{k'}, t'_{k'}) \boxdot\!\!\!\mapsto^+ (B'_k, P'_k, t'_k)$ with $(B_k, P_k)^{s-1} = (B'_k, P'_k)^{s-1}$, which contradicts the definition of weak stratifications. If we suppose that $k > k'$, we have a similar contradiction so $k = k'$. □

Lemma 111. *Let $s > 0$ and (B, P) be a canonical box box with $d = \mathfrak{d}(B)$. There are at most $\mathfrak{M}\left|Can_{s-1}(\vec{\mathrm{E}}_G)\right|^d$ sequences $(e_i)_{1\leq i\leq l}$ of directed edges such that, there exists a simplification of a copy of (B, P), a potential sequence $(P_i)_{1\leq i\leq l}$, and a trace sequence $(T_i)_{1\leq i<l}$ such that:*

$$((\sigma(B), P), [!_t]) \mapsto_{S_s} ((e_1, P_1), T_1) \mapsto_{S_s} \cdots \mapsto_{S_s} ((e_{l-1}, P_{l-1}), T_{l-1}) \mapsto_{S_s} ((e_l, P_l), [!_{\mathtt{e}}])$$

Proof. The proof is quite similar to the proof of Lemma 82. We prove the lemma by induction on d. We can notice that, because $\rightsquigarrow$ is injective, $e_1, \cdots, e_l$ is determined uniquely by (e_l, P_l) and the set of potential edges (e_j, P_j) such that $((e_j, P_j), T_j)$ is of the shape $((\overline{\sigma\_(\_)}, \_), [!_-])$. If $d = 0$, then there are at most $|\mathrm{E}_G| \cdot M^{\partial_G}$ such (e_j, P_j). And, by Lemma 21, they can appear only once in the path. So there are at most $\mathfrak{M}$ possibilities for the sequence $e_1, \cdots, e_l$.

Else $d > 0$. Let us suppose that $((\sigma(B), P), [!_t]) \mapsto_{S_s} ((e_1, P_1), T_1) \mapsto_{S_s} \cdots \mapsto_{S_s} ((e_{l-1}, P_{l-1}), T_{l-1}) \mapsto_{S_s} ((e_l, P_l), [!_{\mathtt{e}}])$. If there exists a context in the path of the shape $((\sigma(C), Q), [!_c])$ with $s_{\gtrdot}(C) < s_{\gtrdot}(B)$, then we set k as the smallest index such that $((e_{k+1}, P_{k+1}), T_{k+1})$ is such a context. Else, we set $k = l$.

We set $((\sigma(B_0), P_0), [!_{t_0}])$ as the last context of the path $((\sigma(B), P), [!_t]) \mapsto^+_{S_s} ((\sigma(C), Q), [!_c])$ of the shape $((\sigma(\_), \_), [!\_])$. By definition of k, we have $\mathfrak{d}(B_0) = d$. We set (D_0, Q_0, u_0) as the element of $CanCop_G$ such that $(B_0, P_0, t_0) \boxdot\!\!\mapsto^* (D_0, Q_0, u_0) \not\boxdot\!\!\mapsto$. We will prove that $(C_{D_0,Q_0,u_0})^{s-1}$ determines $e_1, \cdots, e_k$ in a unique way. Either C_{B_0,P_0,t_0} is not used, in this case $t = t_0$ so there is only one possibility for $e_1, \cdots, e_k$. We now suppose that C_{B_0,P_0,t_0} is used.

We consider a simplification t' of a copy of (B, P). We define $((e'_i, P'_i), T'_i)_{1 \leq i \leq l'}$, k', $((\sigma(B'_0), P'_0), [!_{t'_0}])$ and (D'_0, Q'_0, u'_0) in the same way we defined $((e_i, P_i), T_i)_{1 \leq i \leq l}$, k, $((\sigma(B_0), P_0), [!_{t_0}])$, and (D_0, Q_0, u_0) from (B, P, t). And we suppose that $(C_{D_0,Q_0,u_0})^{s-1} = (C_{D'_0,Q'_0,u'_0})^{s-1}$. By Lemma 110, we can deduce that the paths $((\sigma(B_0), P_0), [!_{t_0}]) \rightsquigarrow^* C_{B_0,P_0,t_0} = ((e_k, P_k), T_k)$ and $((\sigma(B'_0), P'_0), [!_{t'_0}]) \rightsquigarrow^* C_{B'_0,P'_0,t'_0} = ((e'_{k'}, P'_{k'}), T'_{k'})$ have the same length and are pairwise equivalent for $\mapsto_{S_{s-1}}$.

Because $\mathfrak{d}(B) = \mathfrak{d}(B_0)$, for every context of the shape $((\sigma(B_1), P_1), [!_{t_1}])$ in the $\mapsto$ path from $((\sigma(B), P), [!_t])$ to $((\sigma(B_0), P_0), [!_{t_0}])$:

- (B_1, P_1, t_1) is maximal for $\boxdot\!\!\mapsto$. So, if $((\sigma(B_1), P_1), [!_{t_1}]) \rightsquigarrow^* ((\overline{\sigma(C_1)}, Q_1), U_1.?_{u_1})$ with $s_{C_1}(\geq)s$ then the path $((\overline{\sigma(C_1)}, Q_1), U_1.?_{u_1}) \mapsto^* C_{B_1,P_1,t_1}$ leaves (C_1, Q_1, u_1). We can notice that this path is included in the path $((\overline{\sigma(C_1)}, Q_1), U_1.?_{u_1}) \mapsto^* C_{B_0,P_0,t_0}$.

- We make the same "choice" of auxiliary door (as in the proof of Lemma 82).

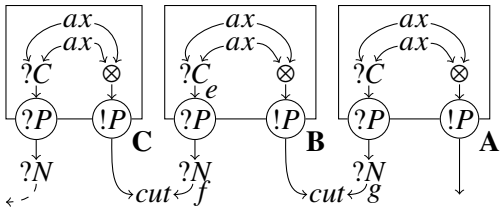
Thus, by Lemma 107, the paths $((\sigma(B), P), [!_t]) \mapsto^k ((e_k, P_k), T_k)$ and $\_ \mapsto^k ((e'_{k'}, P'_{k'}), T'_{k'})$ are pairwise equivalent for $\mapsto_{S_{s-1}}$. If $k \neq k'$, we could trace back this path infinitely, so $k = k'$. In particular the paths $((\sigma(B), P), [!_t]) \mapsto^* ((e_k, P_k), T_k)$ and $((\sigma(B), P), [!_{t'}]) \mapsto^* ((e_{k'}, P_{k'}), T_{k'})$ have the same edges: $[e_1; \cdots; e_k] = [e'_1; \cdots; e'_{k'}]$.

So the choice of ${C_{D_0,Q_0,u_0}}^{s-1}$ determines $e_1, \cdots, e_k$. There are at most $|Can_{s-1}(\vec{\mathrm{E}}_G)|$ choices for $e_1, \cdots, e_k$. And by induction hypothesis there are at most $\mathfrak{M} \cdot |Can_{s-1}(\vec{\mathrm{E}}_G)|^{d-1}$ choices for $e_{k+1}, \cdots, e_l$. In total, there are at most $\mathfrak{M} \cdot |Can_{s-1}(\vec{\mathrm{E}}_G)|^d$ possibilities for $e_1, \cdots, e_l$. □

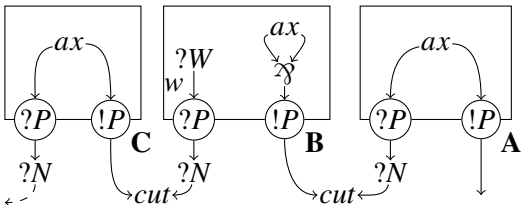
3.5.2 Improved nesting condition

In this section, we will present an improved nesting condition (a condition preventing exponential blow-ups by chains of the type of Figure 3.25a). From the point of view of context semantics, a nesting condition is a criterion such that, if we know all the edges of the path $((\sigma(B), P), [!_t]) \mapsto^* D$, we need a "bounded" amount of information on D to know t. In Section 3.3.2, we used Figure 3.25a to motivate the definition of $\prec\!\!\!\triangleleft$. We need to have $C \prec\!\!\!\triangleleft B$ because $|Cop(C, [])|$ depends non-additively on $|Cop(B, [])|$. Indeed for every $t \in Cop(B, [])$, $\mathtt{n}(\mathtt{l}(\mathtt{e}), t)$ and $\mathtt{n}(\mathtt{r}(\mathtt{e}), t)$ are in $Cop(C, [])$.

However, we can notice that we also have $C \prec\!\!\!\triangleleft B$ in Figure3.25b but it is not necessary. Indeed the set of copies of C is exactly the set of signatures $\mathtt{n}(\mathtt{e}, t)$ with $t \in Cop(B, [])$. So the dependence of $|Cop(C, [])|$ on $|Cop(B, [])|$ is additive. As a comparison, in Figure 3.14b we have $C \curvearrowright B$, the set of copies of $(C, [])$ is exactly the set of signatures $\mathtt{l}(t)$ with $t \in Cop(B, [])$ and we did not require $C \succ B$ (or $\mathfrak{d}(C) > \mathfrak{d}(B)$) because this dependence is additive.



(a) This proof-net (if extended to n boxes) reduces in $O(2^n)$ reduction steps.



(b) This proof-net (if extended to n boxes) reduces in $O(n)$ reduction steps.

Figure 3.25: Motivation for the definition of $\prec\!\!\!\triangleleft$ for the nesting condition.

Definition 112. *Let (B, P) and (C, Q) be potential boxes, then*

$$(B,P) \dot{\prec\!\!\!\triangleleft} (C,Q) \Leftrightarrow \exists t,u \in Cop(B,P), v,w \in Sig \begin{cases} t \neq u, t \sqsubset v, u \sqsubset v \text{ and,} \\ ((\sigma(B),P),[!_v]) \mapsto^* ((\sigma(C),Q),[!_w]) \\ w \text{ is standard} \end{cases}$$

$$B \prec\!\!\!\triangleleft C \Leftrightarrow \exists P,Q,(B,P) \dot{\prec\!\!\!\triangleleft} (C,Q)$$

For example, in Figure 3.25a, we have $(B,[]) \dot{\prec\!\!\!\triangleleft} (A,[])$ because $\mathtt{p(e)}$ is a strict simplification of both $\mathtt{n(r(e),e)}$ and $\mathtt{n(l(e),e)}$ and $((\sigma(B),[]),[!_{\mathtt{p(e)}}]) \mapsto^3 ((\sigma(A),[]),[!_{\mathtt{e}}])$. Similarly, in Figure 3.18, we can notice that $(B,[\mathtt{l(l(e))}]) \dot{\prec\!\!\!\triangleleft} (B,[\mathtt{l(r(e))}])$ because $((\sigma(B),[\mathtt{l(l(e))}]),[!_{\mathtt{p(e)}}]) \mapsto^{30} ((\sigma(B),[\mathtt{l(r(e))}]),[!_{\mathtt{e}}])$. Let us notice that, in Figure 3.25b, we do not have $(B,[]) \dot{\prec\!\!\!\triangleleft} (A,[])$. Indeed, even if we have $((\sigma(B),[]),[!_{\mathtt{p(e)}}]) \mapsto^3 ((\sigma(A),[]),[!_{\mathtt{e}}])$, there is only one copy of $(B,[])$ which has $\mathtt{p(e)}$ as a strict simplification.

For every $(B,P,t) \in CanCop_G$, we define a subset $Repr(B,P,t)$ of the simplifications of t. Intuitively, it is the set of simplifications u of t containing enough information to deduce t. For instance, in the proof-net of Figure 3.25b, $\mathtt{p(e)} \in Repr(B,[],\mathtt{n(e,e)})$ because there is only one copy t of $(B,[])$ such that $t \sqsubseteq \mathtt{p(e)}$. On the contrary, in the proof-net of Figure 3.25a, $\mathtt{p(e)} \notin Repr(B,[],\mathtt{n(e,e)})$ because there are three copies t of $(B,[])$ such that $t \sqsubseteq \mathtt{p(e)}$. This intuition is formalized by Lemma 115.

Definition 113. *Let (B,P) be a potential box and t a standard signature. We define $Repr(B,P,t)$ as the set of simplifications of u such that, for every simplification v of t and paths of the shape*

$$((\sigma(B),P),[!_u]) \mapsto ((e_1,\_),\_) \cdots ((e_k,Q),\_) \not\rightsquigarrow ((\sigma(B),P),[!_v]) \mapsto ((e_1,\_),\_) \cdots ((e_k,\_),\_) \rightsquigarrow$$

Then $\overline{e_k}$ is an auxiliary edge of a box C, with $s \dot{\prec\!\!\!\triangleleft} (C,Q) \geq s \dot{\prec\!\!\!\triangleleft} (B,P)$.

For instance, in the proof-net of Figure 3.25a we have $s \dot{\prec\!\!\!\triangleleft} (C,[]) = 3$, $s \dot{\prec\!\!\!\triangleleft} (B,[]) = 2$ and $s \dot{\prec\!\!\!\triangleleft} (A,[]) = 1$. Let us set $t = \mathtt{n(l(e),n(r(e),e))}$. The simplifications of t are t, $\mathtt{p(n(r(e),e))}$ and $\mathtt{p(p(e))}$. Only t is in $Repr(C,[],t)$ as proved below.

- $t \in Repr(C,[],t)$ because there is no path of the shape $((\sigma(C),[]),[!_t]) \mapsto^* ((\overline{\sigma\_(\_)},\_),[!\_])$.
- $\mathtt{p(n(r(e),e))} \notin Repr(C,[],t)$ because $s \dot{\prec\!\!\!\triangleleft} (B,[]) < s \dot{\prec\!\!\!\triangleleft} (C,[])$ and we have:

$$\begin{aligned} &((\sigma(C),[]),[!_{\mathtt{p(n(r(e),e))}}]) &&\mapsto^2 ((\overline{\sigma_1(B)},[]),[!_{\mathtt{n(r(e),e)}}]) \\ &((\sigma(C),[]),[!_t])) &&\mapsto^2 ((\overline{\sigma_1(B)},[]),[!_{\mathtt{l(e)}};!_{\mathtt{n(r(e),e)}}]) \end{aligned}$$

- Similarly, we can observe that $\mathtt{p}(\mathtt{p}(\mathtt{e})) \notin Repr(C,[],t)$ because $s_{\dashrightarrow\lhd}(B,[]) < s_{\dashrightarrow\lhd}(C,[])$ and we have:

$$\begin{aligned}((\sigma(C),[]),[!_{\mathtt{p}(\mathtt{p}(\mathtt{e}))}])) &\mapsto^2 ((\overline{\sigma_1(B)},[]),[!_{\mathtt{p}(\mathtt{e})}])\\ ((\sigma(C),[]),[!_t])) &\mapsto^2 ((\overline{\sigma_1(B)},[]),[!_{\mathtt{l}(\mathtt{e})};!_{\mathtt{n}(\mathtt{r}(\mathtt{e}),\mathtt{e})}])\end{aligned}$$

Now, let us observe the proof-net of Figure 3.25b. We have $s_{\dashrightarrow\lhd}(C,[]) = s_{\dashrightarrow\lhd}(C,[]) = s_{\dashrightarrow\lhd}(A,[]) = 1$. Let us set $t = \mathtt{n}(\mathtt{e},\mathtt{n}(\mathtt{e},\mathtt{e}))$. The simplifications of t are t, $\mathtt{p}(\mathtt{n}(\mathtt{e},\mathtt{e}))$ and $\mathtt{p}(\mathtt{p}(\mathtt{e}))$. They are all in $Repr(C,[],t)$.

- $t \in Repr(C,[],t)$ because there is no path of the shape $((\sigma(C),[]),[!_t]) \mapsto^* ((\overline{\sigma_\_(\_)},\_),[!\_])$.
- $\mathtt{p}(\mathtt{n}(\mathtt{e},\mathtt{e})) \in Repr(C,[],t)$: the only pair of paths satisfying the conditions of Definition 113 is:

$$\begin{aligned}((\sigma(C),[]),[\mathtt{n}(\mathtt{e},\mathtt{n}(\mathtt{e},\mathtt{e}))]) &\mapsto^2 ((\overline{\sigma_1(B)},[]),[!_{\mathtt{e}};!_{\mathtt{n}(\mathtt{e},\mathtt{e})}])\\ ((\sigma(C),[]),[\mathtt{p}(\mathtt{n}(\mathtt{e},\mathtt{e}))]) &\mapsto^2 ((\overline{\sigma_1(B)},[]),[!_{\mathtt{n}(\mathtt{e},\mathtt{e})}])\end{aligned}$$

  And we have $s_{\dashrightarrow\lhd}(B,[]) = 1 \geq 1 = s_{\dashrightarrow\lhd}(C,[])$.

- $\mathtt{p}(\mathtt{p}(\mathtt{e})) \in Repr(C,[],t)$. Indeed, there are two pair of paths satisfying the conditions of Definition 113:

$$\begin{aligned}((\sigma(C),[]),[\mathtt{n}(\mathtt{e},\mathtt{n}(\mathtt{e},\mathtt{e}))]) &\mapsto^2 ((\overline{\sigma_1(B)},[]),[!_{\mathtt{e}};!_{\mathtt{n}(\mathtt{e},\mathtt{e})}])\\ ((\sigma(C),[]),[\mathtt{p}(\mathtt{p}(\mathtt{e}))]) &\mapsto^2 ((\overline{\sigma_1(B)},[]),[!_{\mathtt{p}(\mathtt{e})}])\end{aligned}$$

  And we have $s_{\dashrightarrow\lhd}(B,[]) = 1 \geq 1 = s_{\dashrightarrow\lhd}(C,[])$.

$$\begin{aligned}((\sigma(C),[]),[\mathtt{p}(\mathtt{n}(\mathtt{e},\mathtt{e}))]) &\mapsto^5 ((\overline{\sigma_1(A)},[]),[!_{\mathtt{e}};!_{\mathtt{e}}])\\ ((\sigma(C),[]),[\mathtt{p}(\mathtt{p}(\mathtt{e}))]) &\mapsto^5 ((\overline{\sigma_1(A)},[]),[!_{\mathtt{e}}])\end{aligned}$$

  And we have $s_{\dashrightarrow\lhd}(A,[]) = 1 \geq 1 = s_{\dashrightarrow\lhd}(C,[])$.

Let us suppose that there exists a weak stratification on a $\lhd$-stratified proof-net G. Then, we could prove a bound on the number of copies of boxes of B as in the proof of Lemma 85. Let us consider $t \in Cop_s(B,P)$. By definition, there exists a path of the shape $((\sigma(B),P),[!_t]) \mapsto^*_{S_s} ((e,Q),[!_{\mathtt{e}}])$. By Lemma 111, there are at most $\mathfrak{M}\left|Can_{s-1}(\vec{\mathrm{E}}_G)\right|^d$ choices for the sequences of edges in the path (with $d = \mathfrak{d}(B)$). Then the choice of $(e,Q)^s$ determines t. We prove that the choices of $\mapsto_{S_s}$-copies we need to make are $\mapsto_{S_s}$-copies of boxes (C,R) such that $B \lhd C$. This allows to bound $Cop_{S_s}(B,P)$ by induction on $s_\lhd(B,P)$.

Let us consider the proof-net of Figure 3.25b, and its weak stratification defined by $S = \varnothing$, $\mathfrak{s}(C) = \mathfrak{s}(B) = \mathfrak{s}(A) = 1$, $\mathfrak{d}(C) = \mathfrak{d}(B) = \mathfrak{d}(A) = 1$ and $\boxdot\mapsto = \varnothing$. Then $\mathtt{n}(\mathtt{e},\mathtt{n}(\mathtt{e},\mathtt{e})) \in Cop_1(C,[])$ and there is a path $((\sigma(C),[]),[!_{\mathtt{n}(\mathtt{e},\mathtt{n}(\mathtt{e},\mathtt{e}))}]) \rightsquigarrow^3 ((\overline{w},[\mathtt{n}(\mathtt{e},\mathtt{e})]),[!_{\mathtt{e}}])$. Using the method of Lemma 111, we would need a bound on $|Cop_1(B,[])|$ to deduce a bound on $|Cop_1(C,[])|$ and this is why we required $C \lhd B$.

Here we can notice that, to determine a copy t of $(C,[])$, it is enough to choose $u \in Repr(C,[],t)$ (Lemma 114). In particular, the maximum (for $\sqsubseteq$) element of $Repr(C,[],t)$ determines t. This is why, instead of considering the path beginning by $((\sigma(C),[]),[\mathtt{n}(\mathtt{e},\mathtt{n}(\mathtt{e},\mathtt{e}))])$, we consider the path beginning by $((\sigma(C),[]),[\mathtt{p}(\mathtt{p}(\mathtt{e}))])$. We have $((\sigma(C),[]),[\mathtt{p}(\mathtt{p}(\mathtt{e}))]) \rightsquigarrow^5 ((\overline{\sigma_1(A)},[]),[!_{\mathtt{e}}])$. Because $\sigma_1(A)$ is not included in any box, once we have chosen the edges of the path, it entirely determines t.

Lemma 114. *Let $((e,P),[!_{t''}]@T)$ be a copy context such that, for every $u \sqsubseteq t''$, $((\sigma(B),P),[!_{t''}]) \rightsquigarrow^k \not\mapsto$ implies that $((\sigma(B),P),[!_u]) \rightsquigarrow^k \not\mapsto$.*

Let t, t' be standard signatures such that $t \sqsubseteq t''$, $t' \sqsubseteq t''$ and $((e,P),[!_t]@T)$, $((e,P),[!_{t'}]@T)$ are copy contexts. Then $t = t'$.

Proof. We prove the lemma by induction on the length k of the longest $\rightsquigarrow$ path beginning by $((e,P),[!_{t''}]@T)$. If $k=0$, then $((e,P),[!_t]@T) \not\mapsto$ and $((e,P),[!_{t'}]@T) \not\mapsto$. Because we supposed that those contexts are copy contexts, $t = t' = \mathtt{e}$.

Most of the other cases are trivial, the only interesting case is when $t'' = \mathtt{p}(u_2'')$, $T = []$ and $((e,P),[!_{t''}]) \rightsquigarrow ((f,P),[!_{u_2''}])$ (crossing a $?N$ node upwards). Because t and t' are standard, they are of the shape $t = \mathtt{n}(u_1,u_2)$ and $t' = \mathtt{n}(u_1',u_2')$ with $u_2 \sqsubseteq u_2''$ and $u_2' \sqsubseteq u_2'$. By induction hypothesis, $u_2 = u_2'$.

According to Lemma 9, $((f,P),[!_{u_1,u_2}]) \rightsquigarrow^{k-1} ((f,P),[!_{u_1}]@\_)$. By hypothesis, of the lemma, we can deduce that $((f,P),[!_{u_1}]@\_) \not\mapsto$. Because $((e,P),[!_t])$ is a copy context, $u_1 = \mathtt{e}$. Because t and t' play symmetric roles, $u_1' = \mathtt{e}$. So $t = \mathtt{n}(\mathtt{e},u_2) = \mathtt{n}(\mathtt{e},u_2') = t'$. □

Lemma 115. *Let $t,t' \in Cop(B,P)$. If $Repr(B,P,t) \cap Repr(B,P,t') \neq \emptyset$ then $t = t'$.*

Proof. Let us consider $t'' \in Repr(B,P,t) \cap Repr(B,P,t'')$, we prove that t and t' are equal by induction on $s \curvearrowright (B,P,t'')$. If (B,P,t'') is maximal for $\curvearrowright$, we have $t = t'$ by Lemma 114.

Then, let us suppose that $(B,P,t'') \curvearrowright (C,Q,u'')$. There exists a path of the shape $((\sigma(B),P),[!_{t''}]) \rightsquigarrow ((e_1,\_),\_)\cdots \rightsquigarrow ((e_k,\_),\_) \rightsquigarrow ((\overline{\sigma_i(C)},Q),[!_{u''}])$. Because t'' is a simplification of t and t', by Lemma 9 there exist paths of the shape

$$((\sigma(B),P),[!_t]) = ((e_1,\_),\_) \rightsquigarrow ((e_2,\_),\_)\cdots \rightsquigarrow ((e_k,\_),\_) = ((\overline{\sigma_i(C)},Q),U.!_u)$$
$$((\sigma(B),P),[!_{t'}]) = ((e_1,\_),\_) \rightsquigarrow ((e_2,\_),\_)\cdots \rightsquigarrow ((e_k,\_),\_) = ((\overline{\sigma_i(C)},Q),U'.!_{u'})$$

Let us prove $u'' \in Repr(C,Q,u) \cap Repr(C,Q,u')$. Because t and t' play symmetric roles, it is enough to prove $u'' \in Repr(C,Q,u)$. Let us suppose that $u'' \notin Repr(C,Q,u)$. Then, by definition of $Repr(\_)$, there exist $w \sqsupseteq u$ and paths of the shape:

$$((\sigma(C),Q),[!_{u''}]) = ((f_1,\_),\_) \rightsquigarrow \cdots ((f_l,R),\_) \not\rightsquigarrow$$
$$((\sigma(C),Q),[!_v]) = ((f_1,\_),\_) \rightsquigarrow \cdots ((f_l,\_),\_) \rightsquigarrow$$

And, either $\overline{e_k}$ is not an auxiliary edge. Or $\overline{e_k} = \sigma_i(C)$ with $s \dot{\prec}(C,Q) < s \dot{\prec}(B,P)$. Then we can notice that there exists $v \sqsupseteq t$ such that:

$$((\sigma(B),P),[!_{t''}]) = ((e_1,\_),\_) \rightsquigarrow \cdots \rightsquigarrow ((e_k,Q),[!_{u''}]) \hookrightarrow ((f_1,Q),[!_{u''}])\cdots((f_l,R),\_) \not\rightsquigarrow$$
$$((\sigma(B),P),[!_{t'}]) = ((e_1,\_),\_) \rightsquigarrow \cdots \rightsquigarrow ((e_k,Q),[!_v]) \hookrightarrow ((f_1,Q),[!_v])\cdots((f_l,R),\_) \rightsquigarrow$$

Which entails that $t'' \notin Repr(B,P,t)$. This is a contradiction, so our hypothesis was false, $u'' \in Repr(C,Q,u)$ (and $u'' \in Repr(C,Q,u')$). By induction hypothesis, we have $u = u'$.

If $U = []$, then there is no $((\_,\_),[!_{\mathtt{p}(x)}]) \rightsquigarrow ((\_,\_),[!_x])$ step in the path from $((\sigma(B),P),[!_{t''}])$ to $((\sigma(C),Q),[!_{u''}])$. Thus, we also have $U' = []$. Thus, by injectivity of $\rightsquigarrow$, $t = t'$. Similarly, if $U' = []$ then $U = []$ and $t = t'$. So, in the remaining of the proof, we suppose that $U \neq []$ and $U' \neq []$. Thus, $t \sqsubset t''$ and $t' \sqsubset t''$. Moreover $((\overline{\sigma_i(C)},Q),U.!_u) \rightsquigarrow$ so, by definition of $Repr(B,P,t)$, $s \dot{\prec}(B,P) \leq s \dot{\prec}(C,Q)$.

Let us notice that there exist $t \sqsubseteq t_0 \sqsubseteq t''$ and $t' \sqsubseteq t_0' \sqsubseteq t''$ such that $((\sigma(B),P),[!_{t_0}]) \rightsquigarrow^* ((\sigma(B),P),[!_u])$ and $((\sigma(B),P),[!_{t_0'}]) \rightsquigarrow^* ((\sigma(B),P),[!_{u'}])$. Because $u = u'$ is standard, if we had $t \neq t'$, we would have $(B,P) \dot{\prec}(C,Q)$ (remember that we deduced $t \sqsubset t''$ and $t' \sqsubset t''$ in the previous paragraph). So $s \dot{\prec}(B,P) > s \dot{\prec}(C,Q)$. This would be a contradiction because we proved in the previous paragraph that $s \dot{\prec}(B,P) \leq s \dot{\prec}(C,Q)$. Thus, our hypothesis was false, $t = t'$. □

The following Lemma corresponds to the Lemma 85 of Section 3.3.2. Let us recall that we defined M as $\max_{(B,P)\in Pot(S)} |Cop(B,P)|$ and $\mathfrak{M}$ as $\left(|\,\mathrm{E}_G\,| \cdot M^{\partial_G}\right)^{|\,\mathrm{E}_G\,|\cdot M^{\partial_G}}$.

Lemma 116. *We prove by induction on (s, n) that for every potential box (B, P) with $n = s \dot{\prec}(B, P)$,*

$$|Cop_s(B,P)| \leq \mathfrak{M} \cdot \left|Can_{s-1}(\vec{\mathrm{E}}_G)\right|^{\mathfrak{d}_G} \cdot \left(\max_{s \dot{\prec}(C,Q)<n} |Cop_s(C,Q)|\right)^{\partial_G}$$

Proof. Let us consider $t, t' \in Cop_s(B,P)$, and u (resp. u') the maximum (for $\sqsubseteq$) element of $Repr(B,P,t)$ (resp. $Repr(B,P,t')$). Then, there exists paths of the shape:

$$\begin{array}{llllll}((\sigma(B),P),[!_u]) \mapsto_s ((e_n,P_n),[!_{t_n}]@T_1) & \mapsto_s \cdots & \mapsto_s ((e_1,P_1),[!_{t_1}]@T_1) & \mapsto_s ((e,Q),[!_{\mathtt{e}}]) \\ ((\sigma(B),P),[!_{u'}]) \mapsto_s ((e'_{n'},P'_{n'}),[!_{t'_{n'}}]@T'_{n'}) & \mapsto_s \cdots & \mapsto_s ((e'_1,P'_1),[!_{t'_1}]@T'_1) & \mapsto_s ((e',Q'),[!_{\mathtt{e}}])\end{array}$$

Let us suppose that $[e_n;\cdots;e_1] = [e'_{n'};\cdots;e'_1]$. By Lemma 111, there are at most $\mathfrak{M} \cdot \left|Can_{s-1}(\vec{\mathrm{E}}_G)\right|^{\mathfrak{d}_G}$ possible choices for $[e_n;\cdots;e_1]$.

If e is contained in the box D, $Q = Q_1@[q]@Q_2$ and $Q' = Q'_1@[q']@Q'_2$. Then we can notice that $((e,Q),[!_{\mathtt{e}}]) \subseteq ((\sigma(D),Q_1),[!_q])$ and $((e,Q'),[!_{\mathtt{e}}]) \subseteq ((\sigma(D),Q'_1),[!_{q'}])$. By Corollary 98 and Lemma 96, there exist $((\sigma(B_1),P_1),[!_{u_1}]) \supseteq ((\sigma(B),P),[!_u])$, $((\sigma(B'_1),P'_1),[!_{u'_1}]) \supseteq ((\sigma(B),P),[!_{u'}])$, $((\sigma(B_2),P_2),[!_{u_2}])$ and $((\sigma(B'_2),P'_2,[!_{u'_2}])$ such that we have: $((\sigma(B_1),P_1),[!_{u_1}]) \mapsto^* ((\sigma(B_2),P_2),[!_{u_2}]) \leftarrow\!\!\!\shortmid^* ((\sigma(D),Q_1),[!_q])$ and $((\sigma(B'_1),P'_1),[!_{u'_1}]) \mapsto^* ((\sigma(B'_2),P'_2),[!_{u'_2}]) \leftarrow\!\!\!\shortmid^* ((\sigma(D),Q'_1),[!_{q'}])$.

Let us set $N = |\max_{s \dot{\prec}(C,Q)<n} Cop_s(C,Q)|^{1+\partial(D)}$. We will prove by induction on $\partial(D)$ that there is at most $N^{1+\partial(D)}$ choices for u_1. If we suppose that this property is true for every box containing D, then there are at most $N^{\partial(D)}$ possibilities for the maximal simplification of u_1. We can suppose that the strict simplifications of u_1 and u'_1 are the same and show that there are at most N possibilities for u_1.

- If $B \neq B_1$ then we can prove that $(B_1,P_1,t_1) = (B'_1,P'_1,t'_1)$ (otherwise it would violate the acyclicity of $\subseteq$ on contexts) so $((\sigma(B_2),P_2),[!_{u_2}]) = ((\sigma(B'_2),P'_2),[!_{u'_2}])$. There is only 1 possibility for u_2.
- Otherwise, it means that $(B_1,P_1) = (B'_1,P'_1) = (B,P)$, $u_1 \sqsupset u$ and $u'_1 \sqsupset u'$. Let us recall that we have $((\sigma(B_1),P_1),[!_{u_1}]) \mapsto^* ((\sigma(B_2),P_2),[!_{u_2}])$ and $((\sigma(B'_1),P'_1),[!_{u'_1}]) \mapsto^* ((\sigma(B'_2),P'_2),[!_{u'_2}])$. Because we supposed that $[e_1;\cdots;e_n] = [e'_1;\cdots;e'_{n'}]$ and the strict simplifications of u_1 and u'_1 are the same, we have $(B_2,P_2) = (B'_2,P'_2)$. If we had $s \dot{\prec}(B_2,P_2) \geq s \dot{\prec}(B,P)$, then $u_1 \in Repr(B,P,t)$ and $u'_1 \in Repr(B,P,t')$ which contradicts the assumption of maximality of u and u'. Thus, $s \dot{\prec}(B_2,P_2) < s \dot{\prec}(B,P) = n$. Moreover, by definition of copies $u_2 \in Cop_s(B_2,P_2)$. Thus, there are at most $|Cop_s(B_2,P_2)|$ possibilities for u_2, which is inferior to N.

Thus, there are at most $\mathfrak{M} \cdot |Can(s-1)\mathrm{E}_G| \, .N^{\partial_G}$ possibilities for u. The lemma follows by Lemma 115. □

Corollary 117. *For $s \in \mathbb{N}$ and $B \in B_G$ we have:*

$$|Can_s(B)| \leq \left(\mathfrak{M} \cdot |\mathrm{E}_G| \cdot \left|Can_{s-1}(\vec{\mathrm{E}}_G)\right|^{\mathfrak{d}_G}\right)^{\partial_G{}^{2\cdot|\dot{\prec}|}}$$

Proof. Thanks to Lemma 116, we prove by induction on n, that for every $(B,P) \in Pot(B_G)$ with $n = s \dot{\prec}(B,P)$ we have:

$$|Cop_s(B,P)| \leq \left(\mathfrak{M} \cdot |\mathrm{E}_G| \cdot \left|Can_{s-1}(\vec{\mathrm{E}}_G)\right|^{\mathfrak{d}_G}\right)^{\partial_G{}^{2\cdot n-1}}$$

We can conclude because, by definition of $|\succcurlyeq|$, for every $(B,P) \in Pot(B_G)$, $s \succcurlyeq (B,P) \leq |\succcurlyeq|$. □

Corollary 118. *For $B \in B_G$ we have:*

$$|Can(B)| \leq \left(\mathfrak{M} M^{\partial_G} \cdot |\mathrm{E}_G| \cdot M^{\partial_G}\right)^{\mathfrak{d}_G^{\mathfrak{s}_G} \cdot {\partial_G}^{2 \cdot \mathfrak{s}_G \cdot |\dot{\prec}|}}$$

Proof. Let us recall that we defined M as $\max_{(B,P) \in Pot(S)} |Cop(B,P)|$. We define $a = \mathfrak{M} \cdot |\mathrm{E}_G|$ and $b = \mathfrak{d}_G \cdot {\partial_G}^{2 \cdot |\dot{\prec}|}$. Then, we define a sequence $(u_s)_{s \in \mathbb{N}}$ by $u_0 = M^{\partial_G}$ and $u_s = (a \cdot u_{s-1})^b$. By definition of M and Corollary 117, for every $s \in \mathbb{N}$, $|Can_s(B)| \leq u_s$. We can prove by induction on s that $u_s \leq (a \cdot u_0)^{b^{2 \cdot s}}$, which proves the corollary because $\mapsto_{\mathfrak{s}_G} = \mapsto$. □

Theorem 119. *Let us suppose that there exists a box-bounded subsystem Sub such that every proof-net $(G)H$[10] of G_{Sub} there exists a weak-stratification $(S, \boxdot\!\!\mapsto, \mathfrak{s}(\_), \mathfrak{d}(\_))$ of $(G)H$ such that:*

- *For every $B \in S$, if $((\sigma(B), P), [!_t]) \mapsto^* ((e, \_), \_)$ then $e \in \mathrm{E}_G$.*
- *If $(B,P) \dot{\prec} (B,Q)$ then every box containing B is in S.*

Then Sub is sound for Poly.

Proof. For every potential box (B,P) with $B \in S$, because the paths starting from $\sigma(B)$ do not leave G, $Cop(B,P)$ does not depend on H. Thus, M and $\mathfrak{M}$ do not depend on H. We also notice that $|\dot{\prec}|$ is bounded by $|B_G| + |B_H| + |Can_{\mapsto_S}(\mathrm{E}_G)| \leq |B_G| + |B_H| + |\mathrm{E}_G| \cdot M^{\partial_G}$ which does not depend on H. The rest of the proof can be done, as in the proof of Theorem 88, using Corollary 118. □

[10] Let us recall that we defined the notation $(G)H$ in Figure 3.3.

Chapter 4

Paths criteria for primitive recursive characterization

4.1 Definition and acyclicity of $\rightsquigarrow\!\!\!\!\rightarrow$

Let us recall that, in Section 3.1.3 we defined a relation $\dot{\twoheadrightarrow}$ on potential boxes by $(B,P)\dot{\twoheadrightarrow}(C,Q)$ if and only if there exists a path of the shape $((\sigma(B),P),[!_t]) \mapsto^* ((e,Q@R),T)$ with $e \in C$. We proved that the acyclicity of $\twoheadrightarrow$ (which is the projection of $\dot{\twoheadrightarrow}$ on boxes) entails an elementary bound on cut-elimination.

Dal Lago, Roversi and Vercelli defined a strategy of reduction on proof-nets called "superlazy reduction" [23]. This strategy is not complete: there exist *blocking* proof-nets which are not in normal form with respect to *cut*-elimination but are normal with respect to superlazy reduction. They prove that this strategy characterizes primitive recursion in the following meaning: superlazy reduction is computable in time bounded by a primitive recursive function, and every primitive recursive function f is representable by a proof-net G_f which does not block (superlazy reduction reduces G_f to a non-blocking proof-net). Dal Lago conjectured that, if $\dot{\twoheadrightarrow}$ is acyclic on a proof-net, this proof-net does not block. Thus, the acyclicity of $\dot{\twoheadrightarrow}$ would entail a primitive recursive bound.

On his suggestion we proved directly that the acyclicity of $\dot{\twoheadrightarrow}$ entails a primitive recursive bound. In Section 3.2, we defined a relation $\rightsquigarrow\!\!\!\!\rightarrow \subseteq \twoheadrightarrow$ whose acyclicity also entails an elementary bound. Similarly, we will prove that the acyclicity of $\rightsquigarrow\!\!\!\!\dot{\rightarrow}$ is enough to entail a primitive recursive bound. For *Poly*, we consider that our criteria may be used in the long run to certify complexity bounds for programs outside the scope of academics. For such an application, one wants to have as few false negatives as possible, thus it is interesting to have criteria as general as possible. However the primitive recursive class is generally not considered feasible and such an application is unlikely. Here, the motivations to get a criterion as general as possible were different:

- To define a type system as simple as possible based on the criterion. To ensure the acyclicity of the unnecessary large relation $\dot{\twoheadrightarrow}$, one had to put additional labels on judgements and require unnecessary constraints on boxes.
- To define a type system large enough to embed simply-typed λ-calculus.
- To define a type system based on the restriction of the quantifiers instead of a restriction on the ! and ? modalities.

We will not work directly with $\rightsquigarrow\!\!\!\!\dot{\rightarrow}$ but with a corresponding relation on $BoxSig_G$. We define relation $\rightsquigarrow\!\!\!\!\ddot{\rightarrow}$ on $BoxSig_G$ as follows:

Definition 120. *Let $(B,P,t),(C,Q,u) \in BoxSig_G$, then $(B,P,t)\rightsquigarrow\!\!\!\!\ddot{\rightarrow}(C,Q,u)$ if and only if there exists a path of the shape*

$$((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(C)},Q),[!_v]@U.?_u)$$

We can notice that $(B,P,t)\rightsquigarrow\!\!\!\!\ddot{\rightarrow}(C,Q,u)$ implies $(B,P)\rightsquigarrow\!\!\!\!\dot{\rightarrow}(C,Q)$. For example, in the proof-net of Figure 4.1, we have $(C,[],\mathtt{r}(\mathtt{l}(\mathtt{e})))\rightsquigarrow\!\!\!\!\ddot{\rightarrow}(B,[],\mathtt{r}(\mathtt{e}))$ because $((\sigma(C),[]),[!_{\mathtt{r}(\mathtt{l}(\mathtt{e}))}]) \rightsquigarrow^* ((\overline{\sigma(B)},[]),[!_{\mathtt{l}(\mathtt{e})};?_{\mathtt{r}(\mathtt{e})}])$. And $(B,[],\mathtt{l}(\mathtt{e}))\rightsquigarrow\!\!\!\!\ddot{\rightarrow}(C,[],\mathtt{l}(\mathtt{e}))$ because of the path $((\sigma(B),[]),[!_{\mathtt{l}(\mathtt{e})}]) \rightsquigarrow^* ((\overline{\sigma(C)},[]),[!_{\mathtt{e}};\otimes_r;?_{\mathtt{l}(\mathtt{e})}])$.

Although this proof-net normalizes in a constant number of steps, this proof-net is not $\rightsquigarrow\!\!\!\!\dot{\rightarrow}$-stratified because $(B,[])\rightsquigarrow\!\!\!\!\dot{\rightarrow}^2(B,[])$ and $(C,[])\rightsquigarrow\!\!\!\!\dot{\rightarrow}^2(C,[])$. Nevertheless, one can notice that $\rightsquigarrow\!\!\!\!\ddot{\rightarrow}$ is acyclic and $|\rightsquigarrow\!\!\!\!\ddot{\rightarrow}|$ does not depend on the input (it does not depend on n). This property is enough to enforce an elementary bound.

Indeed, for every $k \in \mathbb{N}$, we will define an elementary function $e_k(\_)$ such that for every proof-net G, W_G is bounded by $e_{|\rightsquigarrow\!\!\!\!\ddot{\rightarrow}|}\left(\vec{\mathrm{E}}_G\right)$ (Theorem 148). Let us notice that $BoxSig_G$ is infinite so, even when $\rightsquigarrow\!\!\!\!\ddot{\rightarrow}$ is acyclic, $|\rightsquigarrow\!\!\!\!\ddot{\rightarrow}|$ and W_G may be infinite, as illustrated by the next example.

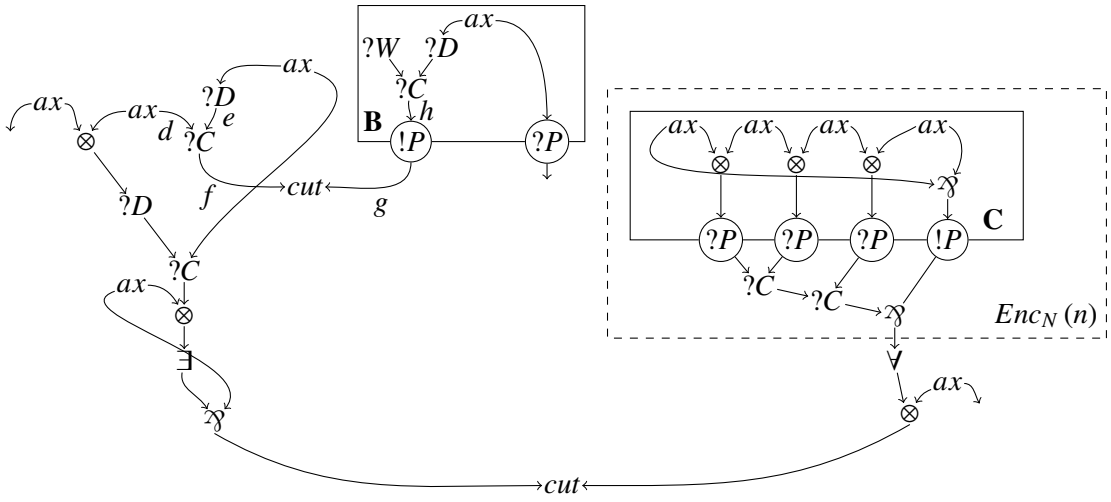


Figure 4.1: In this proof-net, represented with $n = 3$, $\rightsquigarrow$ is cyclic but $|\rightsquigarrow|$ does not depend on n.

The proof-net corresponding to $\Omega = (\lambda x.(x)x)\lambda x.(x)x$ (Figure 4.2) is not $\rightsquigarrow$-stratified because we have $((\sigma(B), []), [!_{\mathtt{l(e)}}]) \mapsto^{11} ((\overline{\sigma(B)}, []), [!_{\mathtt{e}}; \otimes_l; ?_{\mathtt{r(e)}}])$ so $(B, [], [\mathtt{l(e)}])\rightsquigarrow(B, [], [\mathtt{r(e)}])$ and $(B, [])\rightsquigarrow(B, [])$. Let us observe that $(B, [], [\mathtt{l(l(l(r(e))))}])\rightsquigarrow(B, [], [\mathtt{l(l(r(e)))}])\rightsquigarrow(B, [], [\mathtt{l(r(e))}])\rightsquigarrow(B, [], [\mathtt{r(e)}])$. This path can be extended (on its left) infinitely so $|\rightsquigarrow| = \infty$ and the bound mentioned above gives us $W_G \leq \infty$. In fact, because this proof-net does not normalize, $W_G = \infty$. However the signatures are always distinct (they are even orthogonal) and $\rightsquigarrow$ is acyclic on this proof-net.

In fact, we prove in Corollary 123 that $\rightsquigarrow$ is acyclic for every proof-net (even those which are not *LL* proof-nets). To understand why, let us notice that (B, P, t) represents a residue of a box: if $G \rightarrow^*_{cut} H$, $\pi_{G\to H}((\sigma(B_1), P_1), [!_{t_1}]) = ((\sigma(B), P), [!_t])$ and $\pi_{G\to H}((\sigma(B_2), P_2), [!_{t_2}]) = ((\sigma(B), P), [!_t])$ then $(B_1, P_1, t_1) = (B_2, P_2, t_2)$. Thus $(B, P, t)\rightsquigarrow(B, P, t)$ would mean that, along *cut* elimination, a residue B_1 of B would be cut with an edge inside B_1 (not another residue of B, the same residue) which is impossible by construction of proof-nets.

On the contrary (B, P) only represents a residue of a box B when the cuts involving $\sigma(B)$ are not reduced. Thus we may have $(B, P)\rightsquigarrow(B, P)$: let us consider $(B, P, t_1), (B, P, t_2) \in BoxSig_G$, reducing a cut involving B with a $?C$ node can "separate" the two residues. This way, the residue B_2 corresponding to (B, P, t_2) can be opened by a $?D$ node while the residue B_1 corresponding to (B, P, t_1) is preserved. Then $\sigma(B_1)$ can be cut with an edge which was inside B_2. In superlazy reduction, we only allow to reduce a cut between a box B and a $?C$ node c if the exponential tree above c (the tree of $?N$, $?C$, $?W$, $?D$ and $?P$ nodes above c) opens every residue of B (for example if there is no $?N$ node in this tree and every leaf is a $?D$, $?W$ or $?P$ node). We can notice that the proof-nets of Figures 4.1 and 4.2 block: the principal door of each box will meet a contraction node which has a premise which is the conclusion of an *ax* node (so the residue of the box on this side of the contraction is not opened). This was expected since they are not $\rightsquigarrow$-stratified.

The elementary bound mentioned above has no practical interest unless we have a bound on $|\rightsquigarrow|$. More importantly we will prove that the acyclicity of $\rightsquigarrow$ entails a primitive recursive bound on *cut* elimination (Section 4.4).

Lemma 121. *If* $(B, P, t)\rightsquigarrow(C, Q, u)$ *then* u *is a standard signature.*

Proof. By definition, there exists a path of the shape $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q), U.?_u)$. By definition of $BoxSig_G$, t is quasi-standard so $((\sigma(B), P), [!_t])$ is a quasi-standard context. By Lemma 7, $((\overline{\sigma(C)}, Q), U.?_u)$

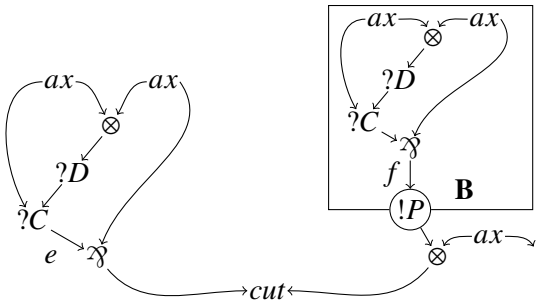


Figure 4.2: This proof net corresponds to Ω.

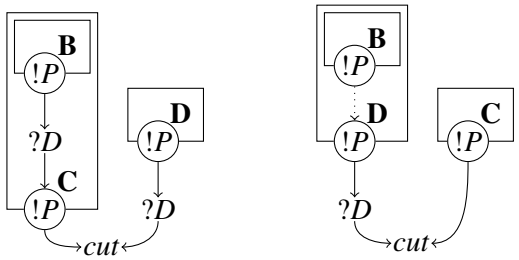


Figure 4.3: Intuition underlying Lemma 122.

is quasi-standard. Because $?_u$ is not the leftmost trace element, u is standard. □

Lemma 122. *If* $(B, P, t)\overset{\cdots}{\rightsquigarrow}^{+}(C, Q, u)$ *then there exists* $U \in Tra$ *such that:*

$$\begin{array}{lll} \textit{Either} & ((\sigma(B), P), [!_t]) & \rightsquigarrow^* ((\sigma(C), Q), U.!_u) \\ \textit{Or} & ((\sigma(B), P), [!_t]) & \rightsquigarrow^* ((\overline{\sigma(C)}, Q), U.?_u) \end{array}$$

Proof. Figure 4.3 provides an intuition for this proof. In both proof-nets we have $(D, [], \mathsf{e})\overset{\cdots}{\rightsquigarrow}(C, [], \mathsf{e})$ and we suppose that $(B, [\mathsf{e}], \mathsf{e})\overset{\cdots}{\rightsquigarrow}^{+}(D, [], \mathsf{e})$ (for the right proof-net, the figure only shows that it satisfies the induction hypothesis).

Let us suppose that $(B, P, t)\overset{\cdots}{\rightsquigarrow}^{k}(C, Q, u)$, we prove the result by induction on k. If $k = 1$, the result is trivial because, by definition of $\overset{\cdots}{\rightsquigarrow}$, there exists $U \in Tra$ such that $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q), U.?_u)$.

If $k > 1$, $(B, P, t)\overset{\cdots}{\rightsquigarrow}^{k-1}(D, R, v)\overset{\cdots}{\rightsquigarrow}(C, Q, u)$. The intuition of the remaining of the proof is shown in Figure 4.3. By definition of $\overset{\cdots}{\rightsquigarrow}$, there exists a trace U such that $((\sigma(D), R), [!_v]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q), [!_u])$. By induction hypothesis, there exists a trace V such that:

- Either $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\sigma(D), R), V.!_v)$. By Lemma 121, $((\sigma(D), R), [!_v])$ is a standard context. In this case, by Lemma 8:

$$((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\sigma(D), R), V.!_v) \rightsquigarrow^* ((\overline{\sigma(C)}, Q), (V@U).?_u)$$

- Or $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(D)}, R), V.?_v)$. By Lemma 121, $((\overline{\sigma(D)}, R), [?_v])$ is a standard context. We know that $((\sigma(D), R), [!_v]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q), U.?_u)$ so $((\overline{\sigma(C)}, Q), U.?_u)$ is standard (Lemma 8) and, by Definition of $\rightsquigarrow$, $((\sigma(C), Q), \overline{U}.!_u) \rightsquigarrow^* ((\overline{\sigma(D)}, R), [?_v])$. Let us observe that $((\sigma(C), Q), \overline{U}.!_u)$ is standard so, by Lemma 8,

$$((\sigma(C), Q), (V@\overline{U}).!_u) \rightsquigarrow^* ((\overline{\sigma(D)}, R), V.?_v)$$

  $\rightsquigarrow$ is injective, so either $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\sigma(C), Q), (V@\overline{U}).!_u)$ or $((\sigma(C), Q), (V@\overline{U}).!_u) \rightsquigarrow^* ((\sigma(B), P), [!_t])$. But the second one is ruled out because $((\sigma(B), P), [!_t])$ has no antecedent for $\rightsquigarrow$.

□

Corollary 123. *For any proof-net* G, $\overset{\cdots}{\rightsquigarrow}$ *is acyclic on* $BoxSig_G$.

Proof. Let us suppose that $(B, P, t) \in BoxSig_G$ and $(B, P, t) \ddot{\rightsquigarrow}^+ (B, P, t)$. Then, by Lemma 122, there exists $T \in Tra$ such that either $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\sigma(B), P), T.!_t)$ or $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(B)}, P), T.?_t)$. The second case is a contradiction because of Lemma 91.

So $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\sigma(B), P), T.!_t)$. Let us consider the first context of the path which is inside B with a potential of the shape $P.t@\_$. The path must enter the box B by one of its doors so there exists a trace U such that either $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma_i(B)}, P), U.!_t)$ (which is a contradiction because of Lemma 90) or $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(B)}, P), U.!_t)$ (which is a contradiction because of Lemma 91). □

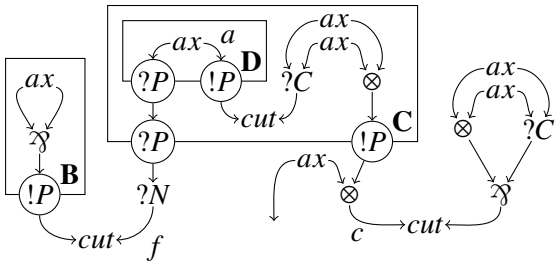


Figure 4.4: Motivation for the definition of mixable sets.

4.2 Tracing back paths

In Section 3.2, to prove that the acyclicity of $\rightsquigarrow$ entails an elementary bound, we defined relations $\mapsto_S$ and $\rightsquigarrow_S$ for every $S \subseteq B_G$. Here, to prove that the acyclicity of $\dot{\rightsquigarrow}$ entails a primitive recursive bound, we will need a refinement of those relations.

Definition 124. *Let G be a proof-net and $S \subseteq BoxSig_G$. We define $\mapsto_S$ and $\rightsquigarrow_S$ as follows:*

$$C \mapsto_S D \Leftrightarrow \begin{cases} C \mapsto D \\ \text{If } C = ((\sigma(B), P), [!_t]), \text{ then } (B, P, t) \in S \end{cases}$$

$$C \rightsquigarrow_S D \Leftrightarrow \begin{cases} C \rightsquigarrow D \\ \text{If } D = ((\overline{\sigma(B)}, P), T.?_t), \text{ then } (B, P, t) \in S \end{cases}$$

In Section 3.2, we proved Theorem 72 which states that for every $S \subseteq B_G$, whenever $C_e \rightsquigarrow_S C_f$ and ${C_f}^{\mapsto_S} = {C'_f}^{\mapsto_S}$, there exists a context C'_e such that $C'_e \rightsquigarrow_S C'_f$ and ${C_e}^{\mapsto_S} = {C'_e}^{\mapsto_S}$. We would like to prove a similar result for $S \subseteq BoxSig_G$. However, adapting the proof of Theorem 3.2 we have two issues:

- Let us suppose that C_e is on the principal edge of a box. For instance, we suppose that $C_e = ((\sigma(B), P), T.!_t) \rightsquigarrow ((\overline{f}, P), T.!_t) = C_f$ (crossing a *cut* node), $C'_f = ((\overline{f}, P'), T'.!_{t'})$ and ${C_f}^{\mapsto_S} = {C'_f}^{\mapsto_S} = ((\overline{f}, P''), T''.!_{t''})$. We may notice that $C'_e = ((\sigma(B), P'), T'.!_{t'}) \rightsquigarrow C'_f$. In the proof of Theorem 72 we had either $B \in S$ (and in this case ${C_e}^{\mapsto_S} = {C'_e}^{\mapsto_S} = ((\sigma(B), P''), \_.!_{t''})$) or $B \notin S$ (and in this case ${C_e}^{\mapsto_S} = {C'_e}^{\mapsto_S} = ((\sigma(B), P''), \_.!_{\mathtt{e}})$). However, here we may have $(B, P, t) \in S$ and $(B, P', t') \notin S$. In this case we would have ${C_e}^{\mapsto_S} = ((\sigma(B), P''), \_.!_{t''}) \neq ((\sigma(B), P''), \_.!_{\mathtt{e}}) = {C'_e}^{\mapsto_S}$.
- Similarly, if $C_f = ((\overline{\sigma(B)}, P), T.?_t)$ with $(B, P, t) \in S$ and $C'_f = ((\overline{\sigma(B)}, P'), T'.?_{t'})$ (crossing a $!P$ node upwards) we may have $(B, P', t') \notin S$. In this case, there exists no context C'_e such that $C'_e \rightsquigarrow_S C'_f$.

In Section 4.2, we deal with the first issue. And in Lemma 136 we will prove that, provided some condition on S, if $C_e \rightsquigarrow_S C_f$, $C'_e \rightsquigarrow_S C'_f$ and ${C_f}^{\mapsto_S} = {C'_f}^{\mapsto_S}$ we have ${C_e}^{\mapsto_S} = {C'_e}^{\mapsto_S}$. We will deal with the second issue in Section 4.3.

For example, let us consider the proof-net of Figure 4.4. It is exactly the same figure as Figure 3.13 used to motivate the definition of $\trianglelefteq$ because the two issues are quite similar. To motivate the definition of $\trianglelefteq$, we considered the relation $\rightarrow$ where the transition $((\sigma(D), [\mathtt{r(e)}]), [!_{\mathtt{l(e)}}]) \mapsto ((\overline{d}, [\mathtt{r(e)}]), [!_{\mathtt{l(e)}}])$ was removed. Here we set $S = \{(C, [], t) \mid t \in Sig\} \cup \{(B, [], \mathtt{n(l(e), e)}), (B, [], \mathtt{n(e, e)}), (D, [\mathtt{r(e)}], \mathtt{e}), (D, [\mathtt{e}], \mathtt{l(e)})\}$, so $((\sigma(D), [\mathtt{r(e)}]), [!_{\mathtt{l(e)}}]) \not\mapsto_S ((\overline{d}, [\mathtt{r(e)}]), [!_{\mathtt{l(e)}}])$.

Let us set $C_e = ((\sigma(B), []), [!_{\mathtt{n(l(e),r(e))}}])$, $C'_e = ((\sigma(B), []), [!_{\mathtt{n(e,r(e))}}])$, $C_f = ((\overline{f}, []), [!_{\mathtt{n(l(e),r(e))}}])$ and $C'_f = ((\overline{f}, []), [!_{\mathtt{n(e,r(e))}}])$. Then we have $C_e \rightsquigarrow_S C_f$, $C'_e \rightsquigarrow_S C'_f$ and ${C_f}^{\mapsto_S} = {C'_f}^{\mapsto_S} = ((\overline{f}, []), [!_{\mathtt{n(e,r(e))}}])$.

Indeed $C_f{}^{\mapsto s}$ is not C_f itself because $C_f \mapsto^* ((\sigma(D), [\mathtt{r}(\mathtt{e})]), [!_{\mathtt{l}(\mathtt{e})}]) \not\mapsto$. However, we can notice that $C_e{}^{\mapsto s} = ((\sigma(B), []), [!_{\mathtt{n}(\mathtt{l}(\mathtt{e}),\mathtt{e})}]) \neq ((\sigma(B), []), [!_{\mathtt{n}(\mathtt{e},\mathtt{e})}]) = C'_e{}^{\mapsto s}$. The right branch of the $\mathtt{n}(\_,\_)$ in the signatures must be $\mathtt{e}$ because $(B, [], \mathtt{n}(\mathtt{e}, \mathtt{r}(\mathtt{e})))$ and $(B, [], \mathtt{n}(\mathtt{l}(\mathtt{e}), \mathtt{r}(\mathtt{e})))$ are not in S. But, because we restrict the right branch of the signature, the $\mathtt{l}(\mathtt{e})$ in C_e can be used because $(D, [\mathtt{e}], \mathtt{l}(\mathtt{e}))$ is in S.

Intuitively the problem is that $(D, [\mathtt{r}(\mathtt{e})], \mathtt{e})$ and $(D, [\mathtt{e}], \mathtt{l}(\mathtt{e}))$ are in S but $(D, [\mathtt{r}(\mathtt{e})], \mathtt{l}(\mathtt{e}))$, obtained by mixing two elements of S, is not in S. We will prove that, if S is *mixable*, then $C_e{}^{\mapsto s} = C'_e{}^{\mapsto s}$.

Definition 125. *Let $t, u \in Sig$, we define* $\mathrm{mix}(t, u)$ *by induction as follows:* $\mathrm{mix}(\mathtt{e}, u) = u$, $\mathrm{mix}(t, \mathtt{e}) = t$, $\mathrm{mix}(\mathtt{l}(t_1), \mathtt{l}(u_1)) = \mathtt{l}(\mathrm{mix}(t_1, u_1))$, $\mathrm{mix}(\mathtt{r}(t_1), \mathtt{r}(u_1)) = \mathtt{r}(\mathrm{mix}(t_1, u_1))$, $\mathrm{mix}(\mathtt{p}(t_2), \mathtt{p}(u_2)) = \mathtt{p}(\mathrm{mix}(t_2, u_2))$ *and* $\mathrm{mix}(\mathtt{n}(t_1, t_2), \mathtt{n}(u_1, u_2)) = \mathtt{n}(\mathrm{mix}(t_1, u_1), \mathrm{mix}(t_2, u_2))$. *Else,* $\mathrm{mix}(t, u)$ *is undefined.*

We extend the definition on potentials by $\mathrm{mix}([p_1; \cdots; p_k], [q_1; \cdots; q_k]) = [\mathrm{mix}(p_1, q_1); \cdots; \mathrm{mix}(p_k, q_k)]$. *Then, we extend the definition on traces by* $\mathrm{mix}([T_1; \cdots; T_k], [U_1; \cdots; U_k]) = [M_1; \cdots; M_k]$ *with M_i defined as follows:*

- *If $T_i = !_t$ and $U_i = !_u$, we set $M_i = !_{\mathrm{mix}(t,u)}$.*
- *If $T_i = ?_t$ and $U_i = ?_u$, we set $M_i = ?_{\mathrm{mix}(t,u)}$.*
- *Else, if $T_i = U_i$, we set $M_i = T_i = U_i$.*
- *Else, M_i is undefined (so* $\mathrm{mix}([T_1; \cdots; T_k], [U_1; \cdots; U_k])$ *is undefined).*

Finally, we extend the definition on contexts by $\mathrm{mix}(((e, P), T), ((e, Q), U)) = ((e, \mathrm{mix}(P, Q)), \mathrm{mix}(T, U))$.

We are interested in restriction of copies. In this case, the signatures considered are truncations of a same signature. Lemma 126 shows that, in this case, $\mathrm{mix}(\_,\_)$ is always defined. Let us notice that, by definition, $\mathrm{mix}(\_,\_)$ is commutative so Lemma 127 also states that $u \blacktriangleleft \mathrm{mix}(t, u)$.

Lemma 126. *If $t \blacktriangleleft v$ and $u \blacktriangleleft v$ then* $\mathrm{mix}(t, u)$ *is defined and* $\mathrm{mix}(t, u) \blacktriangleleft v$.

Proof. By induction on v. If $v = \mathtt{e}$ then $t = u = \mathtt{e}$, so $\mathrm{mix}(t, u) = \mathtt{e}$. And we can verify that $\mathtt{e} \blacktriangleleft \mathtt{e}$. If $v = \mathtt{n}(v_1, v_2)$, then either $t = \mathtt{e}$ (in this case $\mathrm{mix}(t, u) = u$, and $u \blacktriangleleft v$), $u = \mathtt{e}$ (in this case $\mathrm{mix}(t, u) = t$, and $t \blacktriangleleft v$) or t and u are of the shape $\mathtt{n}(t_1, t_2)$ and $\mathtt{n}(u_1, u_2)$ with $t_1 \blacktriangleleft v_1$, $t_2 \blacktriangleleft v_2$, $u_1 \blacktriangleleft v_1$ and $u_2 \blacktriangleleft v_2$. By induction hypothesis, $m_1 = \mathrm{mix}(t_1, u_1)$ and $m_2 = \mathrm{mix}(t_2, u_2)$ are defined. Thus $\mathrm{mix}(t, u) = \mathtt{n}(m_1, m_2)$ is defined. Moreover $\mathrm{mix}(t_1, u_1) \blacktriangleleft v_1$ and $\mathrm{mix}(t_2, u_2) \blacktriangleleft v_2$ so $\mathrm{mix}(t, u) \blacktriangleleft v$.

The other cases are similar. □

Lemma 127. *If* $\mathrm{mix}(t, u)$ *is defined then $t \blacktriangleleft \mathrm{mix}(t, u)$*

Proof. Straightforward induction on t. □

Lemma 128. *If* $\mathrm{mix}(t, u) = u$ *then $t \blacktriangleleft u$.*

Proof. By induction on t. If $t = \mathtt{e}$ then $\mathtt{e} \blacktriangleleft u$. Else, because $u = \mathrm{mix}(t, u) \blacktriangleright t$, $u \neq \mathtt{e}$. We can examine every case. For instance, if $t = \mathtt{l}(t_1)$ and $\mathrm{mix}(t, u) = u = \mathtt{l}(u_1)$ with $\mathrm{mix}(t_1, u_1) = u_1$ then (by induction hypothesis) $t_1 \blacktriangleleft u_1$ so $t \blacktriangleleft u$. □

Definition 129. *Let S be a subset of $BoxSig_G$, we write that S is* mixable *if and only if*

$$\forall (B, P, t), (B, Q, u) \in S, (B, \mathrm{mix}(P, Q), \mathrm{mix}(t, u)) \in S$$
$$\forall (B, P) \in Pot(B_G), (B, P, \mathtt{e}) \in S$$

We wrote that whenever S is mixable, we can deduce that ${C_e}^{\mapsto_S} = {C'_e}^{\mapsto_S}$. And, indeed, in the previous example where we had ${C_e}^{\mapsto_S} \neq {C'_e}^{\mapsto_S}$, the set S was not mixable: $(D, [\mathtt{e}], \mathtt{l}(\mathtt{e}))$ and $(D, [\mathtt{r}(\mathtt{e})], \mathtt{e})$ are in S but $(D, [\mathtt{r}(\mathtt{e})], \mathtt{l}(\mathtt{e})) = (D, \mathrm{mix}([\mathtt{e}], [\mathtt{r}(\mathtt{e})]), \mathrm{mix}(\mathtt{l}(\mathtt{e}), \mathtt{e}))$ is not in S.

Because $C \blacktriangleleft \mathrm{mix}(C, D)$ and $D \blacktriangleleft \mathrm{mix}(C, D)$, the $\mapsto$-path beginning by $\mathrm{mix}(C, D)$ is at least as long as the paths beginning by C and D (Lemma 130). And in fact, if we only consider the paths until the leftmost trace element becomes $!_{\mathtt{e}}$, the $\mapsto$-path beginning by $\mathrm{mix}(C, D)$ is exactly as long as the longest of those two paths. Examining the paths only until they reach a context $((\_, \_), [!_{\mathtt{e}}])$ makes sense because, in Sections 3.2 and 3.3, to bound the number of copies t of a potential box (B, P) we consider the paths of the shape $((\sigma(B), P), [!_t]) \mapsto^k ((e, Q), [!_{\mathtt{e}}])$ and we could suppose k minimal: the last step uses the last constructor on the signature of the leftmost trace element..

Lemma 130. *If C and C' are canonical contexts and $C \mapsto D$ (resp. $D \mapsto C$) and $C \blacktriangleleft C'$ then there exists a canonical context D' such that $C' \mapsto D'$ (resp. $D' \mapsto C'$) and $D \blacktriangleleft D'$.*

Proof. Most of the steps are trivial. The only interesting case is whenever $C = ((\overline{e}, P), T.!_{\mathtt{e}}) \mapsto ((\overline{f}, P), T) = D$ crossing a $?D$ node upwards. Then, by definition of $\blacktriangleleft$, C' is of the shape $((\overline{e}, P'), T'.!_{t'})$ with $P \blacktriangleleft P'$, $T \blacktriangleleft T'$ and $\mathtt{e} \blacktriangleleft t'$. Because we know that C' is a canonical context, $t' = \mathtt{e}$ so if we set $D' = ((\overline{f}, P'), T')$ we have $C' \mapsto D'$ and $D \blacktriangleleft D'$. □

Lemma 131. *If $C_1 \mapsto C'_1$, $C_2 \mapsto C'_2$ and $\mathrm{mix}(C_1, C_2)$ is defined. Then $\mathrm{mix}(C_1, C_2) \mapsto \mathrm{mix}(C'_1, C'_2)$.*

Proof. Straightforward analysis of every $\mapsto$ step possible. For example, if $\overline{e}$ is the conclusion of a $?C$ node, $C_1 = ((e, P_1), [!_{t_1}]) \mapsto ((f, P_1), [!_{u_1}])$ and $((e, P_2), [!_{t_2}]) \mapsto ((g, P_2), [!_{u_2}])$. Because of those steps $t_1 \neq \mathtt{e}$ and $t_2 \neq \mathtt{e}$. Because $\mathrm{mix}(C_1, C_2) = ((e, \mathrm{mix}(P_1, P_2)), [!_{\mathrm{mix}(t_1,t_2)}])$ is defined, $\mathrm{mix}(t_1, t_2)$ is defined. So the top constructors of these signatures are the same. Either $t_1 = \mathtt{l}(u_1)$ and $t_2 = \mathtt{l}(u_2)$, and in this case f and g are both the left premise of the $?C$ node and $\mathrm{mix}(C_1, C_2) = \mathrm{mix}(C_1, C_2) = ((e, \mathrm{mix}(P_1, P_2)), [!_{\mathrm{mix}(t_1,t_2)}]) \mapsto ((f, \mathrm{mix}(P_1, P_2)), [!_{\mathrm{mix}(u_1,u_2)}])$. Or $t_1 = \mathtt{r}(u_1)$ and $t_2 = \mathtt{r}(u_2)$. □

Lemma 132. *Let C_1 and C_2 be canonical contexts. If $\mathrm{mix}(C_1, C_2) \mapsto^k ((\_, \_), [!_t]@\_)$ with $t \neq \mathtt{e}$. Then either $C_1 \mapsto^k ((\_, \_), [!_{u_1}]@\_)$ with $u_1 \neq \mathtt{e}$ or $C_2 \mapsto^k ((\_, \_), [!_{u_2}]@\_)$ with $u_2 \neq \mathtt{e}$.*

Proof. We prove the lemma by induction on k. Let us set $((e, P_1), [!_{t_1}]@T_1) = C_1$ and $((e, P_2), [!_{t_2}]@T_2) = C_2$. Then, $\mathrm{mix}(C_1, C_2) = ((e, \mathrm{mix}(P_1, P_2)), [!_{\mathrm{mix}(t_1,t_2)}]@(\mathrm{mix}(T_1, T_2)))$. Because the leftmost signature decreases along $\mapsto$ paths, $\mathrm{mix}(t_1, t_2) \neq \mathtt{e}$. So, either $t_1 \neq \mathtt{e}$ or $t_2 \neq \mathtt{e}$. This proves the lemma when $k = 0$. Now, we will suppose that $k > 0$.

If $t_1 = \mathtt{e}$ then we have $\mathrm{mix}(t_1, t_2) = t_2$. Because C_2 is a canonical context, there exists a path of the shape $C_2 \mapsto ((f_1, Q_1), [!_{u_1}]@U_1) \mapsto ((f_2, Q_2), [!_{u_2}]@U_2) \cdots \mapsto ((f_l, Q_l), [!_{\mathtt{e}}]@U_l)$. By Lemma 130, for every $1 \leq i \leq l$, $\mathrm{mix}(C_1, C_2) \mapsto^i ((f_i, Q'_i), [!_{u'_i}]@U'_i)$ with $Q_i \blacktriangleleft Q'_i$, $U_i \blacktriangleleft U'_i$ and $u_i \blacktriangleleft u'_i$. In fact, because the edges of the paths are the same, $u_i = u'_i$. In particular, $k < l$ so $C_2 \mapsto^k ((\_, \_), [!_{u_k}]@U_k)$ and $u_k = t \neq \mathtt{e}$. The case $t_2 = \mathtt{e}$ is solved similarly.

If $t_1 \neq \mathtt{e}$ and $t_2 \neq \mathtt{e}$ then we can notice that there exist contexts C'_1 and C'_2 such that $C_1 \mapsto C'_1$ and $C_2 \mapsto C'_2$. Thus, by Lemma 131, $\mathrm{mix}(C_1, C_2) \mapsto \mathrm{mix}(C'_1, C'_2) \mapsto^{k-1} ((\_, \_), [!_t]@\_)$. By induction hypothesis, either $C_1 \mapsto C'_1 \mapsto^{k-1} ((\_, \_), [!_{u_1}]@\_)$ with $u_1 \neq \mathtt{e}$ or $C_2 \mapsto C'_2 \mapsto^{k-1} ((\_, \_), [!_{u_2}]@\_)$ with $u_2 \neq \mathtt{e}$. □

Lemma 133. *Let S be a mixable subset of $BoxSig_G$. If C_1 and C_2 are $\mapsto_S$-copy contexts, then $\mathrm{mix}(C_1, C_2)$ is a $\mapsto_S$-copy context.*

Proof. Let us set $((e,P),[!_{t_1}]@T_1) = C_1$ and $((e,P_2),[!_{t_2}]@T_2) = C_2$. Then, $\mathrm{mix}(C_1,C_2) = ((e,P),[!_t]@T) = ((e,\mathrm{mix}(P_1,P_2)),[!_{\mathrm{mix}(t_1,t_2)}]@(\mathrm{mix}(T_1,T_2)))$. Let us consider $u \sqsupseteq t$. Then, there exist $u_1 \sqsupseteq t_1$ and $u_2 \sqsupseteq t_2$ such that $u = \mathrm{mix}(u_1,u_2)$. Because C_1 and C_2 are $\mapsto_S$-copy contexts, there exist $k_1,k_2 \in \mathbb{N}$ such that $((e,P_1),[!_{u_1}]@T_1) \mapsto_S^{k_1} ((\_,\_),[!_{\mathtt{e}}]@\_)$ and $((e,P_2),[!_{u_2}]@T_2) \mapsto_S^{k_2} ((\_,\_),[!_{\mathtt{e}}]@\_)$. By Lemma 130, there exists a path $((e,P),[!_u]@T) \mapsto^{\max(k_1,k_2)} ((\_,\_),[!_v]@\_)$. By Lemma 132, $v = \mathtt{e}$. To prove the lemma we have to show that all these $\mapsto$ steps are $\mapsto_S$ steps.

Let us suppose without loss of generality that $k_1 \leq k_2$. Then, for $1 \leq i < k_1$, if we name D_1 and D_2 the contexts such that $C_1 \mapsto^i D_1$ and $C_2 \mapsto^i D_2$, we have $\mathrm{mix}(C_1,C_2) \mapsto^i \mathrm{mix}(D_1,D_2)$. In particular, if $\mathrm{mix}(D_1,D_2)$ is of the shape $((\sigma(B),Q),[!_v])$, then $Q = \mathrm{mix}(Q_1,Q_2)$ and $v = \mathrm{mix}(v_1,v_2)$ with $D_1 = ((\sigma(B),Q_1),[!_{v_1}])$ and $D_2 = ((\sigma(B),Q_2),[!_{v_2}])$. Because $D_1 \mapsto_S$ and $D_2 \mapsto_S$, (B,Q_1,v_1) and (B,Q_2,v_2) are in S. We supposed that S is mixable, so $(B,\mathrm{mix}(Q_1,Q_2),\mathrm{mix}(v_1,v_2))$ is in S.

For $k_1 \leq i < k_2$, if $C \mapsto^i ((\sigma(B),Q),[!_v])$ then $C_2 \mapsto^i ((\sigma(B),Q_2),[!_v])$ with $Q_2 \blacktriangleleft Q$. Because $((\sigma(B),Q_2),[!_v]) \mapsto_S$, we know that (B,Q_2,v) is in S. We supposed that S is mixable, so $(B,Q,\mathtt{e}) \in S$. Thus, because S is mixable $(B,Q,v) = (B,\mathrm{mix}(Q,Q_2),\mathrm{mix}(\mathtt{e},v))$ is in S.

□

The motivations for the definitions of mixable sets and $\trianglelefteq$ (Definition 55, page 64) are similar (and that is the reason why we used the same proof-nets as examples). We defined $\trianglelefteq$ because there exist relations $\rightarrow$ on contexts and $(B,P,t) \in BoxSig_G$ such that the set $Restr_{\rightarrow}(((\sigma(B),P),[!_t]))$ does not have a maximum for $\blacktriangleleft$. Here, we will prove that this does not happen whenever $\rightarrow$ is of the shape $\mapsto_S$ with S a mixable set. If S is mixable, and $(B,P,t) \in BoxSig_G$, then $((\sigma(B),P),[!_t])^{/\mapsto_S}$ is the maximum element of $Restr_{\mapsto_S}(((\sigma(B),P),[!_t]))$ for the order $\blacktriangleleft$ (Lemma 134).

Lemma 134. *Let S be a mixable subset of $BoxSig_G$, and $C = ((e,P),[!_t]@T)$ be a canonical context,*

$$Restr_{\mapsto_S}(C) = \{u \blacktriangleleft C^{/\mapsto_S} \mid ((e,P),[!_u]@T) \text{ is a } \mapsto_S\text{-copy context}\}$$

Proof. Let us write t' for $C^{/\mapsto_S}$. Let us consider $u \in Restr_{\mapsto_S}(C)$. By definition of $Restr_{\mapsto_S}(\_)$, $((e,P),[!_u]@T)$ is a $\mapsto_S$-copy context. Let us notice that $t' \blacktriangleleft t$ and $u \blacktriangleleft t$, so $\mathrm{mix}(t',u)$ is defined and $\mathrm{mix}(t',u) \blacktriangleleft t$ (Lemma 126). By Lemma 133, $((e,P),[!_{\mathrm{mix}(t',u)}]@T)$ is a $\mapsto_S$-copy context. By definition of $Restr_{\mapsto_S}(\_)$, $\mathrm{mix}(t',u) \in Restr_{\mapsto_S}(C)$. Moreover, $t' \blacktriangleleft \mathrm{mix}(t',u)$ (Lemma 127) so $t' \trianglelefteq \mathrm{mix}(t',u)$ (Lemma 56). However t' is the maximum element of $Restr_{\mapsto_S}(C)$ for $\trianglelefteq$ so $\mathrm{mix}(t',u) = t'$. By Lemma 128, $u \blacktriangleleft t'$.

Let us consider $u \blacktriangleleft t'$ such that $((e,P),[!_u]@T)$ is a $\mapsto_S$-copy context. Let us notice that $t' \blacktriangleleft t$. Then, by transitivity of $\blacktriangleleft$, we have $u \blacktriangleleft t$. So, by definition of $Restr\_(\_)$, u is in $Restr_{\mapsto_S}(C)$. □

Lemma 135. *Let S be a mixable subset of $BoxSig_G$, $((e,P),[!_t]@T)$ be a canonical context and $t' \in Sig$ such that $((e,P),[!_t]@T)^{/\mapsto_S} = ((e,P),[!_{t'}]@T)^{/\mapsto_S} = t''$. For $Q \blacktriangleleft P$ and $U \blacktriangleleft T$,*

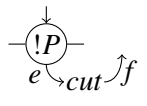
$$((e,Q),[!_t]@U)^{/\mapsto_S} = ((e,Q),[!_{t'}]@U)^{/\mapsto_S} \blacktriangleleft t''$$

Proof. Let us set $u = ((e,Q),[!_t]@U)^{/\mapsto_S}$ and $u' = ((e,Q),[!_{t'}]@U)^{/\mapsto_S}$. By definition of $\mathrm{mix}(\_,\_)$, we can notice that $((e,\mathrm{mix}(P,Q)),[!_{\mathrm{mix}(\mathtt{e},u)}]@\,\mathrm{mix}(T,U)) = ((e,P),[!_u]@T)$ so, by Lemma 133, $((e,P),[!_u]@T)$ is a $\mapsto_S$-copy context. Moreover, because $u \in Restr_{\mapsto_S}(((e,Q),[!_t]@U))$, we know that $u \blacktriangleleft t$, so $u \in Restr_{\mapsto_S}(((e,P),[!_t]@T))$. By Lemma 134, $u \blacktriangleleft t''$. Because t and t' play symmetric roles, $u' \blacktriangleleft t''$.

Thus, $Restr_{\mapsto_S}(((e,Q),[!_t]@U))$ and $Restr_{\mapsto_S}(((e,Q),[!_{t'}]@U))$ are both equal to the set of $v \blacktriangleleft t''$ such that $Restr_{\mapsto_S}(((e,Q),[!_v]@U))$ is a $\mapsto_s$-copy context. In particular, their maximums for $\trianglelefteq$ are the same: it is to say $u = u'$. □

Lemma 136. *For any proof-net G and mixable subset S of $BoxSig_G$, let C_e, C_f, C'_e and C'_f be contexts such that $C_e \rightsquigarrow_S C_f$, $C'_e \rightsquigarrow_S C'_f$ and $C_f{}^{\mapsto_S} = C'_f{}^{\mapsto_S}$, then $C_e{}^{\mapsto_S} = C'_e{}^{\mapsto_S}$.*

Proof. Let S' be the projection of S on B_G: $S' = \{B \in B_G \mid \exists P \in Pot, t \in Sig, (B,P,t) \in S\}$. The proof is quite similar to the proof of Theorem 72. The only important cases, are the cases where $\mapsto_S$ differs from $\mapsto_{S'}$: whenever the path enters or leaves a box by its principal door.



In the case where e is the principal edge of a box B (we consider the case where we cross a *cut*) then $C_e = ((e,P),T.!_t) \rightsquigarrow_S ((f,P),T.!_t) = C_f$. So C'_f is of the shape $((f,P'),T'.!_{t'})$. We set $C'_e = ((e,P'),T'.!_{t'})$. By supposition, $C_f{}^{\mapsto_S} = C'_f{}^{\mapsto_S} = ((f,P''),T''.!_{t''})$. In particular $((f,P''),[!_t])^{/\mapsto_s} = ((f,P''),[!_{t'}])^{/\mapsto_s} = t''$.

$$Restr_{\mapsto_S}((e,P''),[!_t])) = \{\mathsf{e}\} \cup \{u \in Restr_{\mapsto_S}(((f,P''),[!_t])) \mid \forall v \sqsupseteq u, (B,P'',v) \in S\}$$

$$Restr_{\mapsto_S}(((e,P''),[!_t])) = \{\mathsf{e}\} \cup \left\{u \blacktriangleleft ((f,P''),[!_t])^{/\mapsto_S} \;\middle|\; \begin{array}{c} \forall v \sqsupseteq u, (B,P'',v) \in S \\ ((e,P''),[!_u]@T) \text{ is a } \mapsto_S \text{-copy context} \end{array}\right\}$$

$$Restr_{\mapsto_S}(((e,P''),[!_t])) = \{\mathsf{e}\} \cup \left\{u \blacktriangleleft ((f,P''),[!_{t'}])^{/\mapsto_S} \;\middle|\; \begin{array}{c} \forall v \sqsupseteq u, (B,P'',v) \in S \\ ((e,P''),[!_u]@T) \text{ is a } \mapsto_S \text{-copy context} \end{array}\right\}$$

$$Restr_{\mapsto_S}((e,P''),[!_t])) = \{\mathsf{e}\} \cup \{u \in Restr_{\mapsto_S}(((f,P''),[!_{t'}])) \mid \forall v \sqsupseteq u, (B,P'',v) \in S\}$$

$$Restr_{\mapsto_S}((e,P''),[!_t])) = Restr_{\mapsto_S}((e,P''),[!_{t'}]))$$

So $((e,P''),[!_t])^{/\mapsto_S} = ((e,P''),[!_{t'}])^{/\mapsto_S} = u''$, and $u'' \blacktriangleleft t''$. Let $[T_k;\cdots;T_1] = T$, $[T'_k;\cdots;T'_1] = T'$ and $[T''_k;\cdots;T''_1] = T''$, let us prove by induction on i that $((e,P),[T_i;\cdots;T_1;!_t])^{\mapsto_S} = ((e,P'),[T'_i;\cdots;T'_1;!_{t'}])^{\mapsto_S} = ((e,P''),[U''_i;\cdots;U''_1;!_{u''}])$ with $[U''_i;\cdots;U''_1] \blacktriangleleft [T''_i;\cdots;T''_1]$. For $i = 0$, the result is straightforward because we know that $(e,P)^{\mapsto_S} = (e,P')^{\mapsto_S} = (e,P'')$ (Lemma 65) and $((e,P''),[!_t])^{/\mapsto_S} = ((e,P''),[!_{t'}])^{/\mapsto_S} = u''$. For $i > 0$, we use Lemma 135. Let us notice that those steps were not detailed in the proof of Theorem 72. Out of pedagogical concern, we decided do introduce the technical points progressively.

In the case where $\overline{f}$ is the principal edge of a box B (we consider the case where we cross a *cut*) then $C_e = ((e,P),T.?_t) \rightsquigarrow_S ((f,P),T.?_t) = C_f$. So C'_f is of the shape $((f,P'),T'.?_{t'})$. We set $C'_e = ((e,P'),T'.?_{t'})$. By supposition, $C_f{}^{\mapsto_S} = C'_f{}^{\mapsto_S} = ((f,P''),T''.?_{t''})$. By definition of $\rightsquigarrow_S$, (B,P) and (B,P') are in S. So $C_e{}^{\mapsto_S} = C'_e{}^{\mapsto_S} = ((e,P''),T''.?_{t''})$. □

Corollary 137. *For any proof-net G and mixable subset S of $BoxSig_G$. Let us suppose that $C_e \mapsto C_f$, $C'_e \mapsto C'_f$, $C_e{}^{\mapsto_S} = C'_e{}^{\mapsto_S}$. Finally we suppose that, if C_e or C'_e is of the shape $((\sigma(B),P),T.!_t)$ with T a list of trace elements, then $(B,P,t) \in S$. Then, $C_f{}^{\mapsto_S} = C'_f{}^{\mapsto_S}$.*

Proof. Either $C_e \rightsquigarrow C_f$. In this case we also have $C'_e \rightsquigarrow C'_f$. Let us notice that $(C'_e)^{\perp\mapsto_S} = (C'_e{}^{\mapsto_S})^{\perp} = (C_e{}^{\mapsto_S})^{\perp} = (C_e)^{\perp\mapsto_S}$. By definition of $\rightsquigarrow$, $(C_f)^{\perp} \rightsquigarrow (C_e)^{\perp}$ and $(C'_f)^{\perp} \rightsquigarrow (C'_e)^{\perp}$. Moreover, by assumption on C_e and C'_e, we have $(C_f)^{\perp} \rightsquigarrow_S (C_e)^{\perp}$ and $(C'_f)^{\perp} \rightsquigarrow_S (C'_e)^{\perp}$. By Lemma 136, we have $(C^{\perp}_f)^{\mapsto_S} = (C'_f)^{\perp\mapsto_S}$. Which gives us $(C_f{}^{\mapsto_S})^{\perp} = (C'_f{}^{\mapsto_S})^{\perp}$ and, finally, $C_f{}^{\mapsto_S} = C'_f{}^{\mapsto_S}$.

Else, $C_e = ((\overline{\sigma_i(B)},P),[!_t]) \hookrightarrow ((\sigma(B),P),[!_t]) = C_f$. Because $C_e{}^{\mapsto_S} = C'_e{}^{\mapsto_S}$, the context C'_e is of the shape $((\overline{\sigma_i(B)},P'),[!_{t'}])$. Thus C'_f is of the shape $((\sigma(B),P'),[!_{t'}])$. We can notice that $\sigma(B)$ and $\sigma_i(B)$ are contained in the same boxes so, by Lemmas 59 and 65, $C_f{}^{\mapsto_S} = C'_f{}^{\mapsto_S}$. □

Let us notice that we had to suppose that $C_e \rightsquigarrow_S C_f$ *and* $C'_e \rightsquigarrow_S C'_f$ while in Theorem 72, we only need to suppose one of those. To get a formulation similar to Theorem 72, we will need to make further assumptions on S.

4.3 Definition of S_n

By analogy with Section 3.2, one might want to define S_n as the set of $(B,P,t) \in BoxSig_G$ such that $s_{\rightsquigarrow}(B,P,t) \leq n$. However, with such a definition, S_n would not be mixable and we could not use Lemma 136.

To understand the problem, let us sketch an attempt of proof of mixability of $\{(B,P,t) \mid s_{\rightsquigarrow}(B,P,t) \leq n\}$. We prove it by contraposition: let us set $P = \mathrm{mix}(P_1,P_2)$ and $t = \mathrm{mix}(t_1,t_2)$ and let us suppose that $s_{\rightsquigarrow}(B,\mathrm{mix}(P_1,P_2),\mathrm{mix}(t_1,t_2)) > n$, we need to show that either $s_{\rightsquigarrow}(B,P_1,t_1) > n$ or $s_{\rightsquigarrow}(B,P_2,t_2) > n$. Because $s_{\rightsquigarrow}(B,P,t)$, we have $(B,P,t)\rightsquigarrow\!\!\!\!\rightarrow\!\!\!\!>(C,Q,u)$ with $s_{\rightsquigarrow}(C,Q,u) \geq n$. By definition, there exists a path of the shape

$$((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(C)},Q),[!_{t'}]@U.?_u)$$

We would like to prove that there is such a path beginning by $((\sigma(B),P_1),[!_{t_1}])$ or $((\sigma(B),P_2),[!_{t_2}])$. Among the lemmas we proved previously, Lemma 132 gives a very similar result, but it requires $t' \neq \mathtt{e}$ which is not always the case with $\rightsquigarrow\!\!\!\!\rightarrow\!\!\!\!>$.

This is why we define new relations $\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>$ and $\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>$ which are respectively included in $\rightsquigarrow\!\!\!\!\rightarrow\!\!\!\!>$ and $\rightsquigarrow\!\!\!\!\rightarrow\!\!\!\!>$. In particular $\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>$ is acyclic because $\rightsquigarrow\!\!\!\!\rightarrow\!\!\!\!>$ is acyclic (Corollary 123). And, whenever $\rightsquigarrow\!\!\!\!\rightarrow\!\!\!\!>$ is acyclic, $\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>$ is also acyclic.

Definition 138. *For (B,P,t) and (C,Q,u) in $CanCop_G$, we write $(B,P,t)\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>(C,Q,u)$ if and only if there exists a path of the following shape (with $v \neq v'$)*

$$((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(C)},Q),[!_v]@U.?_u) \rightsquigarrow^* ((\_,\_),[!_{v'}]@\_)$$

For $(B,P),(C,Q) \in Pot(B_G)$, we write $(B,P)\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>(C,Q)$ iff $\exists t,u \in Sig, (B,P,t)\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>(C,Q,u)$.

In the remaining of this chapter, we will work with $\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>$. The main reasons for the definition of $\rightsquigarrow\!\!\!\!\rightarrow\!\!\!\!>$ were pedagogical: $\rightsquigarrow\!\!\!\!\rightarrow\!\!\!\!>$ is simpler than $\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>$, it is closely related to the previously defined relation $\rightsquigarrow\!\!\!\!\rightarrow\!\!\!\!>$, and the proof of Lemma 122 is slightly more readable than if we proved it directly on $\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>$.

Definition 139. *For $n \in \mathbb{N}$, we set S_n as the subset of $CanCop_G$ defined by:*

$$\{(B,P,t) \in CanCop_G \mid s_{\rightsquigarrow}(B,P,t) \leq n\}$$

To simplify notations, in this section, we will write $((e,P),T)^{/n}$ for $((e,P),T)^{/\mapsto_{S_n}}$, $(e,Q)^n$ for $(e,Q)^{\mapsto_{S_n}}$, $((e,P),T)^n$ for $((e,P),T)^{\mapsto_{S_n}}$, $Cop_n(B,P)$ for $Cop_{\mapsto_{S_n}}(B,P)$ and $Can_n(B,P)$ for $Can_{\mapsto_{S_n}}(B,P)$.

In the proof-net of Figure 4.1, the only $\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>$ pairs are of the shape $(C,[],\mathtt{r}(\mathtt{l}(x)))\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>(B,[],\mathtt{r}(\mathtt{e}))$ or $(C,[],\mathtt{r}(\mathtt{r}(x)))\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>(B,[],\mathtt{r}(\mathtt{e}))$. The copies of $(B,[])$ are $\{\mathtt{e},\mathtt{l}(\mathtt{e}),\mathtt{r}(\mathtt{e})\}$ and the copies of $(C,[])$ are $\{\mathtt{e},\mathtt{l}(\mathtt{e}),\mathtt{r}(\mathtt{e}),\mathtt{r}(\mathtt{l}(\mathtt{e})),\mathtt{r}(\mathtt{r}(\mathtt{e}))\}$. So the only pairs of $\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>$ in $CanCop_G^2$ are $(C,[],\mathtt{r}(\mathtt{l}(\mathtt{e})))\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>(B,[],\mathtt{l}(\mathtt{e}))$ and $(C,[],\mathtt{r}(\mathtt{r}(\mathtt{e})))\rightsquigarrow\!\!\!\!\dot{\rightarrow}\!\!\!\!>(B,[],\mathtt{r}(\mathtt{e}))$.

So $S_0 = \emptyset$, $S_1 = \{(B,[],\mathtt{e}),(B,[],\mathtt{l}(\mathtt{e})),(B,[],\mathtt{r}(\mathtt{e})),(C,[],\mathtt{e}),(C,[],\mathtt{l}(\mathtt{e})),(C,[],\mathtt{r}(\mathtt{e}))\}$. For every $k \geq 2$, $S_k = \{(B,[],\mathtt{e}),(B,[],\mathtt{l}(\mathtt{e})),(B,[],\mathtt{r}(\mathtt{e})),(C,[],\mathtt{e}),(C,[],\mathtt{l}(\mathtt{e})),(C,[],\mathtt{r}(\mathtt{e})),(C,[],\mathtt{r}(\mathtt{l}(\mathtt{e}))),(C,[],\mathtt{r}(\mathtt{r}(\mathtt{e})))\}$.

So $Cop_0(B,[]) = \{\mathtt{e}\}$, $Cop_1(B,[]) = Cop_2(B,[]) = \{\mathtt{e},\mathtt{l}(\mathtt{e}),\mathtt{r}(\mathtt{e})\}$. For $(C,[])$, we can notice that we have $Cop_0(C,[]) = \{\mathtt{e}\}$, $Cop_1(C,[]) = \{\mathtt{e},\mathtt{l}(\mathtt{e}),\mathtt{r}(\mathtt{e})\}$ and $Cop_2(C,[]) = \{\mathtt{e},\mathtt{l}(\mathtt{e}),\mathtt{r}(\mathtt{e}),\mathtt{r}(\mathtt{l}(\mathtt{e})),\mathtt{r}(\mathtt{r}(\mathtt{e}))\}$.

Lemma 140. *Let us suppose that there exist paths of canonical contexts of the shape:*

$$\begin{aligned}
&((\sigma(A),P_0),[!_{t_0}]) \rightsquigarrow ((e_1,P_1),T_1) \rightsquigarrow ((e_2,P_2),T_2)\cdots((e_k,P_k),T_k) = ((\overline{\sigma(B)},P),U.?_t)\\
&((\sigma(A),P'_0),[!_{t'_0}]) \rightsquigarrow ((e_1,P'_1),T'_1) \rightsquigarrow ((e_2,P'_2),T'_2)\cdots((e_k,P'_k),T'_k) = ((\overline{\sigma(B)},P'),U'.?_{t'})
\end{aligned}$$

Then $s_{\rightsquigarrow}(B,P,t) = s_{\rightsquigarrow}(B,P',t')$

Proof. Because the paths play symmetric roles, it is enough to prove $s_{\rightsquigarrow}(B,P,t) \leq s_{\rightsquigarrow}(B,P,t)$. We prove the lemma by induction on $n = s_{\rightsquigarrow}(B,P,t)$. If $n = 1$ then, by definition of $s_{\_}(\_)$, we have $s_{\rightsquigarrow}(B,P',t') \geq 1$ (this is true for every element of $BoxSig_G$).

If $n > 1$ there exists $(C,Q,u) \in CanCop_G$ such that $(B,P,t)\rightsquigarrow\!\!\!\!\!\!\cdot\;>(C,Q,u)$ and $s_{\rightsquigarrow}(C,Q,u) = n-1$. By definition of $\rightsquigarrow\!\!\!\!\!\!\cdot\;>$, there exists a path of the following shape (we consider the first step after $(\overline{\sigma(C)},Q)$ where the signature of the leftmost trace element decreases)

$$((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(C)},Q),[!_v]@V.?_u) \rightsquigarrow^* ((e,R),[!_v]) \rightsquigarrow ((f,R),[!_w]@W)$$

Because $((\overline{\sigma(B)},P),U.?_t)$ is quasi-standard, t is a standard signature. Thus

$$((\sigma(B),P),U.!_t) \rightsquigarrow^* ((\overline{\sigma(C)},Q),U.!_v@V.?_u) \rightsquigarrow^* ((e,R),U.!_v) \rightsquigarrow ((f,R),U.!_w@W)$$

By definition of the $\rightsquigarrow$ relation, we have the dual path:

$$((\overline{f},R),U^\perp.?_w@W^\perp) \rightsquigarrow ((\overline{e},R),U^\perp.?_v) \rightsquigarrow^* ((\sigma(C),Q),U^\perp.?_v@V^\perp.!_u) \rightsquigarrow^* ((\overline{\sigma(B)},P),U^\perp.?_t)$$

Because $\rightsquigarrow$ is injective and $((\sigma(A),P_0),[!_{t_0}])$ has no antecedent for $\rightsquigarrow$, This path is a suffix of the path $((e_0,P_0),T_0) \rightsquigarrow^k ((e_k,P_k),T_k)$. Thus, the k last steps of the path $((e'_0,P'_0),T'_0) \rightsquigarrow^k ((e_k,P_k),T_k)$ are of the shape

$$((\overline{f},R'),U'^\perp.?_{w'}@W'^\perp) \rightsquigarrow ((\overline{e},R'),U'^\perp.?_{v'}) \rightsquigarrow^* ((\sigma(C),Q'),U'^\perp.?_{v'}@V'^\perp.!_{u'}) \rightsquigarrow^* ((\overline{\sigma(B)},P'),U'^\perp.?_{t'})$$

with $w' \neq v'$ and U' is a strict prefix of every trace. Thus, we can remove U' on every trace and reverse the path. We obtain the following path:

$$((\sigma(B),P'),[!_{t'}]) \rightsquigarrow^* ((\overline{\sigma(C)},Q'),[!_{v'}]@V'.?_{u'}) \rightsquigarrow^* ((e,R'),[!_{v'}]) \rightsquigarrow ((f,R'),[!_{w'}]@W')$$

By definition of $\rightsquigarrow\!\!\!\!\!\!\cdot\;>$, we have $(B,P',t')\rightsquigarrow\!\!\!\!\!\!\cdot\;>(C,Q',v')$. Moreover, we know by induction hypothesis that $s_{\rightsquigarrow}(C,Q',v') = s_{\rightsquigarrow}(C,Q,v) = n-1$ so $s_{\rightsquigarrow}(B,P',t') \geq n$. □

Lemma 141. *For every $(B,P) \in Pot(B_G)$, $s_{\rightsquigarrow}(B,P,\mathsf{e}) = 1$.*

Proof. We can prove the lemma by contradiction. If $s_{\rightsquigarrow}(B,P,\mathsf{e}) > 1$ then there exists (C,Q,u) such that $(B,P,\mathsf{e})\rightsquigarrow\!\!\!\!\!\!\cdot\;>(C,Q,u)$. It is to say, there is a path of the shape:

$$((\sigma(B),P),[!_{\mathsf{e}}]) \rightsquigarrow^* ((\overline{\sigma(C)},Q),[!_v]) \rightsquigarrow^* ((\_,\_),[!_w])$$

with $v \neq w$. However, by definition of $\mapsto$, we have $v = w = \mathsf{e}$. □

Lemma 142. *For every $n \geq 1$, S_n is mixable.*

Proof. The second property of the definition of S_n is proved in Lemma 141. Here, we will prove the first property by induction on n. We have to prove that for every (B,P_1,t_1) and (B,P_2,t_2) in $CanCop_G$, if $(B,P_1,t_1) \in S_n$ and $(B,P_2,t_2) \in S_n$ then $(B,\mathrm{mix}(P_1,P_2),\mathrm{mix}(t_1,t_2))$ is also in S_n.

For $n \geq 1$, we will prove the property by contraposition. Let us consider (B,P_1,t_1) and (B,P_2,t_2) in $CanCop_G$. Let us suppose that $(B,P,t) = (B,\mathrm{mix}(P_1,P_2),\mathrm{mix}(t_1,t_2))$ is not in S_n. We will prove that either $(B,P_1,t_1) \notin S_n$ or $(B,P_2,t_2) \notin S_n$. By definition of this set, $s_{\rightsquigarrow}(B,P,t) > n$. So there exists $(C,Q,u) \in CanCop_G$ such that $(B,P,t)\rightsquigarrow\!\!\!\!\!\!\cdot\;>(C,Q,u)$ and $s_{\rightsquigarrow}(C,Q,u) \geq n$. By definition of $\rightsquigarrow\!\!\!\!\!\!\cdot\;>$, there exists a path of the following shape (with $v \neq w$):

$$((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(C)},Q),[!_v]@V.?_u) \rightsquigarrow^* ((e,R),[!_v]) \rightsquigarrow ((f,R),[!_w]@W)$$

Because $v \neq w$, we can deduce that $v \neq \mathsf{e}$. By Lemma 132, either there exists a path of the following shape (with $v_1 \neq \mathsf{e}$):

$$((\sigma(B), P_1), [!_{t_1}]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q_1), [!_{v_1}]@V_1.?_{u_1}) \rightsquigarrow^* ((e, R_1), [!_{v_1}]) \rightsquigarrow ((f, R_1), [!_{w_1}]@W_1)$$

or there exists a similar path beginning by $((\sigma(B), P_2), [!_{t_2}])$. Because (B, P_1, t_1) and (B, P_2, t_2) play symmetric roles, we suppose without loss of generality that we are in the first case. By definition of $\rightsquigarrow\!\!\!\!\rightarrow$, we have $(B, P_1, t_1) \rightsquigarrow\!\!\!\!\rightarrow (C, Q_1, u_1)$. By Lemma 140, $s_{\rightsquigarrow\!\!\!\!\rightarrow}(C, Q_1, u_1) = s_{\rightsquigarrow\!\!\!\!\rightarrow}(C, Q, u) = n$ so $s_{\rightsquigarrow\!\!\!\!\rightarrow}(B, P_1, t_1) > n$ and $(B, P_1, t_1) \notin S_n$. □

Lemma 143. *Let us consider $(B, P, t) \in S_n$ such that $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\sigma(C), Q), [!_{t_0}]@U.!_u) \rightsquigarrow^* ((\_, \_), [!_{t_1}]@\_)$ with $t_0 \neq t_1$. Then $(C, Q, u) \in S_n$.*

Proof. We prove it by contraposition. Let us suppose that $(C, Q, u) \notin S_n$ then by definition of S_n, we have $s_{\rightsquigarrow\!\!\!\!\rightarrow}(C, Q, u) > n$. So there exists a path of the shape $((\sigma(C), Q), [!_u]) \rightsquigarrow^* ((\overline{\sigma(D)}, R), [!_{u_1}]V.?_v) \rightsquigarrow^* ((\_, \_), [!_{u_2}])$ with $u_1 \neq u_2$ and $s_{\rightsquigarrow\!\!\!\!\rightarrow}(D, R, v) \geq n$.

By Lemma 9, we can deduce the following path:

$$\begin{aligned}((\sigma(B), P), [!_t]) &\rightsquigarrow^* ((\sigma(C), Q), [!_{t_0}]@U.!_u) \\ &\rightsquigarrow^* ((\overline{\sigma(D)}, R), [!_{t_0}]@U@[!_{u_1}]@V.?_v) \rightsquigarrow^* ((\_, \_), [!_{t_0}]@U@[!_{u_2}]) \\ &\rightsquigarrow^* ((\_, \_), [!_{t_1}])\end{aligned}$$

So, by definition of $\rightsquigarrow\!\!\!\!\rightarrow$, we have $(B, P, t) \rightsquigarrow\!\!\!\!\rightarrow (D, R, v)$ and $s_{\rightsquigarrow\!\!\!\!\rightarrow}(B, P, t) > s_{\rightsquigarrow\!\!\!\!\rightarrow}(D, R, v) \geq n$. By definition of S_n, $(B, P, t) \notin S_n$. □

Lemma 144. *Let us consider $n \in \mathbb{N}$, (B, P, t) and (B, P', t') in $CanCop_G$ such that $(B, P, t) \notin S_n$ and $((\sigma(B), P), [!_t])^n = ((\sigma(B), P'), [!_{t'}])^n$ then $(B, P', t') \notin S_n$.*

Proof. We prove the lemma by induction on $(s, s \nrightarrow (B, P, t))$. By definition of S_n, $s_{\rightsquigarrow\!\!\!\!\rightarrow}(B, P, t) > n$. So there exists $(C, Q, u) \in CanCop_G$ such that $s_{\rightsquigarrow\!\!\!\!\rightarrow}(C, Q, u) \geq n$ and there exists a path of the shape:

$$((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q), [!_{t_1}]@U.?_u) \rightsquigarrow^* ((e, P_e), [!_{t_e}])$$

with $t_1 \neq t_e$. Let us consider the lowest i such that $((\sigma(B), P), [!_t]) \mapsto^i ((\sigma(D), R), [!_{t_d}]@V.!_v) \mapsto^* ((e, P_e), [!_{t_e}])$ with $t_d \neq t_e$ and $(D, R, v) \notin S_n$.

- If such a step does not exist, by Lemmas 142 and 137, there exists a path of the shape

  $$((\sigma(B), P'), [!_{t'}]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q'), [!_{t'_1}]@U'.?_{u'}) \mapsto^* ((e, P'_e), [!_{t'_e}])$$

  with $((\overline{\sigma(C)}, Q), [!_{t_1}]@U.?_u)^n = ((\overline{\sigma(C)}, Q'), [!_{t'_1}]@U'.?_{u'})^n$ and $((e, P_e), [!_{t_e}])^n = ((e, P'_e), [!_{t'_e}])^n$. Let us notice that $(B, P', t') \rightsquigarrow\!\!\!\!\rightarrow (C, Q', u')$. By induction hypothesis (we could also use Lemma 140) $s_{\rightsquigarrow\!\!\!\!\rightarrow}(C, Q', u') \geq n$ so $s_{\rightsquigarrow\!\!\!\!\rightarrow}(B, P', t') > n$.

- If such a step does exist, by Lemmas 142 and 137, there exists a path of the shape

  $$((\sigma(B), P'), [!_{t'}]) \rightsquigarrow^i ((\sigma(D), R'), [!_{t'_1}]@V'.!_{v'})$$

  with $((\sigma(D), R), [!_v])^n = ((\sigma(D), R'), [!_{v'}])^n$. We know that $(D, R, v) \notin S_n$ and $(B, P, t) \nrightarrow^+ (D, R, v)$. Thus, by induction hypothesis, we have $(D, R', v') \notin S_n$. So, by Lemma 143, $(B, P', t') \notin S_{n-1}$.

□

Theorem 145. *Let C, D, D' be canonical contexts. If $C \rightsquigarrow_{S_n} D$ and $D^n = D'^n$ then there exists a canonical context C' such that $C' \rightsquigarrow_{S_n} D'$ and $C^n = C'^n$.*

Proof. Immediate from Lemmas 136 and 144. □

Lemma 146. *Let $n \in \mathbb{N}$. If $((\sigma(B), P), [!_t]) \mapsto^*_{S_n} C \mapsto^*_{S_n} D$ and $D^n = D'^n$ then there exists a context C' such that $C' \mapsto^*_{S_n} D'$ and $C^n = C'^n$.*

Proof. This Lemma is deduced from Theorem 145 exactly as Lemma 74 is deduced from Theorem 72. □

Lemma 147. *For $n \in \mathbb{N}$, if $((\sigma(B), P), [!_t]) \mapsto^*_{S_n} ((e, Q), [!_u]) \mapsto^+_{S_n} ((e, Q'), [!_v])$ then $e, Q^{n-1} \neq e, Q'^{n-1}$.*

Proof. This Lemma is deduced from Lemma 146 exactly as Lemma 75 is deduced from Lemma 74. □

Theorem 148. *For any proof-net G, the length of its longest path of reduction is bounded by $2^{2^{|\vec{E}_G|}}_{3|\vec{\rightsquigarrow}|}$.*

Proof. This Lemma is deduced from Lemma 147 exactly as Lemma 78 is deduced from Lemma 75. □

Thus, to have an elementary bound on a proof-net G, it is enough to suppose that, for every proof-net H, the depth of the relation $|\vec{\rightsquigarrow}|$ on $(G)H$ is bounded by M_G which does not depend on H.

4.4 Primitive recursive bound

In this section, we will prove that the acyclicity of $\rightsquigarrow\!\!\!\!\!\!\cdot\!\!\!\!\!\!\rightarrow$ entails a primitive recursive bound on the length of *cut*-elimination. To guide intuition, we will sketch the proof that the acyclicity of $\dashrightarrow$ entails a primitive recursive bound. The idea is to view canonical boxes as a forest whose roots are the canonical boxes $(B, [])$ (with $\partial(B) = 0$), there is an arrow from (B, P) to $(C, P.t)$ iff B is the deepest box containing C. The idea is to progressively unveil the forest, starting by its roots.

Let us consider a finite set S of canonical boxes. Because $\dashrightarrow$ is supposed acyclic, S admits a minimal element (B_0, P_0) for $\dashleftarrow$ (it is to say, $(B_0, P_0)\dashrightarrow(B_1, P_1)$ implies that (B_1, P_1) is not in S). Let us notice that, if there exists an arrow from (B, P) to $(C, P.t)$, then $(C, P.t)\dashrightarrow(B, P)$. Thus (B_0, P_0) must be minimal with respect to the forest: if there is an arrow from (B_1, P_1) to (B_0, P_0) then $(B_1, P_1) \notin S$.

Because of this property, we know that one of the $(B_1, [])$ potential boxes at depth 0 is minimal for $\dashleftarrow$. Then, the elementary stratification property gives us a bound on $|Cop(B_1, [])|$. Thus, for every box C immediately included in B_1, we have a bound on $|Can(C)|$. Then, there is a potential box (B_2, P_2) among $\{(B, []) \in Can(B_G) \mid B \neq B_1\} \cup \{(B, [t]) \in Can(B_G) \mid B \subset B_1\}$ such that $(B_2, P_2)\dashrightarrow(C, Q) \Rightarrow (C, Q) = (B_1, [])$. Thus, by the elementary stratification property, one can prove a bound on $|Cop(B_2, P_2)|$. Then, for every box C immediately included in B_2, we have a bound on $|Can(C)|$...

Let us consider, at each step, the *leaves* of the subforest: the potential boxes $(B, [p_1; \cdots ; p_{\partial(B)}])$ with $B \subset B_{\partial(B)} \subset \cdots \subset B_1$ such that

- For every $1 \leq i < \partial(B)$, we have proved a bound on $Cop(B, [p_1; \cdots ; p_i])$.
- We have not yet proved a bound on $Cop(B, [p_1; \cdots ; p_{\partial(B)}])$.

At each step we delete a leaf (B, P) at a certain depth $\partial(B)$, and the leaves we create (the elements of $\{(C, P.t) \in Pot(B_G) \mid C \subset B \wedge t \in Cop(B, P)\}$) all are at depth $\partial(B) + 1$. Thus, the sets of weight decreases along the multiset order. Such order allows us to deduce a primitive recursive bound on reduction [42].

However, as written before, our goal is not to prove that the acyclicity of $\dashrightarrow$ entails a primitive recursive bound, but that the acyclicity of $\rightsquigarrow\!\!\!\!\!\!\cdot\!\!\!\!\!\!\rightarrow$ entails a primitive recursive bound. In this case, it is possible that no maximal element of S for $\rightsquigarrow\!\!\!\!\!\!\cdot\!\!\!\!\!\!\rightarrow$ is minimal for the forest. Thus, the order we use is a bit more complex. It relies on the acyclicity of $\mapsto$ on contexts (Lemma 100, Section 3.4). As a parallel, one might compare the proof that the acyclicity of $\twoheadrightarrow$ entails an elementary bound (Theorem 32, Section 3.1.3) and the proof that the acyclicity of $\rightsquigarrow\!\!\!\!\!\!\rightarrow$ entails an elementary bound (Theorem 78, Section 3.2.3). For Theorem 32, we bound $\max_{(B,P)\in Pot(B)} |Cop(B, P)|$ one box B at a time. For Theorem 78, we bound the number of restricted copies (which, at the end of the proof, correspond to copies because we consider $n = |\ \rightsquigarrow\!\!\!\!\!\!\rightarrow\ |$). This is the approach we will use. We build, step by step, a set S of canonical boxes and bound the size of $Can_{\mapsto_S}(\mathrm{E}_G)$. Here, the set $Can_{\mapsto_S}(B_G)$ plays the same role as the sub-forest progressively unveiled in the proof sketch for $\dashrightarrow$-stratified proof-nets. The leaves correspond to the set $Can_{\mapsto_S}(B_G) - S$.

In order to prove the bounds by induction, the set S will be required to verify certain constraints. Indeed, S is supposed to be a set of canonical boxes (B, P) for which we have a bound on $Cop_{\mapsto_S}(B, P)$. Let us suppose that B_0 is a box included in B, $(B, P) \in S$, $(B_0, P.\mathtt{l}(\mathtt{e})) \in S$, $(C, Q) \notin S$, $t' \neq \mathtt{e} \in Sig$ and we have paths of the shape:

$$\begin{array}{rl}
 & ((\sigma(B), P), [!_{\mathtt{l}(\mathtt{r}(\mathtt{e}))}]) \quad \rightsquigarrow^* \hookrightarrow ((\sigma(C), Q), [!_{\mathtt{r}(\mathtt{e})}]) \\
 & ((\sigma(B_0), P.\mathtt{l}(\mathtt{r}(\mathtt{e})), [!_t]) \rightsquigarrow^* ((\sigma(C), Q), [!_{t'}]@U@[!_u]) \rightsquigarrow^* ((_, _), [!_{\mathtt{e}}]) \\
((\sigma(B_0), P.\mathtt{l}(\mathtt{r}(\mathtt{e})))^{\mapsto_S}, [!_t]) = & ((\sigma(B_0), P.\mathtt{r}(\mathtt{e})), [!_t]) \quad \rightsquigarrow^* ((\sigma(C), Q), [!_{t'}]@U@[!_{\mathtt{e}}]) \rightsquigarrow^* ((_, _), [!_{t'}]@V) \not\rightsquigarrow
\end{array}$$

Then, if we add (C, Q) in S (i.e. when we define $S' = S \cup \{(C, Q)\}$), the signature t is a $\mapsto_{S'}$-copy of $(B, P.\,\mathtt{l}(\mathtt{r}(\mathtt{e})))^{\mapsto_{S'}}$ even if it is not a copy of $(B, P.\,\mathtt{l}(\mathtt{r}(\mathtt{e})))^{\mapsto_S}$. Such behaviour would make the proof of bounds very hard. This is why, we forbid such behaviour in the following definition of n-coherent sets.

Definition 149. *Let $n \in \mathbb{N}$. A n-coherent set is a subset S of $Can_n(B_G)$ such that:*

- *For every $(B, P) \in S$ and $t \in Sig$, $(B, P, t) \in S_n$.*
- *If $(B, P) \in Can_n(B_G) - S$ then there exists $t \in Sig$ such that $(B, P, t) \notin S_{n-1}$.*
- *If $(B, P) \in S$ and $(B, P)^n = (B, Q)^n$ then $(B, Q) \in S$.*
- *If $(B, P) \in S$ and $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\sigma(C), Q), U@[!_u])$ then $(C, Q) \in S$*

To understand why the proof of this section is delicate, let us consider the following case: $(B, P) \in S$, $(C, Q) \in Can_{\mapsto_S}(B_G) - S$, $((\sigma(B), P), [!_{\mathtt{l}(\mathtt{r}(\mathtt{e}))}]) \rightsquigarrow^* \hookrightarrow ((\sigma(C), Q), [!_{\mathtt{r}(\mathtt{e})}])$ and B_0 (resp. C_0) is a box included in B (resp. C). In this case $(B_0, P.\,\mathtt{l}(\mathtt{e}))$ is in $Can_{\mapsto_S}(B_G)$ and $(B_0, P.\,\mathtt{l}(\mathtt{r}(\mathtt{e})))$ is not in $Can_{\mapsto_S}(B_G)$. Let us suppose that we add (C, Q) to S (i.e. $S' = S \cup \{(C, Q)\}$). Let us observe the effect this step has on the leaves:

- (C, Q) goes from $Can_{\mapsto_S}(B_G) - S$ to S', so there is one less leaf at depth $\partial(C)$.
- $(C_0, Q.\,\mathtt{r}(\mathtt{e}))$ is not in $Can_{\mapsto_S}(B_G)$ but is in $Can_{\mapsto_{S'}}(B_G) - S'$ so we have new leaves at depth $\partial(C) + 1$. Because it is at a higher depth, it is not a problem with respect to the multiset ordering.
- $(B_0, P.\,\mathtt{l}(\mathtt{r}(\mathtt{e}))$ is not in $Can_{\mapsto_S}(B_G)$ but is in $Can_{\mapsto_{S'}}(B_G) - S'$ so we have new leaves at depth $\partial(B) + 1$. However, we may have $\partial(B) < \partial(C)$.

So, the multiset previously defined is not precise enough. We have to define a new ordering. This is the goal of the Definition 150.

Definition 150. *For any coherent set S, the weight of S is the tuple $W^S = \left(|Can_{\mapsto_S}(\mathrm{E}_G)|, (w^S_{i,j})_{0 \le i,j < \partial_G}\right)$ with $w^S_{i,j}$ defined as $\left|e^S_{i,j}\right|$ with $e^S_{i,j}$ a set of pairs of potential boxes defined by:*

$$e^S_{i,j} = \left\{ ((B, P), (C, Q)) \in Can_{\mapsto_S}(B_G)^2 \;\middle|\; \begin{array}{c} \partial(B) = i, \partial(C) = j \\ (C, Q) \notin S, \text{ and } \exists P' \blacktriangleright P, Q' \blacktriangleright Q, \exists t \in Sig \text{ such that} \\ ((\sigma(B), P'), [!_t]) \mapsto^* ((\sigma(C), Q'), [!_{\mathtt{e}}]) \end{array} \right\}$$

We order the weights by the lexicographic order. We write $W^S < W^T$ if and only if $|Can_{\mapsto_S}(\mathrm{E}_G)| \le |Can_{\mapsto_T}(\mathrm{E}_G)|$ and there exists $1 \le a, b \le \partial(G)$ such that:

- $w^S_{a,b} < w^T_{a,b}$
- *For $i < a$ and $0 \le j < \partial_G$, $w^S_{i,j} \le w^T_{i,j}$*
- *For $j < b$, $w^S_{a,j} \le w^T_{a,j}$*

Lemma 151. *There exist primitive recursive functions p, p' satisfying the following condition. For every coherent set S such that $Can_{\mapsto_S}(B_G)$ is finite and $Can(B_G) \not\subseteq S$, there exists a coherent set T such that $W^T < W^S$, $|Can_{\mapsto_T}(\mathrm{E}_G)| \le p\left(|Can_{\mapsto_S}(\mathrm{E}_G)|\right)$ and for every $0 \le a, b < \partial_G$, $w^T_{a,b} \le p'\left(\max_{0 \le i,j < \partial_G} w^S_{i,j}\right)$.*

Proof. Let us consider the set $\overline{S} = Can(B_G) - S$. By assumption the set is not empty. For $(B,P) \in Can(B_G) - S$, we can notice that $(B,P)^{\mapsto_S} \in Can_{\mapsto_S}(B_G)$ and $((B,P)^{\mapsto_S})^{\mapsto_S} = (B,P)^{\mapsto_S}$ so (by definition of coherent sets) $(B,P)^{\mapsto_S} \notin S$. So the set $Can_{\mapsto_S}(B_G) - S$ is not empty. We write n for the natural number such that S is n-coherent

- If there exists $(B,P) \in Can_{\mapsto_S}(B_G) - S$ such that, for every $t \in Sig$, $(B,P,t) \in S_n$ then we set

  $$E = \{(B,P) \in Can_{\mapsto_S}(B_G) - S \mid \forall t \in Sig, (B,P,t) \in S_n\}$$

  We can observe that E is finite and non-empty. Moreover, by Lemma 101, we know that the projection of $\mapsto$ on this set is acyclic. So there exists $(B,P) \in E$ which is maximal in E for $\mapsto$ (i.e. $(B,P,t) \mapsto^* (C,Q,u)$ implies that (C,Q) is not in E). Then we define T as the set $T = S \cup \{(B,P') \in Can_{\mapsto_S}(B) \mid (B,P)^n = (B,P')^n\}$ and we will show that T is n-coherent.

  - Let us consider $(C,Q) \in T$ and $u \in Sig$. If $(C,Q) \in S$ then, because S is a n-coherent set, $(C,Q,u) \leq n$. Else $(B,P)^n = (C,Q)^n$. Because $(B,P) \in E$, $(B,P,u) \in S_n$.
  - Let us consider $(C,Q) \in Can_n(B_G) - T$ then $(C,Q) \in Can_n(B_G) - S$. So, because S is a n-coherent set, there exists $u \in Sig$ such that $(C,Q,u) \notin S_{n-1}$.
  - If $(C,Q) \in T$ and $(C,Q)^n = (C,R)^n$ then, either $(C,Q) \in S$ so (because S is a n-coherent set) $(C,R) \in S \subseteq T$. Or $(B,P)^n = (C,Q)^n = (C,R)^n$ so $(C,R) \in T$.
  - If $(C,Q) \in T$ and there exist a path of the shape $((\sigma(C),Q),[!_t]) \rightsquigarrow^* ((\sigma(D),R),U@[!_u])$. Then either $(C,Q) \in S$ and in this case, because S is a n-coherent set, $(D,R) \in S \subseteq T$. Or $(B,P)^n = (C,Q)^n$. In this case there exists $R' \in Pot$ and $u' \in Sig$ such that $(D,R')^n = (D,R)^n$ and $((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\sigma(D),R'),U'@[!_{u'}])$. Let us notice that $(B,P,t) \mapsto (D,R',u')$. So, $(D,R') \notin E$. It is to say, either $(D,R') \in S$ or there exists $v \in Sig$ such that $(D,R',v) \notin S_n$. In the second case we would have $(C,Q,t) \notin S_n$ (Lemma 143) so $s_{\rightsquigarrow}(B,P,t) > n$ which is a contradiction. So $(D,R') \in S$. And, because S is n-coherent, we have $(D,R) \in S$.

- Else, we can observe that $Can_{\mapsto_S}(B_G) - S$ is finite and non-empty. So, by Lemma 101, there exists $(B,P) \in Can_{\mapsto_S}(B_G) - S$ so that whenever $(B,P,t) \mapsto^+ (C,Q,u)$, $(C,Q) \notin Can_{\mapsto_S}(B_G) - S$. We will show that $T = S \cup \left\{(B,P') \in Can_{\mapsto_S}(B) \mid (B,P)^{n+1} = (B,P')^{n+1}\right\}$ is a $n+1$-coherent set.

  - Let us consider $(C,Q) \in T$ and $u \in Sig$. If $(C,Q) \in S$ then, because S is a n-coherent set, $(C,Q,u) \in S_n \subset S_{n+1}$.
    Else $(B,P)^{n+1} = (C,Q)^{n+1}$. Let us suppose that $(C,Q,u) \notin S_{n+1}$, then we would have $(B,P,u) \notin S_{n+1}$ (Lemma 144). So $s_{\rightsquigarrow\!\!\!\!\rightarrow}(B,P,u) > n+1$. By definition of $s\_(\_)$, there exists (D,R,w) such that $(B,P,u) \rightsquigarrow\!\!\!\!\rightarrow (D,R,w)$ and $s_{\rightsquigarrow\!\!\!\!\rightarrow}(D,R,w) > n$. Because S is n-coherent, $(D,R) \notin S$. So $(D,R) \notin S$ and $(B,P,u) \mapsto (D,R,w)$. This contradicts the minimality of (B,P). So our hypothesis was wrong, $s_{\rightsquigarrow\!\!\!\!\rightarrow}(C,Q) \leq n+1$.
  - Let us consider $(C,Q) \in Can_n(B_G) - T$ then $(C,Q) \in Can_n(B_G) - S$. By supposition, there exists $u \in Sig$ such that $(C,Q,u) \notin S_n$.
  - If $(C,Q) \in T$ and $(C,Q)^{n+1} = (C,R)^{n+1}$ then, either $(C,Q) \in S$. In this case, let us notice that we also have $(C,Q)^n = (C,R)^n$. So, because S is a n-coherent set, $(C,R) \in S \subseteq T$. Or $(B,P)^{n+1} = (C,Q)^{n+1} = (C,R)^{n+1}$ so $(C,R) \in T$.

– If $(C, Q) \in T$ and there exists a path of the shape $((\sigma(C), Q), [!_t]) \rightsquigarrow^* ((\sigma(D), R), U@[!_u])$. Then either $(C, Q) \in S$ and in this case, because S is a n-coherent set, $(D, R) \in S \subseteq T$. Or $(B, P)^{n+1} = (C, Q)^{n+1}$. In this case there exists $R' \in Pot$ and $u' \in Sig$ such that $(D, R')^{n+1} = (D, R)^{n+1}$ and $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\sigma(D), R'), U'@[!_{u'}])$. Let us notice that $(B, P, t) \nmapsto (D, R', u')$. So, $(D, R') \notin Can_{\mapsto_S}(B_G) - S$. So $(D, R') \in S$. So, because $(D, R')^{n+1} = (D, R)^{n+1}$, $(D, R')^n = (D, R)^n$. Because S is n-coherent, we have $(D, R) \in S \subseteq T$.

As we proved above, for every $t \in Sig$, if $(B, P, t) \subseteq (C, Q, u)$ or $(B, P, t) \dddot{\rightsquigarrow} (C, Q, u)$ then $(C, Q) \in S$. We set $m = n$ if we are in the first case, $m = n + 1$ otherwise. Let us consider $(B, P') \in Can_{\mapsto_S}(B_G)$ such that $(B, P)^m = (B, P')^m$. Then, for every $t \in Sig$, if $(B, P', t) \subseteq (C, Q', u)$ or $(B, P', t) \dddot{\rightsquigarrow} (C, Q', u)$ then there exists $(C, Q)^m = (C, Q)^m$ such that $(B, P, t) \subseteq (C, Q, u)$ or $(B, P, t) \dddot{\rightsquigarrow} (C, Q, u)$. So $(C, Q) \in S$ and $(C, Q') \in S$.

Let us give a bound on $|Cop_{\mapsto_T}(B, P')|$. Let us consider $t \in Cop_{\mapsto_T}(B, P')$ and $u \sqsupseteq t$. By definition there exists a path of the shape $((\sigma(B), P'), [!_u]) \mapsto_T^* ((\_, \_), [!_{\mathtt{e}}])$. Thus, $((\sigma(B), P'), [!_u]) \mapsto_T \mapsto_S^* ((\_, \_), [!_{\mathtt{e}}])$. Let us suppose that there exists two contexts $((e, Q), [!_v])$ and $((e, Q'), [!_{v'}])$ in the paths such that $(e, Q)^{\mapsto_S} = (e, Q')^{\mapsto_S}$. Then, as in the proof of Lemma 75, we can prove that it entails a contradiction. So the number of contexts of the shape $((\_, \_), [!\_])$ in the path is bounded by $\left|Can_{\mapsto_S}(B_G)\right|$. Thus, as in the proof of Theorem 78, we can deduce that:

$$\left|Cop_{\mapsto_T}(B, P')\right| \leq 2^{2^{2 \cdot \left|Can_{\mapsto_S}(\mathrm{E}_G)\right|}}$$

We can notice that $\left|Can_{\mapsto_S}(B_G)\right| \leq \sum_{0 \leq i,j < \partial_G} w^S_{i,j}$. Thus, there exists a primitive recursive function q such that $\left|Cop_{\mapsto_T}(B, P')\right| \leq q\left(|\mathrm{E}_G|, \max_{0 \leq i,j < \partial_G} w^S_{i,j}\right)$. So there exists a primitive recursive function p such that $|Can_{\mapsto_T}(\mathrm{E}_G)| \leq p\left(|Can_{\mapsto_S}(\mathrm{E}_G)|\right)$.

Let b be the depth of B. And let a the minimal index such that there exists a box C at depth a, $(C, Q) \in Can_{\mapsto_S}(B_G)$, $Q' \blacktriangleright Q$ and $P' \blacktriangleright P$ such that $(C, Q') \between (B, P')$. Let us notice that $(B, P) \between (B, P)$ so a is well-defined and $a \leq b$.

- Let us consider $i \leq a$, $j \leq b$ and $((C, Q), (D, R)) \in e^T_{a,b}$. Then (C, Q) and (D, R) are in $Can_{\mapsto_T}(B_G)$, $(D, R) \notin T$, $|Q| = a$, $|R| = b$ and there exist $Q' \blacktriangleright Q$, $R' \blacktriangleright R$ and $v \in Sig$ such that $((\sigma(C), Q'), [!_v]) \mapsto^* ((\sigma(D), R'), [!_{\mathtt{e}}])$.

  If (C, Q) is not in $Can_{\mapsto_S}(B_G)$, then it would mean that there exists $(C_0, Q'_0) \supset (C, Q')$ such that $(C_0, Q'_0) \between (B, P')$ which contradicts the definition of a. So $(C, Q) \in Can_{\mapsto_S}(B_G)$.

  If (D, R) is not in $Can_{\mapsto_S}(B_G)$, then it would mean that there exists $(C_0, Q'_0) \supset (C, Q')$ such that $(C_0, Q'_0) \between (B, P')$. So there exists $(D_0, R'_0) \supset (D, R')$ such that: either $(D_0, R'_0) \dot{\between} (B, P')$ or there exists $w \in Sig$ such that $(D, R', w) \dddot{\rightsquigarrow} (B, P, t)$ with t the signature such that $s_{\dddot{\rightsquigarrow}}(B, P, t) = m$. Then, we would have $s_{\dddot{\rightsquigarrow}}(D, R', w) > m$ so $(D, R) \notin S$. In both case, we have a contradiction so $(D, R) \in Can_{\mapsto_S}(B_G)$.

  Then, (C, Q) and (D, R) are in $Can_{\mapsto_S}()$. Thus, $((C, Q), (D, R)) \in e^S_{a,b}$. So $e^T_{i,j} \subseteq e^S_{i,j}$ and $w^T_{i,j} \leq w^S_{i,j}$.

  Moreover, we can notice that, there exists $((C, Q), (B, P)) \in e^S_{a,b}$ but $((C, Q), (B, P)) \notin e^T_{a,b}$ because $(B, P) \in T$. So $e^T_{a,b} \subset e^S_{a,b}$ and $w^T_{a,b} < w^S_{a,b}$.

- Otherwise, because $w^T_{i,j} \subseteq Can_{\mapsto_T}(B_G)$, so:

$$w^T_{i,j} \leq \left(\max_{(B,P)^m = (B,P')^m} \|Cop_{\mapsto_T}(B, P)\| \cdot \left|Can_{\mapsto_S}(B_G)\right|\right)^2$$

So there exists a primitive recursive function p' such that $w^T_{i,j} \leq p'\left(|\mathrm{E}_G|, \max_{0\leq i,j<\partial_G} w^S_{i,j}\right)$.

□

Theorem 152. *For every $k \in \mathbb{N}$, there exists a primitive recursive function p_k such that, if G is a $\rightsquigarrow\!\!\!\!\!\!\rightarrow$-stratified proof-net and $\partial(G) \leq k$ then $W_G \leq p_k(|\mathrm{E}_G|)$.*

Proof. We define a symbol ϕ of arity k^2+1 and a term-rewriting system defined by the transition:

$$\phi(W^S) \to \phi(W^T) \text{ if } T \text{ is the coherent set obtained from } S \text{ in the proof of Lemma 151}$$

Then, with the notations of [42], for every $l \to r$, we have $l >_{MPO} r$ (with the precedence relation $>= \varnothing$). Then there exists a primitive recursive function q_k such that the length of any $\to$ sequence starting from $\phi(n_1, \cdots, n_{k^2+1})$ is bounded by $q_k(size(\phi(n_1, \cdots, n_{k^2+1})))$.

In particular, let us notice that $\varnothing$ is a 0-coherent set and $W^\varnothing = (n_1, \cdots, n_{k^2+1})$ with $n_i \leq |\mathrm{E}_G|$ for $1 \leq i \leq k^2+1$. Thus, there exists a primitive recursive function p_k such that there exists a coherent set S with $\left|Can_{\mapsto_S}(\mathrm{E}_G)\right| \leq p_k(|\mathrm{E}_G|)$ and $\phi(W^S) \twoheadrightarrow$. So, by definition of $\to$, $S = Can(B_G)$. Thus:

$$Can_{\mapsto_S}(\mathrm{E}_G) = Can(\mathrm{E}_G)$$
$$Can_{\mapsto_S}(\mathrm{E}_G) \leq p_k(|\mathrm{E}_G|)$$

□

Chapter 5

Linear logic subsystems and λ-calculus type-systems

In Chapters 3 and 4 we defined semantic criteria on proof-nets implying bounds on the length of *cut* elimination. Those criteria are all decidable in the following meaning: given a strongly normalizing proof-net G (let us recall that every LL proof-net is strongly normalizing), one can determine whether the relations $\twoheadrightarrow$, $\rightsquigarrow$, $\succ\!\!\!\rightarrow$, $\prec\!\!\!\blacktriangleleft$, $\prec\!\!\!\triangleleft$ and $\dot{\rightsquigarrow}$ are acyclic. Indeed, one can compute every path of context semantics by reducing the proof-net. Then, one can check each of those paths to compute the above relations. Similarly one can compute, for every possible assignments $\mathfrak{s}(\_)$ and $\mathfrak{d}(\_)$ and every subset S of B_G, the minimum relation $\boxdot\!\!\rightarrow$ satisfying the definition of weak stratification. Finally, one can verify whether this relation is acyclic.

However, computing the complexity of G by normalizing G has no practical interest. We are interested in the complexity of functions. In this section, we will define subsystems of linear logic by decorating formulae with labels and require them to verify some local criteria. Every proof-net G normalizes in at most $f_{|B_G|}(|\mathrm{E}_G|)$. For every LL proof-net G, the set of possible decorations to consider is finite. Thus, checking if there exists a decoration for G is decidable (even if searching labels this way is inefficient, because there may be more than x^x possible decorations with $x = |\mathrm{E}_G|$). Let us suppose that there exists a decoration of G such that the conclusion of G is $A \multimap B$, then for every (decorated) proof-net H of conclusion A, then $(G)H$. Let us also suppose that there exists $M_A \in \mathbb{N}$ such that, for every such proof-net H in normal form, $|B_G| \leq M_A$ (we noticed in Section 3.1.4 that it is a reasonable assumption). Then, $(G)H$ normalizes it at most $f_{|B_G|+M_A}(|\mathrm{E}_G|)$ steps, and we can notice that the function does not depend on H.

The first system we define is $SDNLL$, a type system enforcing the acyclicity of $\rightsquigarrow$, $\succ\!\!\!\rightarrow$ and $\prec\!\!\!\blacktriangleleft$ (Sections 3.2 and 3.3) than linear logic is. The ! and ? modalities are labeled with three indices s, d and n in $\mathbb{N}$. The labels represent respectively the depth of a box for $\rightsquigarrow$, $\succ\!\!\!\rightarrow$ and $\prec\!\!\!\blacktriangleleft$. This subsystem can be seen as a generalization of LLL, L^4 and of a maximal system of MS. The long term goal is to use our criteria on programming language outside of academics. Even if λ-calculus is not used directly it is closer to functionnal programming languages (such as Caml or Haskell). This is why we adapt $SDNLL$ to a type-system $SDNLL_\lambda$ for λ-calculus. By translating type derivations of $SNLL_\lambda$ into proof-nets of $SDNLL$, we prove the that $SDNLL_\lambda$ enjoys subject reduction and enforces a polynomial bound on β-reduction.

Then we define a subsystem $SwLL$ (based on the weak stratifications of Section 3.5) of linear logic generalizing $SDNLL$. One can embed SLL and every system of MS in $SwLL$. This subsystem is quite complex. The main purpose of $SwLL$ is to demonstrate that the improvements made from Section 3.3 to Section 3.5 on our semantic criteria, can be used by syntactical (and decidable) criteria.

5.1 Stratified Dependence control Nested Linear Logic

5.1.1 Definition of *S DNLL*

We define a *LL* subsystem, called *S DNLL* (for *S*tratification *D*ependence control *N*esting Linear Logic) characterizing *Poly*. In *S DNLL*, to enforce $\rightsquigarrow$-stratification[1], $\succ\!\!\!\rightarrow$-stratification[2] and $\prec\!\!\!\triangleleft$-stratification[3], we label the ! and ? modalities with integers s, d and n. Let us consider a *S DNLL* proof-net G and boxes B and B' with $\beta_G(\sigma(B)) = !_{s,n,d}A$ and $\beta_G(\sigma_G(B')) = !_{s',d',n'}A'$. Then we will have the following implications: ($B \rightsquigarrow B'$ implies $s > s'$), ($B \succ\!\!\!\rightarrow B'$ implies $d > d'$) and ($B \prec\!\!\!\triangleleft B'$ implies $n > n'$). This implies that G is $\rightsquigarrow$-stratified, $\succ\!\!\!\rightarrow$-stratified and $\prec\!\!\!\triangleleft$-stratified.

Definition 153. *For $s \in \mathbb{N}$, we define $\mathcal{F}_s$ by the following grammar (with $t, d, n \in \mathbb{N}$, $t \geq s$ and X ranges over a countable set of variables). Notice that $\mathcal{F}_0 \supseteq \mathcal{F}_1 \supseteq \cdots$*

$$\mathcal{F}_s := X_t \mid X_t^{\perp} \mid \mathcal{F}_s \otimes \mathcal{F}_s \mid \mathcal{F}_s \parr \mathcal{F}_s \mid \forall X_t.\mathcal{F}_s \mid \exists X_t.\mathcal{F}_s \mid !_{t,d,n}\mathcal{F}_{t+1} \mid ?_{t,d,n}\mathcal{F}_{t+1}$$

For any formula of the shape $A = !_{s',d',n'}A'$, we write s_A for s', d_A for d' and n_A for n'. For $A \in \mathcal{F}_0$, s_A^{min} refers to the minimum $s \in \mathbb{N}$ such that $A = h[!_{s,\_,\_}]$. To gain expressivity, we define a subtyping relation $\leq$ on $\mathcal{F}_0$. The relation $\leq$, defined as the transitive closure of the following $\leq^1$ relation, follows the intuition that a connective $!_{s,d,n}$ in a formula means that this connective "comes" from a box B with stratum $s_{\rightsquigarrow}(B) \geq s$, $s_{\succ\!\!\!\rightarrow}(B) \geq d$ and $s_{\prec\!\!\!\triangleleft}(B) \geq n$.

$$A \leq^1 B \Leftrightarrow \begin{cases} \text{Either} & A = g[!_{s,d,n}D], B = g[!_{s',d',n'}D], s \geq s', d \geq d' \text{ and } n \geq n' \\ \text{Or} & A = g[?_{s,d,n}D], B = g[?_{s',d',n'}D], s \leq s', d \geq d' \text{ and } n \leq n' \end{cases}$$

Definition 154 (*S DNLL* proof-net)**.** *A S DNLL* proof-net *is a graph-like structure defined inductively by the graphs of Figure 5.1 (G and H being proof-nets). Every edge e is labelled by a $\mathcal{F}_0$ formula $\beta(e)$.*

In order to prove the *Poly* soundness of *S DNLL*, we first have to prove a technichal lemma. Whenever $((e, P), [!_t]@T) \rightsquigarrow^* ((f, Q), [!_u]@U)$, the formulae of e and f are related. To be more precise, we will prove that (if G does not contain any $\exists$ or $\forall$ node, we will deal with those later), $(\beta(e))_{|T} \leq (\beta(f))_{|U}$, with $A_{|T}$ defined as follows:

Definition 155. *Let A be a formula and T a trace, we define $A_{|T}$ by induction on A as follows: $A_{|[]} = A$, $(A \otimes B)_{|T.\otimes_l} = (A \parr B)_{|T.\parr_l} = A_{|T}$, $(A \otimes B)_{|T.\otimes_r} = (A \parr B)_{|T.\parr_r} = B_{|T}$, $(\forall X.A)_{|T.\forall} = (\exists X.A)_{|T.\exists} = (!_{s,d,n}A)_{|T.!_t} = (?_{s,d,n}A)_{|T.?_t} = A_{|T}$.*

Let us suppose that $((e, P), T) \rightsquigarrow ((f, Q), U)$ implies $\beta(e)_T \leq \beta(f)_U$ (it will be formally proved in Lemma 166). For instance, if $((e, P), T) \rightsquigarrow ((f, P), T.parr_r)$ then $\beta_G(e)$ and $\beta(f)$ are of the shape B and $A' \parr B'$ with $B \leq B'$. Let us notice that $\beta_G(f)_{T.\parr_r} = (B')_T \geq (B)_T = \beta_G(f)_T$. Let us notice that the $\hookrightarrow$ step breaks this property: if $\beta(\sigma_i(B)) = ?_{s,d,n}A$ and $\beta(\sigma(B)) = !_{s',d',n'}B$ then A and B are a priori unrelated. The only relation required on those formulae is that $d \geq d'$ and $n \geq n'$. Thus, whenever $((\sigma(B), P), [!_t]) \mapsto^* ((\sigma(B'), P'), [!_{t'}])$ with $\beta(\sigma(B)) = !_{s,d,n}A$ and $\beta(\sigma(B')) = !_{s',d',n'}A'$, we have $d \geq d'$ and $n \geq n'$ (it will be formally proved in Lemma 167).

To understand why every *S DNLL* proof-net is $\rightsquigarrow$-stratified, $\succ\!\!\!\rightarrow$-stratified and $\prec\!\!\!\triangleleft$-stratified, we consider boxes B and B' with $\sigma(B) = !_{s,d,n}A$ and $\sigma(B') = !_{s',d',n'}A'$.

[1] Defined in Section 3.2
[2] Defined in Section 3.3.1
[3] Defined in Section 3.3.2

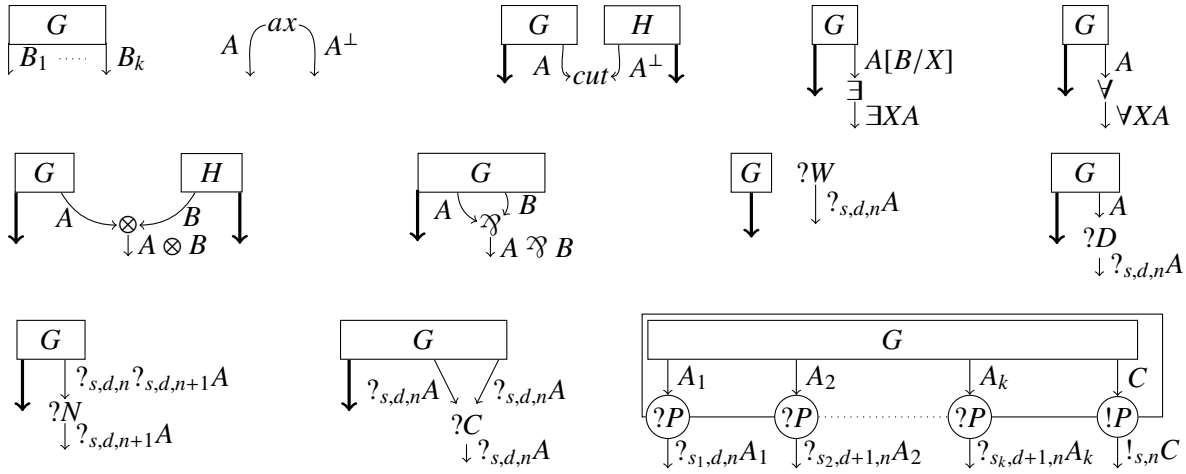


Figure 5.1: Construction of *SDNLL* proof-nets. For the ∀ rule, we require X not to be free in the formulae labelling the other conclusions of G. For the ∃ rule, we require $s_B^{min} \geq s$. For the top-left rule, we suppose G is a proof-net of conclusions $A_1,\ldots,A_k$ and $\forall i, A_i \leq B_i$.

Let us suppose that $B \rightsquigarrow B'$, then there exists a path $((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(B')},P'),[!_v]@U.?_u)$. So, $!_{s,d,n}A = (!_{s,d,n}A)_{[]} \leq (\beta_G(f))_{U.?_u}$, which means that $A' = h[?_{t,e,o}B^\perp]$ with $!_{s,d,n}A \leq !_{t,e,o}B$. By definition of $\leq$, $s \geq t$. And, by definition of $\mathcal{F}_0$, $s' < t$. Thus $s > s'$.

Let us suppose that $B \succ B'$. Then there exists paths of the shape $((\sigma(B),P),[!_t]) \mapsto^+ ((\overline{\sigma_i(B')},P'_1),[!_e])$ and $((\sigma(B),P),[!_u]) \mapsto^+ ((\overline{\sigma_j(B')},P'_2),[!_e])$ with $i \neq j$. Let $?_{\_,d_i,\_}A_i = \beta(\sigma_i(B))$ and $?_{\_,d_j,\_}A_j = \beta(\sigma_j(B))$, then $d \geq d_i$ and $d \geq d_j$. Either $i \neq 1$ (so $d \geq d_i > d'$ by definition of SDNLL) or $j \neq 1$ (so $d \geq d_j > d'$ by definition of SDNLL).

Let us suppose that $B \prec B'$, then there exists a non-standard signature v such that $((\sigma(B),P),[!_v]) \mapsto^* ((\sigma(B'),P'),[!_e])$. Let us consider the first context of the path whose leftmost trace is element is standard. Thus, there exists an edge e which is the conclusion of a $?N$ node such that $((\sigma(B),P),[!_v]) \mapsto^* ((\overline{e},Q),[!_{p(w)}]) \mapsto^* ((\overline{f},Q),[!_w]) \mapsto^* ((\sigma(B'),P'),[!_e])$. Let $?_{s_e,d_e,n_e}A_e = \beta(e)$ and $?_{s_f,d_f,n_f}A_f = \beta(f)$. Then, $n \geq n_e$ and $n_f \geq n'$. Moreover, because of the *SDNLL* constraints, $n_e > n_f$ so $n > n'$.

5.1.2 Underlying Formula

5.1.2.1 Intuition of the proof

The idea of this subsection is to prove that, if a context $((e,P),T) \mapsto^* C$, then C still has information about the formula $\beta(e)$. We can define an underlying formula for the contexts, and this underlying formula will be stable by $\rightsquigarrow$ in typed proof-nets. A consequence from this is that, if we begin a path by a context with a "right trace", then this path will never be blocked by a mismatch between the trace and the type of link above our edge (the tail of our edge). For example we will never have $((\overline{e},P),T.\forall)$ with the tail of e being a ⊗ link. However, it does not prevent blocking situations such as $((\overline{e},P),T.!_e)$ with the tail of e being a $?C$ link.

Our first idea to define the underlying formula of $((e,P),T)$ would be $(\beta_G(e))_T$. For example the underlying formula of $((e,P),[!_{1(e)};⅋_r;\forall])$ with $\beta(e) = \forall X.?(X \otimes X^\perp) ⅋ !(X^\perp ⅋ X)$ would be $X^\perp ⅋ X$. Let us remember that, if we defined $\beta(\overline{e})$ as $\beta(e)^\perp$. Thus, if $((e,P),T) \rightsquigarrow ((\overline{f},P),T)$ by crossing a *cut* node, $\beta(e) = \beta(\overline{f})$ and $\beta(e)_T = \beta(\overline{f})_T$.

However, there is a problem with this definition when we cross a $\exists$ link downwards. For example if $((c,[\,]),[⅋_r;!_e;\otimes_r]) \mapsto ((d,[\,]),[⅋_r;!_e;\otimes_r;\exists])$ with $\beta(c) = ?(X \otimes X^\perp) \otimes !(X^\perp ⅋ X)$ and $\beta(d) = \exists Y.Y \otimes Y^\perp$. Then $\beta(c)_{[⅋_r;!_e;\otimes_r]} = X$, but $\beta(d)_{[⅋_r;!_e;\otimes_r;\exists]}$ is undefined: the trace is not compatible with the syntactic tree of $\beta(d)$. We will have to pay attention to the substitution caused by the $\forall/\exists$ *cut*-elimination.

Let us recall that $\mathrm{N}_G^\forall$ (resp. $\mathrm{N}_G^\exists$) represents the set of $\forall$ nodes (resp. $\exists$ nodes) of G. We define a mapping ϕ_G from $Pot(\mathrm{N}_G^\forall)$ to $\mathcal{F}_0 \times Pot(\mathrm{N}_G^\exists)$. Intuitively, if $\phi_G(l,P) = (A,(m,Q))$, the node residues corresponding to (l,P) and (m,Q) are cut together, and A is the formula replaced by m. More formally, if l is a $\forall$ node, we define $\phi_G(l,P)$ by:

$$\phi_G(l,P) = (A,(m,Q)) \Leftrightarrow \begin{cases} m \text{ is a } \exists \text{ node and its associated formula is } A \\ ((concl_l,P),[\forall]) \mapsto^* ((\overline{concl_m},Q),[\forall]) \end{cases}$$

Let us notice that if $\phi_G(l,P) = (A,(m,Q))$, A might itself contain variables which will be susbstituted during reduction. This creates a chain of dependence of variables, described below:

Definition 156. *A $\forall/\exists$ dependence sequence is a sequence of maximal canonical nodes $(l_i,P_i)_{0\leq i\leq n}$ (with $n \geq 1$) such that*

- *For every $0 \leq 2i < n$, l_{2i} is a $\forall$ node and $\phi_G(l_{2i},P_{2i}) = (\_,(l_{2i+1},Q_{2i+1}))$.*
- *For every $0 \leq 2i+1 < n$, l_{2i+1} is a $\exists$ node, and if we set X_{2i+2} as the variable associated to l_{2i+2}, then X_{2i+2} is a free variable of the formula associated to l_{2i+1}, and P_{2i+2} is a prefix of P_{2i+1}.*

We define the relation $<_\forall$ on maximal canonical $\forall$ and $\exists$ nodes by $(l,P) <_\forall (l',P')$ if and only if there exists a $\forall/\exists$ dependence sequence $(l_i,P_i)_{0\leq i\leq n}$ with $(l_0,P_0) = (l,P)$ and $(l_n,P_n) = (l',P')$.

Lemma 157. *If G is a normalizing proof-net, then $<_\forall$ is a strict order on maximal canonical nodes of G.*[4]

Proof. The transitivity of $<_\forall$ is trivial, so we only have to prove its irreflexivity. It is enough to prove that there is no maximal canonical $\forall$ node (l,P) such that $(l,P) <_\forall (l,P)$: if there exists a maximal canonical $\exists$ node $(m,Q) <_\forall (m,Q)$, then there exists a maximal canonical $\forall$ node (l,P) such that $((l,P),[\forall]) \mapsto^* ((\overline{m},Q),[\forall])$ and $(l,P) <_\forall (l,P)$.

The proof is done by induction on the length of the longest normalizing sequence of G. We consider a $\forall/\exists$ dependence sequence $[(l_0,P_0);(m_0,Q_0);(l_1,P_1);(m_1,Q_1);\cdots;(l_n,P_n)]$. For $0 \leq i \leq n$ we set X_i as the variable associated with l_i and C_i as the formula associated with m_i.

- If every *cut* node of G has a premise which is the conclusion of a $?W$ node, then for every maximal canonical node (l,P) with l a $\forall$ node, $\phi_G(l,P)$ is undefined.
- Let us suppose that $G \to_{cut} G'$ and this reduction step is neither a $\forall/\exists$ reduction step nor a $!P/?W$ reduction step.

  Then for every $0 \leq i \leq n$, there exists a maximal canonical $\forall$ node (l'_i,P'_i) and a maximal canonical $\exists$ node (m'_i,Q'_i) such that $\pi_{G\to G'}((concl_{l'_i},P'_i),[\forall]) = ((concl_{l_i},P_i),[\forall])$ and $\pi_{G\to G'}((\overline{concl_{l'_i}},Q'_i),[\forall]) = ((\overline{concl_{l_i}},Q_i),[\forall])$. Moreover, let us notice that if X_i is a free variable of the formula C_i associated to l_i and is associated with l_{i+1}, and P_{i+1} is a prefix of Q_i, then the formula associated to l'_i is also equal to C_i and the $\forall$ node associated to X_i is (l'_{i+1},P'_{i+1}). For this last property, notice that when l_i has several residues (if $G \to_{cut} G'$ is a $!P/?C$ reduction reducing a box B containing l_i) knowing that

[4] Let us recall that maximal canonical nodes are defined in Definition 26

$\pi_{G\to G'}((l'_{i+1}, P'_{i+1}), [\forall]) = ((l_i, P_i), [\forall])$ and $\pi_{G\to G'}((m'_i, Q'_i), [\exists]) = ((m_i, Q_i), [\exists])$ is not enough. Indeed l'_{i+1} might be the "left" residue of l_i and m'_i the "right" residue of m_i. However this is ruled out because we know that P_{i+1} is a prefix of Q_i.

$(l'_i, P'_i)_{0\le i\le n}$ and $(m'_i, Q'_i)_{0\le i\le n}$ satisfy the hypotheses of the lemma so, by induction hypothesis, $(l'_1, P'_1) \neq (l'_n, P'_n)$. Thus $(l_1, P_1) \neq (l_n, P_n)$.

- Let us suppose that every *cut* node is either a $\forall/\exists$ or $?W/!P$ cut. Let us consider, in the construction of G, the last step of the shape $H = \overset{\vec{G_1}}{\underset{\searrow}{\forall}}\ \overset{\vec{G_2}}{\underset{\swarrow}{\exists}}\ _{cut}$. Let us name c, l and m the *cut*, $\forall$ and $\exists$ nodes of the figure. And let G' be the proof-net obtained by reducing c.

  We suppose that $l_1 = l_n$ and is a minimal such path (for $0 < i < j \le n$, $l_i \neq l_j$) and we will find a contradiction.

  If $l_1 \neq l$, then we can prove the lemma similarly to the case where the $G \to_{cut} G'$ step is not a $\forall/\exists$ step. We only have to be careful whenever $l_i = l$ (with $0 < i < n$). In this case, let (m'_{i-1}, Q'_{i-1}) and (l'_{i+1}, P'_{i+1}) such that $\pi_{G\to G'}((\overline{concl_{m'_{i-1}}}, Q'_{i-1}), [\forall]) = ((\overline{concl_{m_{i-1}}}, Q_{i-1}), [\forall])$ and $\pi_{G\to G'}((concl_{l'_{i+1}}, P'_{i+1}), [\forall]) = ((concl_{l_{i+1}}, P_{i-1}), [\forall])$. By assumption X_i is a free variable of C_{i-1} and X_{i+1} is a free variable of C_i. Let us observe that the formula associated with m'_{i-1} is $C_{i-1}[C_i/X_i]$. Moreover, P_i is a prefix of Q_{i-1} and P_{i+1} is a prefix of $Q_i = P_i$ so P_{i+1} is a prefix of Q_{i-1}. Thus, $(l'_j, P'_j)_{\substack{0\le j\le n\\ l_j\neq l}}$ and $(m'_j, Q'_j)_{\substack{0\le j\le n\\ m_j\neq m}}$ are sequences of maximal canonical nodes of G' satisfying the hypotheses of the lemma. By induction hypothesis, $l'_1 \neq l'_n$ so $l_1 \neq l_n$.

  If $l_1 = l$ then we can prove that for every $0 \le i \le n$, l_i and m_i are nodes of H. Indeed,

  - Let us suppose that m_i is in H. We know that $((concl_{l_i}, P_i), [\forall]) \mapsto^* ((\overline{concl_{m_i}}, Q_i), [\forall])$. Let us suppose that there is a context $((e, R), T)$ in the path with $e \notin H$, let us consider the last such context. The edge e is upward and $concl_{l_i}$ is downward so the $\mapsto$ path from $((concl_{l_i}, P_i), [\forall])$ to $((e, R), T)$ crosses a *cut* node. The last such *cut* node is under e so, it is a $!P/?W$ cut. It is a contradiction because the $\mapsto$ should either begin or stop at a $?W$ node. So $((concl_{l_i}, P_i), [\forall]) \mapsto^* ((\overline{concl_{m_i}}, Q_i), [\forall])$ is a path of H. (and in particular m_i is a node in H).
  - Let us suppose that l_i is in H. We know that the variable corresponding to l_i is a free variable of the formula corresponding to m_{i-1}. So the rule creating l_i is under the rule creating m_{i-1}, m_{i-1} is a node of H.

  We can normalize G_1 and G_2 to respectively G'_1 and G'_2. The reduction steps do not break the cycle, the only difficult steps are $\forall/\exists$ reducing l_j with m_j (as in the case where we supposed that $l_1 \neq l$). In this case, let (m'_{j-1}, Q'_{j-1}) and (l'_{j+1}, P'_{j+1}) such that $\pi_{G\to G'}((\overline{concl_{m'_{j-1}}}, Q'_{j-1}), [\forall]) = ((\overline{concl_{m_{j-1}}}, Q_{j-1}), [\forall])$ and $\pi_{G\to G'}((concl_{l'_{j+1}}, P'_{j+1}), [\forall]) = ((concl_{l_{j+1}}, P_{j-1}), [\forall])$. By assumption X_j is a free variable of C_{j-1} and X_{j+1} is a free variable of C_j. Let us observe that the formula associated with m'_{j-1} is $C_{j-1}[C_j/X_j]$. Moreover, P_j is a prefix of Q_{j-1} and P_{j+1} is a prefix of $Q_j = P_j$ so P_{j+1} is a prefix of Q_{j-1}.

  So there exist sequences $(l'_j, P'_j)_{0\le j\le k}$ and $(m'_j, Q'_j)_{0\le j\le k}$ of maximal canonical nodes of $G'' = \overset{\vec{G'_1}}{\underset{\searrow}{\forall}}\ \overset{\vec{G'_2}}{\underset{\swarrow}{\exists}}\ _{cut}$ satisfying the hypotheses of the lemma, with $l'_0 = l'_k = l$. We know that the variable corresponding to l_1 is a free variable of the formula corresponding to $m'_0 = m$. So the rule creating l'_1 is under the rule creating m. We know that $((concl_{l'_1}, P'_1), [\forall]) \mapsto^* ((\overline{m'_1}, Q'_1), [\forall])$. Let us consider the first *cut* node of

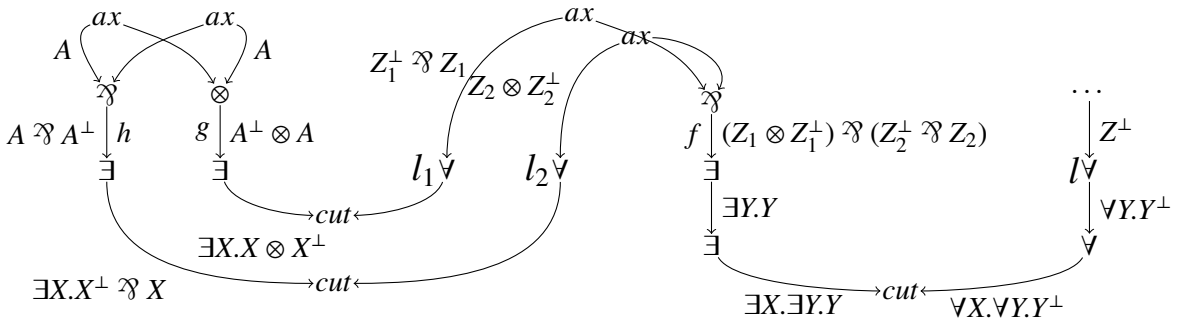


Figure 5.2:

$$\begin{aligned}\phi_G^{lim}(l,[]) &= (Z_1 \otimes Z_1^\perp) \parr (Z_2^\perp \parr Z_2)[\phi_G^{lim}(l_1,[])/Z_1; \phi_G^{lim}(l_2,[])/Z_2]\\ \phi_G^{lim}(l,[]) &= (Z_1 \otimes Z_1^\perp) \parr (Z_2^\perp \parr Z_2)[A^\perp/Z_1; A^\perp/Z_2]\\ \phi_G^{lim}(l,[]) &= (A^\perp \otimes A) \parr (A \parr A^\perp)\end{aligned}$$

the path. Because G'_2 is in normal form, the *cut* can not be in G'_2. And because of our assumptions, it must be in G'. So the first *cut* of the path is the *cut* node between l and m. So the rule creating l'_1 is above the rule creating m. This is a contradiction, so our assumption is wrong. There is no cycle.

□

If l is a $\exists$ node then $\phi_G^{lim}(l,P)$ is a formula A such that there is a residue l' of (l,P) whose associated formula is A. If l is a $\forall$ node then $\phi_G^{lim}(l,P)$ is a formula A such that there is a residue l' of (l,P) which is cut with a $\exists$ node m' whose associated formula is A. Finally, if (e,P) is a potential edge then $\theta(e,P)$ is the substitution which associates to every variable X in $\beta(e)$ the formula which will replace X along reduction.

Definition 158. *We define ϕ_G^{lim} as the mapping from $Pot(\mathrm{N}_G^\forall)$ to $\mathcal{F}_0$ by induction on $s_{>_\forall}(l,P)$:*

- *If $\phi_G(l,P)$ is undefined then $\phi_G^{lim}(l,P) = X$ (with X the variable associated with X).*
- *If $\phi_G(l,P) = (A,(m,[q_1;\cdots;q_{\partial(m)}]))$, $Z_1,\cdots,Z_k$ are the free variables of A which are associated to a $\forall$ node (which we write respectively $l_1,\cdots,l_k$) then:*

$$\phi_G^{lim}(l,P) = A[\phi_G^{lim}(l_1,[q_1;\cdots;q_{\partial(l_1)}])/Z_1;\cdots;\phi_G^{lim}(l_k,[q_1;\cdots;q_{\partial(l_k)}])/Z_k]$$

Definition 159. *Let $(e,[p_1;\cdots;p_{\partial(e)}]) \in Pot(\vec{\mathrm{E}}_G)$, $Z_1,\cdots,Z_k$ be the free variables of A which are associated to a $\forall$ node (which we write respectively $l_1,\cdots,l_k$) then:*

$$\theta(e,[p_1;\cdots;p_{\partial(e)}]) = \left\{Z_1 \mapsto \phi_G^{lim}\left(l_1,[p_1;\cdots;p_{\partial(l_1)}]\right),\cdots,Z_k \mapsto \phi_G^{lim}\left(l_k,[p_1;\cdots;p_{\partial(l_k)}]\right)\right\}$$

Definition 160. *We extend the mapping ϕ_G^{lim} on potential $\mathrm{N}_G^\exists$ as follows. Let $(l,P) \in Pot(\mathrm{N}_G^\exists)$ with B the formula associated to l, $\phi_G^{lim}(l,P) = B[\theta(prem_l,P)]$.*

Lemma 161. *If G is a normalizing proof-net, then there exists $M \in \mathbb{N}$ such that for every maximal canonical node (l,P) with l a $\forall$ node, the depth of $\phi_G^{lim}(l,P)$ is at most M.*

Proof. Let D be the maximum depth of formulae associated to $\exists$ node. One can prove by induction on the depth of (l,P) for the $<_\forall$ order (written $s_{<_\forall}(l,P)$) that the depth of $\phi_G^{lim} l,P$ is at most $D \cdot s_{<_\forall}(l,P)$. So we can define M as $D \cdot |MaxCan(\mathrm{E}_G)|$. □

We will define the notion of underlying formulae on objects a bit different from contexts, because it is easier to make an induction definition on these.

Definition 162. *Let $(e, P) \in Pot(E_G)$, $A \in \mathcal{F}_0$ and T, T' be two lists of trace elements (traces or empty list) and p a polarity, $\beta(A, e, P, T, T')$ is defined by induction on $|T|$ by:*

$$\beta(A, e, P, [], T) = A$$
$$\beta(A \otimes B, e, P, T.\otimes_l, U) = \beta(A, e, P, T, [\otimes_l]@U) \otimes B \qquad \beta(A \otimes B, e, P, T.\otimes_r, U) = A \otimes \beta(B, e, P, T, [\otimes_r]@U)$$
$$\beta(A \parr B, e, P, T.\parr_l, U) = \beta(A, e, P, T, [\parr_l]@U) \parr B \qquad \beta(A \parr B, e, P, T.\parr_r, U) = A \parr \beta(B, e, P, T, [\parr_r]@U)$$
$$\beta(!_{s,d,n}A, e, P, T.!_t, U) = !_{s,d,n}\beta(A, e, P, T, [!_t]@U) \qquad \beta(?_{s,d,n}A, e, P, T.?_t, U) = ?_{s,d,n}\beta(A, e, P, T, [?_t]@U)$$

If l is a $\forall$ node and $((concl_l, Q), [\forall]) \rightsquigarrow^* ((e, P), [\forall]@U)$, then

$$\beta(\forall X.A, e, P, T.\forall, U) = \beta(A, e, P, T, [\forall]@U)[\phi_G^{lim}(l, Q)/X]$$

If l is a $\exists$ node, C is the formula associated to l and $((concl_l, Q), [\exists]) \rightsquigarrow^* ((e, P), [\exists]@T')$, then

$$\beta(\exists X.A, e, P, T.\exists, U) = \beta(A, e, P, T, [\exists]@U)[C[\theta(concl_l, Q)]/X]$$

Lemma 163. *For every $\rightsquigarrow$-path $((e, P), T) \rightsquigarrow^* ((f, Q), U)$ and for every list of trace elements V, the formula $\beta(\beta(e)[\theta(e, P)], e, P, V@T, [])_{|T}$ is defined if and only if $\beta(\beta(f)[\theta(f, Q)], f, Q, V@U, [])_{|T}$ is and, if these formulae are defined,*

$$\beta(\beta(e)[\theta(e, P)], e, P, V@T, [])_{|T} \leq \beta(\beta(f)[\theta(f, Q)], f, Q, V@U, [])_{|U}$$

Proof. We make a disjunction on the $\rightsquigarrow$ rule used.

- If the rule is neither a $\exists$ nor a $\forall$ rule, then the proof is technical but straightforward. The formulae $\beta(f)$ and $\beta(e)$ are almost the same. The only possible differences are a modification of indices (because of the subtyping relation) and the addition or deletion of the head connective and this modification is compensated by the modification of the trace. For instance, if we cross a $\parr$ node downwards, $\beta(e) = A$, there exists a formula B such that $\beta(f) \geq A \parr B$, $P = Q$ and $U = T.\parr_l$. Thus $\beta(f) = A' \parr B'$ with $A' \geq A$. Let us notice that the free variables of A are included in the free variables of B and $P = Q$ so $\theta(e, P) = \theta(f, Q)$. Thus,

$$\begin{aligned}
\beta(\beta(f)[\theta(f, Q)], f, Q, V@U, [\,])_{|U} &= \beta((A' \parr B')[\theta(f, Q)], f, P, V@T.\parr_l, [\,])_{|T.\parr_l} \\
&= (\beta(A'[\theta(f, Q)], f, P, V@T, [\parr_l]) \parr B'[\theta(f, Q)])_{|T.\parr_l} \\
&= \beta(A'[\theta(f, Q)], f, P, V@T, [\parr_l])_{|T} \\
\beta(\beta(f)[\theta(f, Q)], f, Q, V@U, [\,])_{|U} &\geq \beta(\beta(e)[\theta(e, P)], e, P, V@T, [])_{|T}
\end{aligned}$$

  For the last equality, notice that the only cases where the edge and the second trace are used are the quantifiers cases. They are used to determine the $\exists$ or $\forall$ node to consider with a $\rightsquigarrow$ path. We can notice that for every context C which is on the conclusion of a quantifier node, $C \rightsquigarrow^* ((e, P), U)$ if and only if $C \rightsquigarrow^* ((f, P), U.\parr_l)$.

In the case of crossing an $\exists$ link downward, $((e,P),T) \rightsquigarrow ((f,P),T.\exists)$. Then $\beta(e) = A[C/X]$ and $\beta(f) = \exists X.A'$ with $A \leq A'$.

$$\begin{aligned}
\beta(\beta(f)[\theta(f,P)],f,P,V@T.\exists,[\,])_{|U} &= \beta(\exists X.A'[\theta(f,P)],f,P,V@T.\exists,[\,])_{|T.\exists}\\
&= \beta(A'[\theta(f,P)],f,P,V@T,[\exists])(C[\theta(e,P)]/X])_{|T}\\
&= \beta(A'[\theta(f,P)][C[\theta(e,P)]/X],f,P,V@T,[\exists])_{|T}\\
&= \beta(A'[C/X][\theta(e,P)],f,P,V@T,[\exists])_{|T}\\
&= \beta(A'[C/X][\theta(e,P)],e,P,V@T,[\,])_{|T}\\
&\geq \beta(A[C/X][\theta(e,P)],e,P,V@T,[\,])_{|T}\\
\beta(\beta(f)[\theta(f,P)],f,P,V@T.\exists,[\,])_{|U} &\leq \beta(\beta(e)[\theta(e,P)],e,P,T,[\,])_{|T}
\end{aligned}$$

In the case of crossing a $\forall$ node l downward. $((e,P),T) \rightsquigarrow ((f,P),T.\forall)$. Then $\beta(e) = A$ and $\beta(f) = \forall X.A'$. Let us notice that the free variables of $\beta(e)$ are exactly the free variables of $\beta(f)$ and X. Moreover (e,P) and (f,P) belong to the same potential boxes, so $\theta(e,P) = \phi_G^{lim}(l,P) \circ \theta(f,P)$.

$$\begin{aligned}
\beta(\beta(f)[\theta(f,P)],f,P,V@T.\forall,[\,])_{|U} &= \beta(\forall X.A'[\theta(f,P)],f,P,V@T.\forall,[\,])_{|T.\forall}\\
&= \beta(A'[\theta(fe,P)],f,P,V@T,[\exists])_{|T}[\phi_G^{lim}(l,P)]\\
&= \beta(A'[\theta(f,P)][\phi_G^{lim}(l,P)],f,P,V@T,[\exists])_{|T}\\
&= \beta(A'[\theta(e,P)],f,P,V@T,[\exists])_{|T}\\
&\geq \beta(A[\theta(e,P)],e,P,V@T,[\,])_{|T}\\
\beta(\beta(f)[\theta(f,P)],f,P,V@T.\forall,[\,])_{|U} &\geq \beta(\beta(e)[\theta(e,P)],e,P,V@T,[\,])_{|T}
\end{aligned}$$

□

Whenever the trace begins by a $!_{\_}$ trace element, we have a slightly better result. Let us suppose that $((e,P),[!_t]) \rightsquigarrow^* ((f,Q),[!_u])$ with $\beta_G(e) = !_{s,d,n}A$ and $\beta_G(f) = !_{s',d',n'}A'$. Then, by Lemma 163, $\beta_G(e)_{|!_t} = A \leq A' = \beta_G(f)_{|!_u}$. In fact, examining closely the rules of $\rightsquigarrow$, one can deduce that $s \geq s'$, $d \geq d'$ and $n \geq n'$, so $\beta_G(e)_{|[]} \leq \beta_G(f)_{|[]}$. The following Lemma is the generalization of this statement.

Lemma 164. *If $((e,P),[!_t]@T) \rightsquigarrow^* ((f,Q),[!_u]@U)$ then for every list of trace elements V, the formula $\beta(\beta(e)[\theta(e,P)],e,P,V@T,[])_{|T}$ is defined if and only if $\beta(\beta(f)[\theta(f,Q)],f,Q,V@U,[])_{|T}$ is and, if these formulae are defined,*

$$\beta(\beta(e)[\theta(e,P)],e,P,V@T,[])_{|T} \leq \beta(\beta(f)[\theta(f,Q)],f,Q,V@U,[])_{|T}$$

Proof. It is enough to prove it whenever the length of the path is 1, i.e. $((e,P),[!_t]@T) \rightsquigarrow ((f,Q),[!_u]@U)$. If $T \neq []$ and $U \neq []$, then $((e,P),T) \rightsquigarrow^* ((f,Q),U)$ so the result can be deduced from Lemma 163. Else, if $T \neq []$ or $U \neq []$ or the $\rightsquigarrow$ step crosses an *ax* or *cut* node, one can prove the result as in the first case of the proof of Lemma 163. The only interesting cases are whenever $T = []$, $U = []$ and the $\rightsquigarrow$ step crosses a node n upward whose label is in $\{?N, ?C\}$. In those cases, let us notice that the labels in the premise of the node ($\overline{f}$ in this case) are lower than the labels in the conclusion of the ($\overline{e}$ in this case). □

Definition 165. *For any context $((e,P),[!_t]@T)$, we write $\boldsymbol{\beta}((e,P),[!_t]@T)$ for $\beta(\beta(e)[\theta(e,P)],e,P,T,[])_{|T}$*

Lemma 166. *If $C \rightsquigarrow^* C'$, we have $\boldsymbol{\beta}(C) \leq \boldsymbol{\beta}(C')$.*

Proof. This is a direct consequence of Lemma 164. □

Lemma 167. *If $C \mapsto^* C'$ and $\boldsymbol{\beta}(C) = !_{\_,d,n}A$ then $\boldsymbol{\beta}(C') = !_{\_,d',n'}B$ with $d \geq d'$ and $n \geq n'$.*

Proof. For the $\rightsquigarrow$ steps, we can use Lemma 166. For the $\hookrightarrow$ steps, one can notice that, if $\beta(\sigma_i(B)) = ?_{s,d,n}A$ and $\beta(\sigma(B)) = !_{s',d',n'}B$ then $d \geq d'$ and $n \geq n'$. □

5.1.3 *S DNLL* is sound for *Poly*

Thanks to Lemmas 164 and 168, we will formalize the proofs we sketched in the end of Section 5.1.1.

Lemma 168. *If $C \mapsto^* C'$ then $\boldsymbol{\beta}(C)$ is defined if and only if $\boldsymbol{\beta}(C')$ is. If these formulae are defined and respectively equal to $!_{s,d,n}A$ and $!_{s',d',n'}A'$, then $d \geq d'$ and $n \geq n'$.*

Proof. Most of the cases can be deduced from Lemma 164. The only interesting case is $((e,P),[!_t]) \hookrightarrow ((f,P),[!_u])$. In this case $\overline{e}$ and f are an auxiliary edge and a principal edge of the same box. So the formula $\beta(\beta(e)[\theta(e,P)],e,P,[],[])_{|[]}$ is defined and equal to $\beta(e)[\theta(e,P)]$ and $\beta(\beta(f)[\theta(f,P)],f,P,[],[])_{|[]}$ is defined and equal to $\beta(f)[\theta(f,P)]$. From the constraints of *S DNLL*, we deduce that $d \geq d'$ and $n \geq n'$. □

Lemma 169. *If $B \rightsquigarrow\!\!> B'$, $\beta(\sigma(B)) = !_{s,d,n}A$ and $\beta(\sigma(B')) = !_{s',d',n'}A'$ then $s > s'$.*

Proof. Let us suppose that $B \rightsquigarrow\!\!> B'$, $\beta_G(\sigma(B)) = !_{s,d,n}A$ and $\beta_G(\sigma(B')) = !_{s',d',n'}A'$. Then there exist $P, P' \in Pot$ and $t, u \in Sig$ such that $((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(B')},P'),!_uT) \rightsquigarrow^* ((e,Q),[!_{\mathtt{e}}])$. We can notice that $\boldsymbol{\beta}((\sigma(B),P),[!_t]) = \beta(\sigma(B))[\theta(\sigma(B),P)] = !_{s,d,n}(A[\theta(\sigma(B),P)])$ so, according to Lemma 163, $\boldsymbol{\beta}((\overline{\sigma(B')},P'),[!_u]@T)$ is defined and

$$!_{s,d,n}(A[\theta(\sigma(B),P)]) \leq \boldsymbol{\beta}((\overline{\sigma(B')},P'),[!_u]@T)$$

We set $!_{s'',d'',n''}A'' = \boldsymbol{\beta}((\overline{\sigma(B')},P'),[!_u]@T)$ (let us notice that by definition of $\leq$, $s \geq s''$). Thus, $A'[\theta(\sigma(B'),P')]$ is of the shape $H[!_{s'',d'',n''}A'']$. Either A' is of the shape $H[!_{s'',d'',n''}A'']$ (in this case by definition of $\mathcal{F}_0$, $s' < s''$ so $s > s'$), or there exist sequences $A' = A_0, A_1 \cdots, A_k$ of formulae, $X^1, \cdots, X^k$ of variables and $s_1, \cdots, s_k$ such that for $0 \leq i < k$, A_i is of the shape $H_i[X^i_{s_i}]$ and there exists a $\exists$ node n_i whose associated variable is $X^i_{s_i}$ and whose associated formula is A_{i+1}. And A_k is of the shape $H_k[!_{s'',d'',n''}A'']$. In this case, $s' < s_0 \leq s_1 \leq \cdots \leq s_{k-1} \leq s''$ so $s' < s'' \leq s$. □

Lemma 170. *If $B \succ\!\!\rightarrow B'$, $\beta(\sigma(B)) = !_{s,d,n}A$ and $\beta(\sigma(B')) = !_{s',d',n'}A'$ then $d > d'$*

Proof. By definition of $\succ\!\!\rightarrow$, there exists $i \neq j$ and paths of the shape $((\sigma(B),P),[!_t]) \mapsto^+ ((\overline{\sigma_i(B')},P'_1)[!_{\mathtt{e}}])$ and $((\sigma(B),P),[!_u]) \mapsto^+ ((\overline{\sigma_j(B')},P'_2),[!_{\mathtt{e}}])$. Either $i \neq 1$ or $j \neq 1$. We suppose without loss of generality that $i \neq 1$. By definition of *S DNLL*, $\beta(\sigma_i(B)) = !_{\_,d'',\_}$ with $d'' > d'$. Then, by Lemma 168, $d \geq d'' > d'$. □

Lemma 171. *If $B \prec\!\!\!\cdot\, B'$, $\beta(\sigma(B)) = !_{s,d,n}A$ and $\beta(\sigma(B')) = !_{s',d',n'}A'$ then $n > n'$*

Proof. By definition of $\prec\!\!\!\cdot$, there exist $P, Q \in Pot$, $t \in Cop(B,P)$ and $v \sqsupset t$, such that $((\sigma(B),P),[!_v]) \mapsto^* ((\sigma(C),Q),[!_{\mathtt{e}}])$. Let us consider the first context of the path such that the leftmost trace element is $!_u$ with u a standard signature. This step must be of the shape $((e,R),[!_{\mathtt{p}(u)}]) \rightsquigarrow ((f,R),[!_u])$ with $\overline{e}$ the conclusion of a $?N$ node. By definition of *S DNLL*, $\beta(e) = ?_{\_,\_,n_e\_}$ and $\beta(f) = ?_{\_,\_,n_f\_}$ with $n_e > n_f$. Then, by Lemma 168, $n \geq n_e > n_f \geq n'$. □

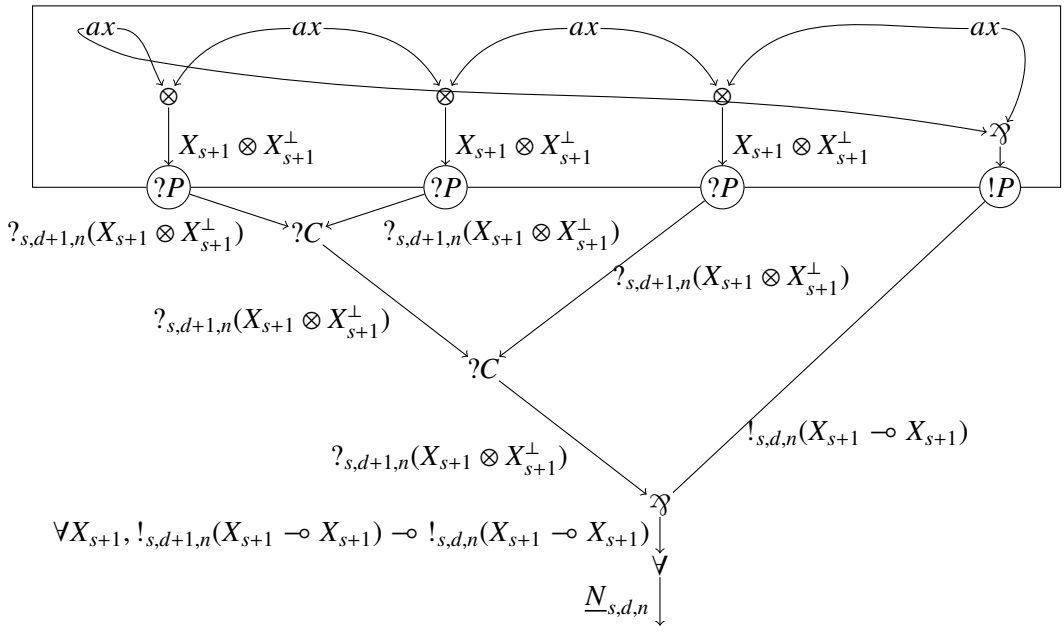


Figure 5.3: Encoding $Enc_{s,d,n}(3)$ of 3

Corollary 172. *Let G be a SDNLL proof-net, then G is $\rightsquigarrow$-stratified, $\succ$-stratified and $\lessdot$-stratified. Moreover, for every $B \in B_G$ with $\beta_G(\sigma(B)) = !_{s,d,n}A$, $s_{\rightsquigarrow}(B) \leq s$, $s_{\succ}(B) \leq d$ and $s_{\lessdot}(B) \leq n$.*

Proof. Immediate consequence of the three previous lemmas. □

Theorem 173. *Let G be a SDNLL proof-net, then the maximal reduction length of G (with $n = |\vec{E}_G|$, $\partial = \partial_G$, $S = \max_{B \in B_G} s_{\sigma(B)}$, $D = \max_{B \in B_G} d_{\sigma(B)}$ and $S = \max_{B \in B_G} n_{\sigma(B)}$) is bounded by*

$$W_G \leq n^{1+D^S \cdot \partial^{1+N \cdot S}}$$

Proof. The bound is an immediate consequence of Corollaries 172 and 87. □

To prove that *SDNLL* is *Poly* sound, we need to define an encoding of binary lists. For any $s, d, n \in \mathbb{N}$, we define the formulae $\underline{N}_{s,d,n}$ and $\underline{B}_{s,d,n}$ by

$$\underline{N}_{s,d,n} = \forall X_{s+1}, !_{s,d+1,n}(X_{s+1} \multimap X_{s+1}) \multimap !_{s,d,n}(X_{s+1} \multimap X_{s+1})$$
$$\underline{B}_{s,d,n} = \forall X_{s+1}, !_{s,d+1,n}(X_{s+1} \multimap X_{s+1}) \multimap !_{s,d+1,n}(X_{s+1} \multimap X_{s+1}) \multimap !_{s,d,n}(X_{s+1} \multimap X_{s+1})$$

Then for any $s, d, n \in \mathbb{N}$, $k \in \mathbb{N}$ and binary list l, we can define encodings $Enc_{s,d,n}(k)$ and $Enc_{s,d,n}(l)$ of k and l as shown in Figures 5.3 and 5.4. We can verify that the sizes of $Enc_{s,d,n}(k)$ and $Enc_{s,d,n}(l)$ depend linearly on the size of k and l. Finally, there is exactly one box in $Enc_{s,d,n}(k)$ and $Enc_{s,d,n}(l)$ so the encoding is box-bounded.

Theorem 174. *SDNLL is sound for Poly*

Proof. Immediate from Theorem 173, Theorem 88 and the box-boundedness of the encoding. □

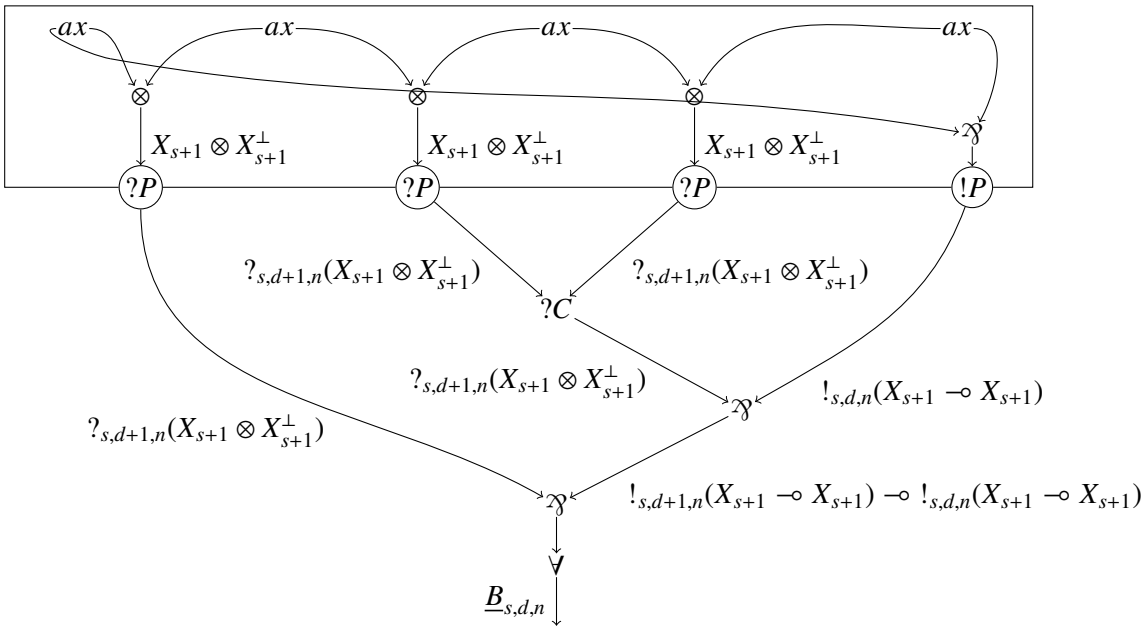


Figure 5.4: Encoding $Enc_{s,d,n}([0;1;1])$ of $[0;1;1]$

5.1.4 Encoding of light logics

There are already many subsystems of *LL* characterizing *Poly*. We argue that the interest of *SDNLL* over the previous systems is its intentional expressivity. To support our claim we define encodings of L^4 [8] and a maximal subsystem of *MS* [64] in L^4. Baillot and Mazza already proved that *LLL* can be embedded in L^4, thus *SDNLL* is at least as expressive as the union of those systems.

5.1.4.1 Encoding of L^4

The formulae of L^4 are defined as the formulae with an additional modality § and an element of $\mathbb{N}$ indexing the formula. More formally, the set $\mathcal{F}_{L4}$ of formulae of L^4 is defined by the following grammar.

$$\mathcal{F}_{L4} = \mathcal{G}_{L4} \times \mathbb{N}$$
$$\mathcal{G}_{L4} = X \mid X^{\perp} \mid \mathcal{G}_{L4} \otimes \mathcal{G}_{L4} \mid \mathcal{G}_{L4} \parr \mathcal{G}_{L4} \mid \forall X.\mathcal{G}_{L4} \mid \exists X.\mathcal{G}_{L4} \mid !\mathcal{G}_{L4} \mid ?\mathcal{G}_{L4} \mid \S\mathcal{G}_{L4}$$

The index in $\mathbb{N}$ (called *level*) is usually written as an exponent. Intuitively, if the principal edge of B is labelled with $(!A)^s$, the label s represents the stratum of B for $\rightsquigarrow$. More precisely, it corresponds to a formula of the shape $!_{s+\partial(B),\_,\_}A$ in *SDNLL*. Let us notice that, to connect two boxes B and B' labelled with $(!A)^s$ and $(!A')^{s'}$ with $s \neq s'$, we need to use § nodes.

Let us notice that every box of L^4 proof-nets have only one auxiliary door. Thus $\succ\!\!\!\rightarrow = \varnothing$ and, for every box B $s_{\succ\!\!\!\rightarrow}(B) = 1$. We can also notice that there is no $?N$ node in L^4 proof-nets, so for every box B, we have $s_{\prec\!\!\!\rightarrow}(B) = 1$.

We define a mapping $\|\_\|$ from hole-formulae of $\mathcal{G}_{L4}$ to $\mathbb{N}$ which will be used to decide the indices of variables and exponential modalities. For every formulae A in $\mathcal{G}_{L4}$ and hole-formula H, $\|\circ\| = 0$, $\|C \otimes H\| = \|H \otimes C\| = \|C \parr H\| = \|H \parr C\| = \|H\|$, $\|\forall X.H\| = \|\exists X.H\| = \|H\|$, and $\|!H\| = \|?H\| = \|\S H\| = 1 + \|H\|$.

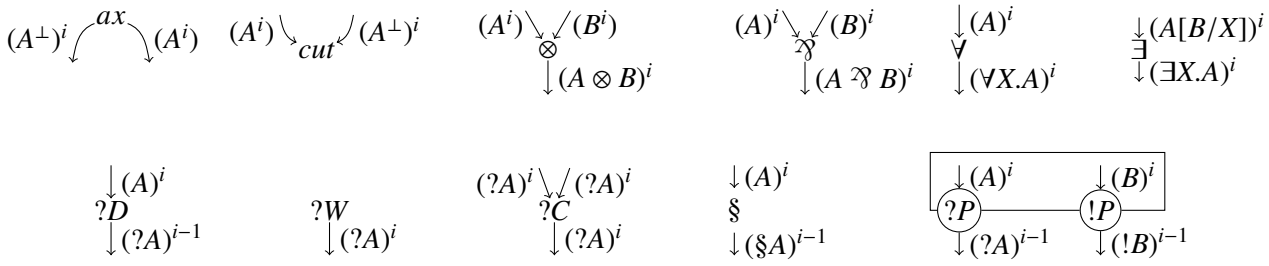


Figure 5.5: Relations between levels of neighbour edges in L^4. We also allow boxes with 0 auxiliary doors.

Any L^4 proof-net G can be transformed into a $SDNLL$ proof-net G' as follows:for every variable X appearing in the proof-net, we define M_X as the maximum of the set

$$\left\{ s + \|H\| \,\middle|\, \beta(e) = H[X] \text{ or } \beta(e) = (H[X^\perp])^s \right\}$$

Then, we replace every occurrence of X by X_{M_X}. If $\beta(e) = (H[!A])^s$ (resp. $(H[?A])^s$), we replace the modality by $!_{s+\|H\|,1,0}$ (resp. $?_{s+\|H\|,1,0}$). One can easily verify that G is a valid $SDNLL$ proof-net. The § node becomes trivial (it does not change the sequent). Let us consider the boxes:

- Let us suppose that e is the premise of the i-th auxiliary door of a box B and f is its conclusion. If $\beta_G(e) = (H[!A])^s$ then $\beta_G(f) = ?(H[!A])^{s-1}$. We can notice that $\beta_{G'}(e) = H'[!_{s+\|H\|,1,0\_}]$ and $\beta_{G'}(f) = H'[!_{s-1+\|?H\|,1,0\_}]$. Those labels are the same because $s + \|H\| = (s-1) + (1 + \|H\|) = (s-1) + \|?H\|$.

- If we have $\beta_{G'}(\sigma_1(B)) = ?_{s,d,n}A$ and $\beta_{G'}(\sigma(B)) = !_{s',d',n'}A'$ then we have $d \geq d'$. Indeed: either B is a ! box in G and $d = 1 \geq 1 = d'$, or B is a § box in G and $d \geq 0 = d'$.ie

5.1.4.2 Encoding of a maximum subsystem of MS

In [64], Roversi and Vercelli proposed to generalize the "one auxiliary door" condition by considering a framework of logics: MS. MS is defined as a set of subsystems of ELL with indexes on ! and ? connectives. The formulae of MS are defined as the formulae of LL with a label $d \in \mathbb{N}$ on ! and ? modalities. More formally, the set $\mathcal{F}_{MS}$ of formulae of MS is defined by the following grammar, with d ranging over $\mathbb{N}$.

$$\mathcal{F}_{MS} = X \mid X^\perp \mid \mathcal{F}_{MS} \otimes \mathcal{F}_{MS} \mid \mathcal{F}_{MS} \parr \mathcal{F}_{MS} \mid \forall X.\mathcal{F}_{MS} \mid \exists X.\mathcal{F}_{MS} \mid !_d\mathcal{F}_{MS} \mid ?_d\mathcal{F}_{MS}$$

Roversi and Vercelli provide a characterization of the MS systems which are sound for *Poly*. This criterion says that a MS system is sound for *Poly* if and only if for every $k \in \mathbb{N}$, one of the two following condition holds:

- If $?_iA$ and $?_jA$ can be contracted in $?_kA$, then $k \leq i$, $k \leq j$ **and at least one of those inequality is strict**. And for every box whose principal door is indexed by $!_kA$, the indexes on the ?-s of the auxiliary doors are smaller or equal to k.

- If $?_iA$ and $?_jA$ can be contracted in $?_kA$, then $k \leq i$, $k \leq j$. And for every box whose principal door is indexed by $!_kA$, the indexes on the ?-s of the auxiliary doors are smaller or equal to k **with all but (at most) one of those inequalities being strict**.

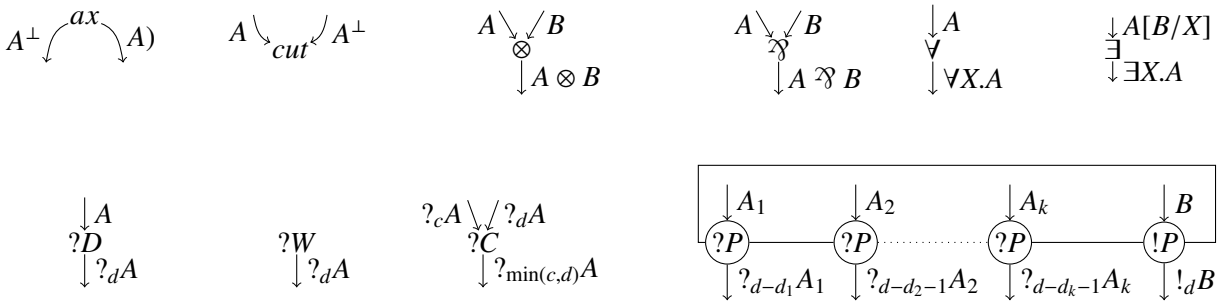


Figure 5.6: Constraints of MS_{max} on the labels of exponential connectives.

According to this characterization, the subsystem MS_{max} defined in Figure 5.6 is sound for *Poly* (we are in the second case).

Following the intuitions given above, we define a mapping $\|\_\|$ from hole-formulae of MS_{max} to $\mathbb{N}$ which is similar to the mapping defined on the hole-formulae of L^4. For every formulae A in $\mathcal{F}_{MS_{max}}$ and hole-formula H, $\|\circ\| = 0$, $\|C \otimes H\| = \|H \otimes C\| = \|C \parr H\| = \|H \parr C\| = \|H\|$, $\|\forall X.H\| = \|\exists X.H\| = \|H\|$, and $\|!_d H\| = \|?_d H\| = 1 + \|H\|$.

Let us consider a MS proof-net G, then we transform it into a $SDNLL$ proof-net G' as follows. First let us set D as the maximul label of ? and ! labels in G. And, for every variable X appearing in the proof-net, we define M_X as the maximum of the set

$$\left\{\partial(e) + \|H\| \;\middle|\; (\beta(e) = H[X] \text{ or } \beta(e) = H[X^\perp])\right\}$$

Then, we replace every occurrence of X be X_{M_X}. If $\beta(e) = H[!_d A]$ (resp. $H[?_d A]$) and $s = \partial(e) + \|H\|$, we replace the modality by $!_{s,D-d,0}$ (resp. $?_{s,D-d,0}$).

To prove that the proof-net G' obtained is in $SDNLL$, most of the conditions are straightforward to check. For example,

- Let us suppose that e is the premise of the i-th auxiliary door of a box B and f is its conclusion. If $\beta_G(e) = H[!A]$ then $\beta_G(f) = ?H[!A]$. We can notice that $\beta_{G'}(e) = H'[!_{\partial(e)+\|H\|,1,0\_}]$ and $\beta_{G'}(f) = H'[!_{\partial(f)+\|?H\|,1,0\_}]$. Those labels are the same because $\partial(e) + \|H\| = \partial(f) + 1 + \|H\| = \partial(f) + \|?H\|$.

- Let us suppose that $\beta_{G'}(\sigma_1(B)) = ?_{\_,d'_1,\_}A$ and $\beta_{G'}(\sigma(B)) = !_{\_,d',\_}$. Then we can notice that d'_1 and d' are of the shape $d'_1 = D - (d - d_1)$ and $d' = D - d$. Thus, $d'_1 \geq d'$.

- Let us suppose that $\beta_{G'}(\sigma_i(B)) = ?_{\_,d'_i,\_}$ and $\beta_{G'}(\sigma(B)) = !_{\_,d',\_}$ with $i > 1$ then we can notice that d'_i and d' are of the shape $d'_i = D - (d - d_i - 1)$ and $d' = D - d$. Thus $d'_i \geq D - d + 1 > d'$.

Concerning the other LL subsystems characterizing *Poly*: L^4_0 [8], SLL, BLL and $QBAL$ contain proof-net which are not $\rightsquigarrow$-stratified, so which are not in $SDNLL$. However, no λ-term typable in L^4_0 and untypable in $SDNLL$ has been found yet. Even if SLL is orthogonal to LLL, SLL is generally thought to be less expressive than LLL. Finally, type-checking is undecidable in BLL and $QBAL$ so they do not fit in our goal of automatic bound inferring.

Concerning intensional expressivity, we are more interested in λ-calculus than in proof-nets. The subsystems SLL and LLL of linear logic have been transformed into type systems of λ-calculus[12, 33]. We conjecture that MS, L^4 and $SDNLL$ could be transformed similarly into type systems for λ-calculus enforcing polynomial time bounds, such that the embeddings between LL systems become embeddings between

$$\frac{}{x : A^{\varnothing} \vdash x : A}\, ax \qquad \frac{\Gamma \vdash t : A \qquad X_s \text{ is not free in } \Gamma}{\Gamma \vdash t : \forall X_s.A} \qquad \frac{\Gamma \vdash t : \forall X.A \qquad s_B^{min} \geq s}{\Gamma \vdash t : A[B/X]}$$

$$\frac{\Gamma, x : A^{\varnothing} \vdash t : B}{\Gamma, x : A^{s,d,n} \vdash t : B}\, ?D \qquad \frac{\Gamma \vdash t : B}{\Gamma, x : A^{s,d,n} \vdash t : B}\, ?W \qquad \frac{\Gamma, y : A^{s,d,n}, z : A^{s,d,n} \vdash t : B}{\Gamma, x : A^{s,d,n} \vdash t[x/y; x/z] : B}\, ?C$$

$$\frac{\Gamma, x : A^{\varnothing} \vdash t : B}{\Gamma \vdash \lambda x.t : A \multimap B}\, \multimap_i \qquad \frac{\Gamma, x : A^{s,d,n} \vdash t : B}{\Gamma \vdash \lambda x.t : !_{s,d,n}A \multimap B}\, \Rightarrow_i \qquad \frac{\Gamma \vdash t : A \multimap B \qquad \Delta \vdash u : A}{\Gamma, \Delta \vdash (t)u : B}\, \multimap_e$$

$$\frac{\Gamma \vdash t : !_{s,d,n}A \multimap B \qquad \Delta, \Sigma^{\varnothing} \vdash u : A \qquad d(\Delta \cup \Sigma) \geqq d \;\; n(\Sigma) \geq n \;\; n(\Delta) > n}{\Gamma, \Delta, \Sigma \vdash (t)u : B}\, \Rightarrow_e$$

Figure 5.7: $SDNLL_\lambda$ as a λ-calculus type-system.

the corresponding type systems for λ-calculus. In Section 5.1.5, we define a type system $SDNLL_\lambda$ inspired by *DLAL*. However the embedding of *LLL* in *SDNLL* can not be transposed into an embedding of *DLALL* in $SDNLL_\lambda$ because, in $SDNLL_\lambda$, we do not allow weakening on linear variables.

Because we are interested in expressivity in λ-calculus, it would not be very relevant to give examples of *SDNLL* proof-nets which are neither *LLL*, L^4 nor MS_{max} proof-nets. Proof-nets correspond to type derivations. However, the existence of *SDNLL* proof-nets which are not in *LLL*, does not imply that the λ-calculus type system corresponding to *SDNLL* type strictly more λ-terms than the λ-calculus type system corresponding to *LLL*. It is possible that a $SDNLL_\lambda$ type derivation D_1 of conclusion $\vdash t : A$ in $SDNLL_\lambda$ does not correspond to a *DLAL* type derivation, while there exists another derivation D_2 of conclusion $\vdash t : A$ which is in *DLAL*.

5.1.5 *SDNLL* as a type-system for λ-calculus

As noticed by Baillot and Terui [12], translating naively a subsystem of linear logic into a type-system for λ-calculus can result in a type-system which enjoys neither subject reduction nor the complexity bound enforced by the linear logic subsystem. The subsystem we define is heavily inspired by *DLAL*. For instance, the proof of subject reduction follows the proof of subject reduction of *DLAL* presented in [14].

We restrict the formulae considered by only allowing ! modalities on the left side of $\multimap$ connectives.

Definition 175. *For $s \in \mathbb{N}$, we define $\mathcal{F}_s^\lambda$ by the following grammar (with $t, d, n \in \mathbb{N}$, $t \geq s$ and X ranges over a countable set of variables). Notice that $\mathcal{F}_0^\lambda \supseteq \mathcal{F}_1^\lambda \supseteq \cdots$*

$$\mathcal{F}_s^\lambda := X_t \mid \mathcal{F}_s^\lambda \multimap \mathcal{F}_s^\lambda \mid !_{t,d,n}\mathcal{F}_{t+1}^\lambda \multimap \mathcal{F}_s^\lambda \mid \forall X_t.\mathcal{F}_s^\lambda$$

We define contexts[5] as sets of the shape $\{x_1 : A_1^{l_1}, \cdots, x_k : A_k^{l_k}\}$ where the x_is are pairwise distinct variables of λ-calculus, the A_is are formulae of $\mathcal{F}_0^\lambda$ and the l_is are elements of $\{\varnothing\} \cup \mathbb{N}^3$. Intuitively $A^{s,d,n}$ represents $!_{s,d,n}A$ while $A^{\varnothing}$ represents A. The sets of all contexts is written Con_λ, the set of contexts whose labels are all in $\mathbb{N}^3$ is written $Con_!$, the set of contexts whose labels are all equal to $\varnothing$ is written $Con_\S$.

In this paragraph, we consider $\Gamma = \{x_1 : A_1^{s_1,d_1,n_1}, \cdots, x_k : A_k^{s_k,d_k,n_k}\} \in Con_!$. Then we write $\Gamma^{\varnothing}$ for the context $\{x_1 : A_1^{\varnothing}, \cdots, x_k : A_k^{\varnothing}\}$. For $s, d, n \in \mathbb{Z}$, we write $\Gamma^{s,d,n}$ for $\{x_1 : A_1^{s_1+s,d_1+d,n_1+n}, \cdots, x_k : A_k^{s_k+s,d_k+d,n_k+d}\}$. We write $s(\Gamma)$ for the multiset of left indices, more formally $s(\Gamma) = \{x \mapsto |\{i \in \mathbb{N} \mid s_i = x\}|\}$. We define $d(\Gamma)$ and $n(\Gamma)$ similarly.

[5] Because we do not use context semantics in this subsection, there is no ambiguity

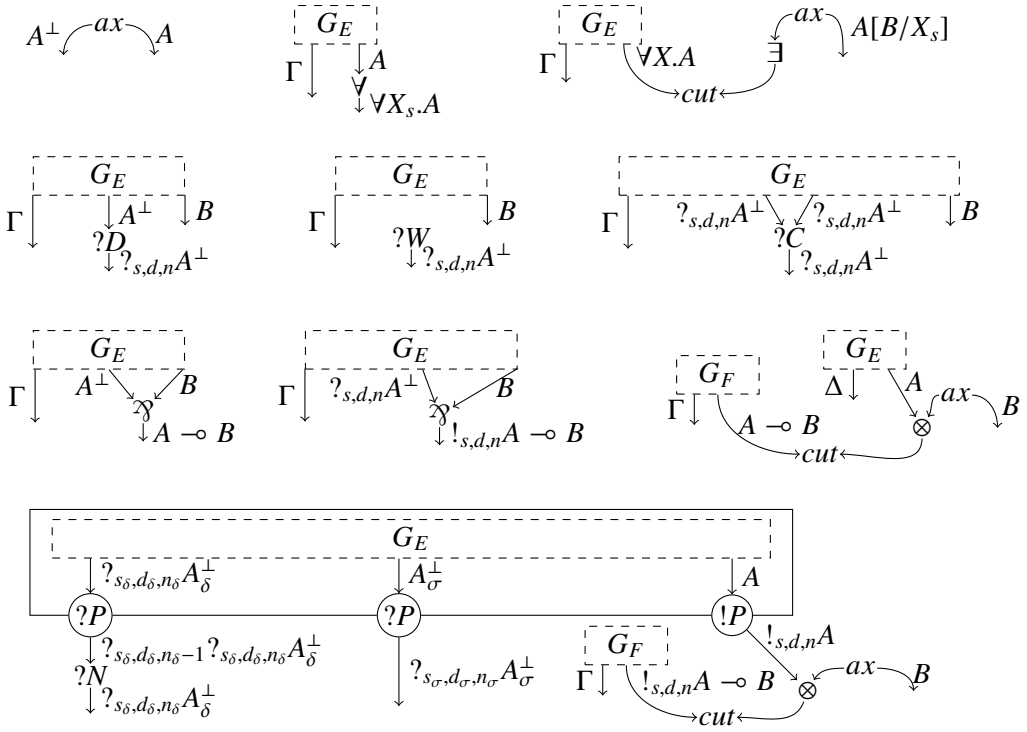


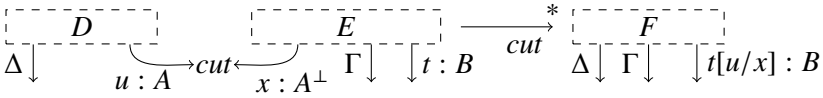
Figure 5.8: Derivations of $SNLL_\lambda$ can be translated into $SNLL$ proof-nets

If M is a multiset, then we write $M \geq x$ for "for every y such that $M(y) > 0$, we have $y \geq x$". Similarly, we write $M > y$ for "for every y such that $M(y) > 0$, we have $y > x$". Finally, we write $M \geqq y$ for "$M \geq x$ and $M(x) \leq 1$".

We present the type system $SDNLL_\lambda$ in Figure 5.7. In the type derivations, judgements are of the shape $\Gamma \vdash t : A$ with Γ a context. If $x : B^\varnothing$ is in Γ then x_i appears exactly once in t.

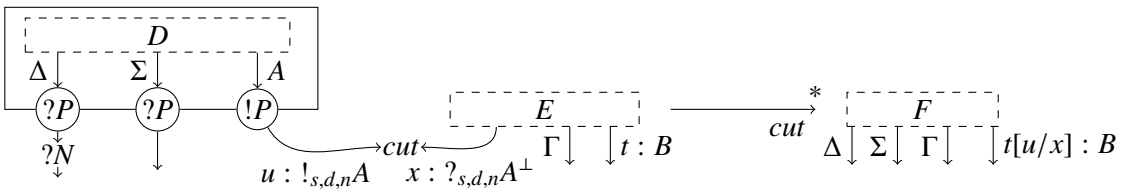
To prove subject reduction and the polynomial bound we define (in Figure 5.8) for every type derivation D of $A_1^\varnothing, \cdots, A_k^\varnothing, B_1^{s_1,d_1,n_1}, \cdots, B_l^{s_l,d_l,n_l} \vdash t : C$, we define a $SDNLL$ proof-net G_D with $k+l+1$ conclusions labelled with $A_1^\perp, \cdots, A_k^\perp, ?_{s_1,d_1,n_1}B_1^\perp, \cdots, ?_{s_l,d_l,n_l}B_l^\perp$ and C. In Figure 5.8, we suppose that the derivation D is obtained by applying a rule r (the rule used is at the same position in Figure 5.7) to the derivation E (if the last rule is binary, the derivation on the right is named F).

Lemma 176 (linear substitution)**.** *Let us consider derivations D and E of respective conclusions $\Delta \vdash u : A$ and $\Gamma, x : A^\varnothing \vdash t : B$. Then there exists a derivation F of conclusion $\Gamma, \Delta \vdash t[u/x] : B$ and*



Proof. Simple induction on E. Because the label of x is $\varnothing$, x is not the conclusion of a $?D$, $?C$ or $?P$ node. □

Lemma 177 (exponential substitution)**.** *Let us consider derivations D and E of respective conclusions $\Delta, \Sigma^\varnothing \vdash u : A$ and $\Gamma, x : A^{s,d,n} \vdash t : B$ with $d(\Delta \cup \Sigma) \geqq d$, $n(\Sigma) \geq n$ and $n(\Delta) > n$. Then there exists a derivation F of conclusion $\Gamma, \Delta, \Sigma \vdash t[u/x] : B$ and*



Proof. By induction on E. The most interesting step is the $\Rightarrow_e$ step. In this case, let us write C the box created in this step, and let us set $!_{s',d',n'\_} = \sigma(C)$. Either $d > d'$ so $d(\Delta \cup \Sigma) \geq d > d'$. Or $d = d'$, $d(\Delta \cup \Sigma) \geqq d \geq d'$ and $d(\Gamma) > d'$, thus $d(\Delta \cup \Sigma \cup \Gamma) \geqq d'$. □

Lemma 178. *Let us consider a derivation D of conclusion $\Delta \vdash \lambda x.t : A$ then $G_D \rightarrow^*_{cut} G_{D'}$ (considering the untyped proof-nets) with D' a derivation of conclusion $\Gamma \vdash \lambda x.t : A$ and the last rule R introduces the top connective of A (if $A = \forall A_1$ then $R = \forall_i$, if $A = !_{s,d,n}A_1 \multimap A_2$ then R is $\Rightarrow_i$, if $A = A_1 \multimap A_2$ with $A_1 \in \mathcal{F}_0^\lambda$ then $R = \multimap_i$)*

Proof. We prove it by induction on D. The last rule R can not be in $\{ax, \multimap_e, \Rightarrow_e\}$, because the λ-term of the conclusion would be of the shape $\lambda x.t$. If R in $\{\forall_i, \multimap_i, \Rightarrow_i\}$, then the lemma is trivial. If R is in $\{?D, ?W, ?C\}$, the derivation is of the shape:

$$\dfrac{\dfrac{E}{\Delta \vdash \lambda x.t : A}}{\Gamma \vdash \lambda x.t : A}\, R$$

By induction hypothesis, $G_E \rightarrow^*_{cut} G_{E'}$ with E' a derivation of conclusion $\Delta \vdash \lambda x.t : A$ and the last rule of E' introduces the top connective of A. We will examine the case where this last rule is $\multimap_i$, the two other cases are similar. Let us examine the following derivation D'' (let us notice that, because $G_E \rightarrow^*_{cut} G_{E'}$, we have $G_D \rightarrow^*_{cut} G_{D''}$).

$$\dfrac{\dfrac{\dfrac{F'}{\Delta, x : A_1 \vdash t : A_2}}{\Delta \vdash \lambda x.t : A_1 \multimap A_2}\multimap_i}{\Gamma \vdash \lambda x.t : A_1 \multimap A_2}\, R$$

In every case we can define D' as the following derivation (let us notice that $G_{D''} = G_{D'}$)

$$\dfrac{\dfrac{\dfrac{F'}{\Delta, x : A_1 \vdash t : A_2}}{\Gamma, x : A_1 \vdash t : A_2}\multimap_i}{\Gamma \vdash \lambda x.t : A_1 \multimap A_2}\, R$$

The last case to examine is $R = \forall_e$. In this case, the derivation is of the shape:

$$\dfrac{\dfrac{E}{\Gamma \vdash \lambda x.t : \forall X_s.A_1}}{\Gamma \vdash \lambda x.t : A_1[B/X_s]}\, R$$

By induction hypothesis, $G_E \rightarrow^*_{cut} G_{E'}$ with E' a derivation of conclusion $\Gamma \vdash \lambda x.t : \forall X_s.A$ and the last rule of E' is $\forall_i$. Let us examine the following derivation D'' (let us notice that, because $G_E \rightarrow^*_{cut} G_{E'}$, we have $G_D \rightarrow^*_{cut} G_{D''}$)

$$\dfrac{\dfrac{\dfrac{F'}{\Gamma \vdash \lambda x.t : A_1}}{\Gamma \vdash \lambda x.t : \forall X_s.A_1}\forall_i}{\Gamma \vdash \lambda x.t : A_1[B/X]}\forall_e$$

Then we can set D' as the proof-net obtained from F' by replacing X_s by B in the derivation. We can notice that $D'' \to^2_{cut} D'$ (a $\forall/\exists$ step and an axiom step).

$$\dfrac{\dfrac{\dfrac{F'}{\Delta, x : A_1 \vdash t : A_2}}{\Gamma, x : A_1 \vdash t : A_2} \multimap_i}{\Gamma \vdash \lambda x.t : A_1 \multimap A_2} R$$

□

Lemma 179 (subject reduction)**.** *If there exists a type derivation D whose conclusion is $\Gamma \vdash t : B$ and $t \to_\beta t'$ then there exists a type derivation D' whose conclusion is $\Gamma \vdash t' : B$ and $G_D \to^+_{cut} G_{D'}$.*

Proof. We prove the lemma by induction on D. Because there is a redex in t, t can not be a variable so the last rule is not an *ax* rule. Let us suppose that the last rule is a unary rule. Then D is of the shape:

$$\dfrac{\dfrac{E}{\Delta \vdash u : A}}{\Gamma \vdash t : B} R$$

In every case, u is a subterm of t containing the redex. So, by induction hypothesis, u reduces to a λ-term u'. By induction hypothesis, there exists a derivation E' of conclusion $\Delta \vdash u' : B$ and $G_E \to^k_{cut} G_{E'}$ (with $k \geq 1$). We can verify that in every case we can define D' as the following derivation.

$$\dfrac{\dfrac{E'}{\Delta \vdash u' : A}}{\Gamma \vdash t' : B} R$$

If the last rule is a $\multimap_e$ or $\Rightarrow_e$ step which does not correspond to the redex, the lemma is proved similarly.

If the last rule is a $\multimap_e$ rule corresponding to the redex then, by Lemma 178, $G_D \mapsto^* G_E$ with E a derivation of the following shape:

$$\dfrac{\dfrac{\dfrac{E_l}{\Gamma, \Delta x : A \vdash v : B}}{\Gamma, \Delta \vdash \lambda x.v : A \multimap B} \multimap_i \quad \dfrac{E_r}{\Delta \vdash u : A}}{\Gamma, \Delta \vdash (\lambda x.v)u : B}$$

By Lemma 176, G_E reduces to a derivation of conclusion $\Gamma, \Delta \vdash v[u/x] : B$. If the last rule is a $\Rightarrow_e$ rule corresponding to the redex then, the result follows similarly by Lemmas 178 and 177. □

Theorem 180. *If there exists a type derivation E whose conclusion is $\Gamma \vdash t : B$, x the size of the E, $S - 1$, $D - 1$ and $N - 1$ the maximum indexes in E, and ∂ be the depth of E (in terms of $\Rightarrow_e$ rules).*

$$t \to^k_\beta t' \qquad \Rightarrow \qquad k \leq x^{1+D^S \cdot \partial^{1+N \cdot S}}$$

Proof. Immediate from Theorem 173 and Lemma 179. □

We can notice that, contrary to *DLAL*, $SDNLL_\lambda$ does not allow weakening on linear variables. Thus one can never derive $\vdash \lambda x.t : A \multimap B$ when x is not a free variable of t. We are confident that adding the following rule to *SDNLL* does not break Lemma 180.

$$\dfrac{\Gamma \vdash t : B}{\Gamma, x : A^\varnothing \vdash t : B}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{g : (X_s \multimap X_s)^{\emptyset} \vdash g : X_s \multimap X_s} \quad \dfrac{\dfrac{}{h : (X_s \multimap X_s)^{\emptyset} \vdash h : X_s \multimap X_s} \quad \dfrac{}{a : (X_s)^{\emptyset} \vdash a : X_s}}{h : (X_s \multimap X_s)^{\emptyset}, a : (X_s)^{\emptyset} \vdash (h)a : X_s}}{g : (X_s \multimap X_s)^{\emptyset}, h : (X_s \multimap X_s)^{\emptyset}, a : (X_s)^{\emptyset} \vdash (g)(h)a : X_s}}{g : (X_s \multimap X_s)^{s-1,d,n}, h : (X_s \multimap X_s)^{\emptyset}, a : (X_s)^{\emptyset} \vdash (g)(h)a : X_s}}{g : (X_s \multimap X_s)^{s-1,d,n}, h : (X_s \multimap X_s)^{s-1,d,n}, a : (X_s)^{\emptyset} \vdash (g)(h)a : X_s}}{f : (X_s \multimap X_s)^{s-1,d,n}, a : (X_s)^{\emptyset} \vdash (f)(f)a : X_s}}{f : (X_s \multimap X_s)^{s-1,d,n} \vdash \lambda a.(f)(f)a : X_s \multimap X_s}}{\vdash \lambda f.\lambda a.(f)(f)a : !_{s-1,d,n}(X_s \multimap X_s) \multimap X_s \multimap X_s}}{\vdash \lambda f.\lambda a.(f)(f)a : \forall X_s, !_{s-1,d,n}(X_s \multimap X_s) \multimap X_s \multimap X_s}$$

Figure 5.9: Type derivation of $2 : \forall X_s, !_{s-1,d,n}(X_s \multimap X_s) \multimap X_s \multimap X_s$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{m : N \vdash m : N}}{m : N \vdash m : !_{s,d,n}F \multimap F} \quad \dfrac{}{g : F \vdash g : F}}{m : N, g : F^{s,d,n} \vdash (m)g : F} \quad \dfrac{\dfrac{\dfrac{\dfrac{}{n : N \vdash n : N}}{n : N \vdash n : !_{s,d,n}F \multimap F} \quad \dfrac{}{h : F \vdash h : F}}{n : N, h : F^{s,d,n} \vdash (n)h : F} \quad \dfrac{}{x : X_s^{\emptyset} \vdash x : X_s}}{n : N, h : F^{s,d,n}, x : X_s^{\emptyset} \vdash ((n)h)x : X_s}}{m : N, n : N, g : F^{s,d,n}, h : F^{s,d,n}, x : X_s^{\emptyset} \vdash ((m)g)((n)h)x : X_s}}{m : N, n : N, f : F^{s,d,n}, x : X_s^{\emptyset} \vdash ((m)f)((n)f)x : X_s}}{m : N, n : N, f : F^{s,d,n} \vdash \lambda x.((m)f)((n)f)x : X_s \multimap X_s}}{m : N, n : N \vdash \lambda f.\lambda x.((m)f)((n)f)x : \underline{N}_{s,d,n}}}{m : N \vdash \lambda n.\lambda f.\lambda x.((m)f)((n)f)x : \underline{N}_{s,d,n} \multimap \underline{N}_{s,d,n}}}{\vdash \lambda m.\lambda n.\lambda f.\lambda x.((m)f)((n)f)x : \underline{N}_{s,d,n} \multimap \underline{N}_{s,d,n} \multimap \underline{N}_{s,d,n}}$$

Figure 5.10: Type derivation of $add : \underline{N}_{s,d,n} \multimap \underline{N}_{s,d,n} \multimap \underline{N}_{s,d,n}$. To simlify the proof derivation, we write F for $X_s \multimap X_s$ and N for $\underline{N}^{\emptyset}_{s,d,n}$.

However, one can not extend the encoding of Figure 5.8 to this rule because Linear Logic does not allow weakening on a formula A unless A is of the shape $?A'$. Thus we would have to prove the bound directly on λ-calculus (or a similar language as in [12]). Which makes the proof more difficult, because we can not use the lemmas we proved on context semantics. If we had defined the context semantics and the criteria on a more general framework (for example interaction nets, for which we define a context semantics in Chapter 6) we would not have problems to accomodate such a simple modification.

To give an intuition on the system, let us give some examples of proof derivations. For any $s \geq 1$, and $k \in \mathbb{N}$, $\underline{k}$ can be typed with the following type (see Figure 5.9 for the type derivation of $\underline{2}$):

$$\mathbf{N}_{s,d,n} = \forall X_s, !_{s-1,d,n}(X_s \multimap X_s) \multimap X_s \multimap X_s$$

Then, the addition can be typed as shown in Figure 5.10. Finally, although this type system has no built-in mechanism to type tuples, we can encode them by the usual church encoding (Figure 5.11). Let us notice that this encoding does not require any additional constraint on the types, contrary to L^4 (where the terms must have the same level).

We isolate four constraints that each light logic has and which *SDNLL* does not have. We illustrate each constraint with an intuitive description and a λ-term which can be typed in *SDNLL* but seemingly not in any other light logic because of this constraint. We set $S = \lambda m.\lambda f.\lambda x.((m)f)(f)x$ implementing succesor,

.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{f : (A \multimap B \multimap X)^{\varnothing} \vdash f : A \multimap B \multimap X} \qquad \Gamma \vdash t : A}{\Gamma, f : (A \multimap B \multimap X)^{\varnothing} \vdash (f)t : B \multimap X} \qquad \Delta \vdash u : B}{\Gamma, \Delta, f : (A \multimap B \multimap X)^{\varnothing} \vdash ((f)t)u : X}}{\Gamma, \Delta \vdash \lambda f.((f)t)u : (A \multimap B \multimap X) \multimap X}}{\Gamma, \Delta \vdash \lambda f.((f)t)u : \forall X.(A \multimap B \multimap X) \multimap X}}{\Gamma, \Delta \vdash \langle t, u \rangle : \langle A, B \rangle}$$

Figure 5.11: Simple encoding of pairs.

and $+ = \lambda m.\lambda n.\lambda f.\lambda x.((m)f)((n)f)x$ implementing addition on Church integers and $x + y + z$ is a notation for $((+)((+)x)y)z$.

- In light logics, in $\langle t, u \rangle$, t and u must have the same stratum indices (depth in *LLL* and *MS*, level in L^4). The term $(\underline{k})\lambda\langle x, y, z\rangle.\langle x, ((x)S)\underline{0}, x + y + z\rangle$ is not typable in light logics: because the function $\lambda\langle x, y, z\rangle.\langle x, ((x)S)\underline{0}, x + y + z\rangle$ is iterated, we have $s(y) = s\left(((x)S)\underline{0}\right)$ so $s(y) > s(x)$. But, because they are in the same tuple, it must be $s(y) = s(x)$. The $x + y + z$ term ensures that the stratum indices of y and x can not be modified by § modalities.

- There is no N rule in light logics and in their encodings in *SDNLL*. This seems to prevent the typing of $\underline{k}(\lambda\langle v, w, x, y, z\rangle\langle w, w, x, w + x + y, ((x)(+)v)0\rangle)$.

- Contrary to *LLL* and L^4, one can have several variables in the context during a $\Rightarrow_e$ rule. So, $t = \underline{k}(\lambda\langle x, y, z\rangle.\langle x, x + y, y\rangle)$ is typable in *SDNLL* but not in *LLL* and L^4. Moreover, the maximum nest of terms is not a priori bounded by the type system, so if we set $u = \lambda\langle x, y, z\rangle.\langle z, z, z\rangle$, then $(t)(u)(t)\cdots(u)t$ is typable in *SDNLL* whatever the length of the chain of applications, whereas in *MS* the maximum length of such a chain is bounded.

- In light logics, there is no subtyping. For example, in L^4, a A^i formula can not be considered as a A^{i-1} formula. The example in the first item of this list would be typable in L^4 if it was allowed to decrease the level of a formula by mean of a subtyping relation.

5.2 Quantifier Predicative Linear Logic

In this section we define a type system, named Quantifier Predicative Linear Logic (*QPLL*), entailing a primitive recursive bound on *cut*-elimination. This type system is inspired by the constraints of *SDNLL* on its leftmost index. The constraints are relaxed to allow $(B,P)\ddot{\rightsquigarrow}(B,Q)$ when $P \neq Q$. The idea is that, if $((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(B)},P),U.?_u)$, by Lemma 166 we have

$$\boldsymbol{\beta}((\sigma(B),P),[!_t]) = \boldsymbol{\beta}((\overline{\sigma(B)},P),U.?_u)$$

Thus, U must be of the shape $U_1@[\exists]@U_2$ trace element (otherwise $\boldsymbol{\beta}((\overline{\sigma(B)},P),U.?_u)$ would be a subformula of $\overline{\boldsymbol{\beta}((\sigma(B),P),[!_t])}$). So there exists a $\exists$ node l such that there exist paths of the shape:

$$((\sigma(B),P),[!_t]) \rightsquigarrow^* ((concl_l,Q),U_1.\exists)$$
$$((concl_l,Q),[\exists]) \rightsquigarrow^* ((\overline{\sigma(B)},P),[\exists]@U_2.?_u)$$

Let us set $!_sH[\forall X_{t,q}._] = \beta(\sigma(B))$ with H the hole-formula corresponding to U_2: it is to say that $\beta(\sigma(B))_{|U_2^{\perp}.!_u} = \overline{\beta(\overline{\sigma(B)})}_{|U_2.?_u} = \forall X_{t,q}$. Because the $\forall X_{t,q}._$ is under the $!_s$ we have $s \leq t$. Then, because $((\sigma(B),P),[!_t]) \rightsquigarrow^* ((concl_l,Q),U_1.\exists)$ the formula associated with l contains a label equal to s so $t \leq s$. So we know that $s = t$.

Finally, the path $\boldsymbol{\beta}((\sigma(B),P),[!_t]) = \boldsymbol{\beta}((\overline{\sigma(B)},P),U.?_u)$ must cross a $?C$ node n or $?N$ node upwards with $[!\_]$ on its trace (Lemma 182). Thus $\beta(concl_n)$ is of the shape $?_sH[\exists X_{s,q}._]$. According to the rules of *QPLL*, the premises of n are labelled with formulae of the shape $?_sH[\exists X_{s,\perp}._]$. Thus, $\beta(concl_l)$ must be of the shape $\exists X_{s,\perp}._$. The formula associated with such a context can not contain labels equal to s, this is a contradiction. So our hypothesis was false, $\ddot{\rightsquigarrow}$ is acyclic.

The formulae of *QPLL* are defined by the following grammar, with X ranging over a countable set of variable, s,t ranging over $\mathbb{N}$ with $t \geq s$ and $q \in \{\perp,\top\}$.

$$\mathcal{F}^s_{QP} = X_t \mid X_t^{\perp} \mid \mathcal{F}^s_{QP} \parr \mathcal{F}^s_{QP} \mid \mathcal{F}^s_{QP} \otimes \mathcal{F}^s_{QP} \mid \forall X_{t,q}.\mathcal{F}^s_{QP} \mid \exists X_{t,q}.\mathcal{F}^s_{QP} \mid !_t\mathcal{F}^t_{QP} \mid ?_t\mathcal{F}^t_{QP}$$

In the introduction of this section, we wrote that for every $?C$ or $?N$ node n such that $\beta(concl_n)$ is of the shape $?_sH[\exists X_{s,q}._]$, the premises of n are labelled with formulae of the shape $?_sH[\exists X_{s,\perp}._]$. To implement this condition, for $s \in \mathbb{N}$, we define a mapping $(\_)_{/s}$ as follows: $(\exists X_{s,q}.A)_{/s} = \exists X_{s,\perp}.A_{/s}$ and in the other cases $(\_)_{/s}$ has no effect on the constructor: $(X_t)_{/s} = X_t$, $(X_t^{\perp})_{/s} = X_t^{\perp}$, $(A \parr B)_{/s} = A_{/s} \parr B_{/s}$, $(A \otimes B)_{/s} = A_{/s} \otimes B_{/s}$, $(\forall X_{t,q}.A) = \forall X_{t,q}.(A_{/s})$, $(!_tA)_{/s} = !_tA_{/s}$, $(?_tA)_{/s} = ?_tA_{/s}$ and (if $t > s$) $(\exists X_{t,q}.A)_{/s} = \exists X_{t,q}.A_{/s}$.

We present the system as a proof derivation system because it is easier to present side conditions (for the $\exists$ rules) this way.

Definition 181. *The proofs of QPLL are defined inductively by the rules of Figure 5.12. A proof derivation is considered valid if and only if every formula is in* $\mathcal{F}^0_{QP}$.

We only prove that this system is sound for primitive recursive functions, we do not know yet whether the system is complete. If *QPLL* is not powerful enough, there would be various ways to increase its expressivity. For instance, because the proof of soundness of this system is rather technical, we did not define a subtyping relation. Defining a subtyping relation on indices as in *SDNLL* (Section 5.1.1) and *SwLL* (Section 5.3) could increase the expressivity of the system.

Lemma 182. *Let G be a LL proof-net, let us suppose that* $((\sigma(B),P),[!_t]) \mapsto^* ((\overline{\sigma(B)},P),[!\_]@U.?_u)$ *with* (B,P) *a anonical box and t a copy of* (B,P)*, then the path contains a context* $((\overline{e},Q),[!_v])$ *with e the conclusion of a* $?N$ *or* $?C$ *node.*

$$\frac{A \in \mathcal{F}^0_{QP}}{\vdash A, A^\perp} \qquad \frac{\vdash \Gamma, A \qquad \vdash \Delta, A^\perp}{\vdash \Gamma, \Delta} \qquad \frac{\vdash \Gamma, A \qquad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B} \qquad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}$$

$$\frac{\vdash \Gamma, A \qquad X_s \notin FV(\Gamma)}{\vdash \Gamma, \forall X_{s,q}.A} \qquad \frac{\vdash \Gamma, A[B/X_s] \qquad B \in \mathcal{F}^s_{QP}}{\vdash \Gamma, \exists X_{s,\top}.A} \qquad \frac{\vdash \Gamma, A[B/X_s] \qquad B \in \mathcal{F}^{s+1}_{QP}}{\vdash \Gamma, \exists X_{s,\perp}.A}$$

$$\frac{\vdash \Gamma, ?_s A_{/s}, ?_s A_{/s}}{\vdash \Gamma, ?_s A} \qquad \frac{\vdash \Gamma, ?_s ?_s A_{/s}}{\vdash \Gamma, ?_s A} \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, ?_s A} \qquad \frac{\vdash \Gamma}{\vdash \Gamma, ?_s A} \qquad \frac{\vdash A_1, \cdots, A_n, A}{\vdash ?_{s_1} A_1, \cdots, ?_{s_n} A_n, !_s A}$$

Figure 5.12: Definition of *QPLL*, s ranges over $\mathbb{N}$ and q ranges over $\{\perp, \top\}$

Proof. If there is no such node, for every $x \in Sig$, we have $((\sigma(B), P), [!_x]) \mapsto^* ((\overline{\sigma(B)}, P), [!_x]@U.?_u)$. In particular, we can notice that $((\sigma(B), P), [!_u]) \mapsto^* ((\overline{\sigma(B)}, P), [!_u]@U.?_u)$. Thus, we have $(B, P, u) \rightsquigarrow\!\!\!\!\!\!\cdot\;\; (B, P, u)$ which contradicts Corollary 123. □

Lemma 183. *Let θ and θ' be substitutions on variables. If $(\beta(A[\theta], e, P, T, U)[\theta'])_{|V}$ is defined, then either $A_{|V}$ is defined or there exists a suffix V' of V such that $A_{|V'}$ is a variable.*

Proof. We can prove it by induction on $\min(|V|,$ depth of $A)$.

- If $V = []$, then $A_{|V} = A_{|[]} = A$ is defined.
- If $A = X$ or $A = X^\perp$, then we set $V' = []$ and can verify that $A_{|V'} = A_{|[]} = A$ is a variable.
- Else, the cases are similar so we will only show the cases where V is of the shape $W.\exists$. In this case, we know that $(\beta(A[\theta], e, P, T, U)[\theta'])_{|W.\exists}$ is defined so $\beta(A[\theta], e, P, T, U)[\theta']$ is of the shape $\exists X.A'$. Because we know that A is not a variable, A is of the shape $\exists X.A'$. So, by Definition 162, there exists a subsitution θ'' such that:

  $$(\beta(A[\theta], e, P, T, U)[\theta'])_{|W.\exists} = (\exists X.\beta(A'[\theta], e, P, T, U)[\theta''][\theta'])_{|W.\exists}$$
  $$(\beta(A[\theta], e, P, T, U)[\theta'])_{|W.\exists} = (\beta(A'[\theta], e, P, T, U)[\theta''][\theta'])_{|W}$$

  By induction hypothesis, either $A'_{|W}$ is defined (and in this case $A_{|V} = (\exists X.A')_{|W.\exists} = A'_{|W}$ is defined) or there exists a suffix W' of W such that $A'_{|W'}$ is a variable (and in this case $A_{|W'.\exists} = (\exists X.A')_{|W'.\exists} = A'_{|W'}$ is a variable).

□

Lemma 184. *Let us suppose that $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(B)}, P), U.?_u)$. Then there is a finite suffix V of $\cdots @\overline{U}@U@\overline{U}@U@\overline{U}$ such that $\sigma(B)_{|V.!_t}$ is a variable.*

Proof. For $i \in \mathbb{N}$, we define V_i by induction on i by $V_0 = []$ and $V_{i+1} = \overline{V_i@U}$. Then we will prove by induction on i that either $\beta(\sigma(B))_{|V_i.!_t}$ is defined or there exists a suffix V of V_i such that $\beta(\sigma(B))_{|V.!_t}$ is a variable. Then, because the depth of $\beta(\sigma(B))$ is finite, there exists some i and a suffix V of V_i such that $\beta(\sigma(B))_{|V.!_t}$ is a variable.

The formula labelling $\sigma(B)$ is of the shape $!A$ so $\beta(\sigma(B))_{|V_0.!_t} = A, \beta(\sigma(B))_{|V_0.!_t}$ is defined.

Let us suppose that either $\beta(\sigma(B))_{|V_i.!_t}$ is defined or there exists a suffix V of V_i such that $\beta(\sigma(B))_{|V.!_t}$ is a variable. If we are in the second case, then we can notice that V is also a prefix of V_{i+1}, thus proving the property for $i + 1$. Else, let us notice that $((\sigma(B), P), V_i.!_t) \rightsquigarrow^* ((\overline{\sigma(B)}, P), V_i@U.?_u)$. We

know that $\beta(\sigma(B))_{|V_i.!_t}$ is defined so $\beta(\beta(\sigma(B))[\theta(\sigma(B),P)],\sigma(B),P,[!_t],[])_{|V_i.!_t}$ is defined. By Lemma 164, $\beta(\overline{\beta(\sigma(B))[\theta(\sigma(B),P)]},\overline{\sigma(B)},P,U.?_u,[])_{|U.?_u}$ is defined and we have the following equalities:

$$\beta(\beta(\sigma(B))[\theta(\sigma(B),P)],\sigma(B),P,[!_t],[])_{|[!_t]} = \beta(\overline{\beta(\sigma(B))[\theta(\sigma(B),P)]},\overline{\sigma(B)},P,U.?_u,[])_{|U.?_u}$$

$$\beta(\beta(\sigma(B))[\theta(\sigma(B),P)],\sigma(B),P,[!_t],[])_{|V_i.!_t} = \beta(\overline{\beta(\sigma(B))[\theta(\sigma(B),P)]},\overline{\sigma(B)},P,U.?_u,[])_{|V_i@U.?_u}$$

$$\beta(\beta(\sigma(B))[\theta(\sigma(B),P)],\sigma(B),P,[!_t],[])_{|V_i.!_t} = \overline{\beta(\beta(\sigma(B))[\theta(\sigma(B),P)],\sigma(B),P,\overline{U}.!_u,[])_{|\overline{V_i@U}.!_u}}$$

$$\beta(\beta(\sigma(B))[\theta(\sigma(B),P)],\sigma(B),P,[!_t],[])_{|V_i.!_t} = \overline{\beta(\beta(\sigma(B))[\theta(\sigma(B),P)],\sigma(B),P,\overline{U}.!_u,[])_{|V_{i+1}.!_t}}$$

So $\beta(\beta(\sigma(B))[\theta(\sigma(B),P)],\sigma(B),P,\overline{U}.!_u,[])_{|V_{i+1}.!_t}$ is defined. By Lemma 183, either $\beta(\sigma(B))_{|V_{i+1}.!_t}$ is defined or there exists a suffix V of V_{i+1} such that $\beta(\sigma(B))_{|V.!_t}$ is a variable. □

Lemma 185. *If* $((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(B)},Q),U.?_u)$ *then* $P \neq Q$.

Proof. We prove the lemma by contradiction. Let us suppose that there exists $(B,P) \in Pot(B_G)$, $U \in Tra$ and $t,u \in Sig$ and $((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(B)},P),U.?_u)$. We consider the shortest such path.

Let us write $!_sA = \beta(\sigma(B))$. By Lemma 164, we know that

$$\beta(\beta(\sigma(B))[\theta(\sigma(B),P)],\sigma(B),P,[!_t],[])_{|[!_t]} = \beta(\overline{\beta(\sigma(B))[\theta(\sigma(B),P)]},\overline{\sigma(B)},P,U.?_u,[])_{|U.?_u}$$

So, if U does not contain $\exists$ or $\forall$ trace elements, $A[\theta(\sigma(B),P)] = \overline{A[\theta(\sigma(B),P)]}_{|U}$ which is contradiction (it would mean that $A[\theta(\sigma(B),P)]$ is strictly deeper than itself). So U contains a $\exists$ or a $\forall$ trace element. So, $U = R_1@R_2$ with either $R_2 = [\exists]@R_2'$ or $R_2 = [\forall]@R_2'$.

By Lemma 184, there exists a suffix V of $\cdots@\overline{U}@U@\overline{U}$ such that $\beta(\sigma(B))_{|V.!_t}$ is a variable. Thus $\beta(\beta(\sigma(B))[\theta(\sigma(B),P)],\sigma(B),P,[!_t],[])_{|V.!_t} = \overline{\beta(\beta(\sigma(B))[\theta(\sigma(B),P)],\sigma(B),P,\overline{U}.!_u,[])_{|\overline{V@U}.!_u}}$. Let us notice that there exist S_1,S_2 such that $\overline{V}@U = S_1@S_2@V$ and (either $S_2 = [\forall]@\_$ or $S_2 = [\exists]@\_$). Let us consider the smallest suffix U' of U such that $\beta(\beta(\sigma(B))[\theta(\sigma(B),P)],\sigma(B),P,\overline{U'}.!_u,[])_{|S_2@V.!_u}$ is defined.

So there exists lists of trace elements U_1,U_2 verifying the following property $\mathcal{P}(U_1,U_2)$:

- Either $U = U_1@[\forall]@U_2$ and there exists $(l,Q) \in Pot(\mathrm{N}_G^{\forall})$, and trace elements lists U_1',U_2',W_1,W_2,W_2' such that $((concl_l,Q),[\forall]) \rightsquigarrow^* ((\overline{\sigma(B)},P),[\forall]@U_2.?_u)$, $U_1 = U_1'@W_1@W_2@U_2'$, $\beta(concl_l)_{|U_2'}$ is the variable corresponding to l, $\phi_G^{lim}(l,Q)_{|W_1@W_2}$ is defined and (either $W_2 = [\forall]@W_2'$ or $W_2 = [\exists]@W_2'$).
- Or $U = U_1@[\exists]@U_2$ and there exists $(l,Q) \in Pot(\mathrm{N}_G^{\exists})$ and trace elements lists U_1',U_2',W_1,W_2,W_2' such that $((concl_l,Q),[\exists]) \rightsquigarrow^* ((\overline{\sigma(B)},P),[\exists]@U_2.?_u)$, $U_1 = U_1'@W_1@W_2@U_2'$, $\beta(concl_l)_{|U_2'}$ is the variable corresponding to l, $\phi_G^{lim}(l,Q)_{|W_1@W_2}$ is defined and (either $W_2 = [\forall]@W_2'$ or $W_2 = [\exists]@W_2'$).

We consider the couple (U_1,U_2) with the shortest U_2 possible.

- Either $U = U_1@[\forall]@U_2$ and there exists $(l,Q) \in Pot(\mathrm{N}_G^{\forall})$ and trace elements lists U_1',U_2',W_1,W_2,W_2' such that $((concl_l,Q),[\forall]) \rightsquigarrow^* ((\overline{\sigma(B)},P),[\forall]@U_2.?_u)$, $U_1 = U_1'@W_1@W_2@U_2'$ such that $\beta(concl_l)_{|U_2'}$ is the variable corresponding to l, $\phi_G^{lim}(l,Q)_{|W_1@W_2}$ and (either $W_2 = [\forall]@W_2'$ or $W_2 = [\exists]@W_2'$). We will prove that this case would lead to a contradiction. Indeed, let us reduce every *cut* except those which have $(\sigma(B),P)$ (or its reduct) as one of its premises. We would still have a path of the shape $((\sigma(B),[\mathtt{e};\cdots;\mathtt{e}]),[!_t]) \rightsquigarrow^* ((concl_l,Q),U_1@[\forall]) \rightsquigarrow^* ((\overline{\sigma(B)},P),U_1@[\forall]@U_2)$ and the only *cut* has $\sigma(B)$ as one of its premises, there is a suffix U_2' of U_1 such that $\beta(concl_l)_{|U_2'}$ is the variable Z_l corresponding to l and (either $\phi_G^{lim}(l,Q) = \_@[\exists]@\_$ or $\phi_G^{lim}(l,Q) = \_@[\forall]@\_$). Thus

the path has the following shape: $((\sigma(B), P), [!_t]) \rightsquigarrow ((\overline{e_1}, \_), [!\_]) \rightsquigarrow \cdots \rightsquigarrow ((\overline{e_{i-1}}, \_), [!\_]) \rightsquigarrow ((e_i, \_), [!\_]) \rightsquigarrow ((e_{i+1}, \_), \_) \cdots ((e_j, \_), \_) \rightsquigarrow ((\overline{\sigma(B)}, P), U.?_u)$ (with $e_1, \cdots, e_j$ downwards arrows). Z_l can not be free in $\sigma(B)$. Let us consider the last context C preceding $((concl_l, Q), U_1@[\forall])$ in the path whose formula does not contain any free occurrence of Z_l. Either $C = ((\overline{concl_l}, Q), W@[\exists])$ or $C = ((\overline{concl_m}, Q')W@[\forall])$ (with m a $\exists$ node whose associated formula contains Z_l). In both cases, we have a contradiction because the edge is upward and different from $\overline{\sigma(B)}$, the trace is not limited to a $!\_$ trace element.

- Or $U = U_1@[\exists]@U_2$ and there exists $(l, Q) \in Pot(\mathrm{N}_G^{\exists})$ and trace elements lists $U_1', U_2', W_1, W_2, W_2'$ such that $((concl_l, Q), [\exists]) \rightsquigarrow^* ((\overline{\sigma(B)}, P), [\exists]@U_2.?_u)$, $U_1 = U_1'@W_1@W_2@U_2'$, $\beta(concl_l)_{|U_2'}$ is the variable corresponding to l, and $\phi_G^{lim}(l, Q) = W_1@W_2$ (and either $W_2 = [\exists]@W_2'$ and $W_2 = [\forall]@W_2'$). We will prove that it leads to a contradiction. By Lemma 163, we know that the context $\beta(\beta(\overline{\sigma(B)}), \overline{\sigma(B)}, P, [\exists]@U_2.?_u, [])_{|[\exists]@U_2.?_u}$ is well-defined. So either, $\overline{\beta(\sigma(B))}_{|[\exists]@U_2.?_u}$ is defined, or $U_2 = V_1@[\exists]@V_2$ or $U_2 = V_1@[\forall]@V_2$ and $\mathcal{P}(U_1@[\exists]@V_1, V_2)$ which contradicts our hypothesis that (U_1, U_2) is the pair satisfying $\mathcal{P}$ with the shortest right element.

  So $\overline{\beta(\sigma(B))}_{|[\exists]@U_2.?_u}$ is defined, $\beta(\sigma(B))_{|[\forall]@\overline{U_2}.!_t}$ is defined. Thus $\beta(\sigma(B)) = !_sH[\forall_{t,q}.A']$ with $s \leq t$.

  There exists edges e, f, g such that e, f are premises of a $?C$ (or $?N$) node whose conclusion is g and:

$$((\sigma(B), P), [\forall]@\overline{U_2}@[!_u]) \rightsquigarrow^* ((\overline{g}, \_), [\forall]@\overline{U_2}@[!\_]) \rightsquigarrow ((\overline{e}, \_), [\forall]@\overline{U_2}@[!\_]) \rightsquigarrow^* ((\overline{concl_l}, Q), [\forall])$$
$$((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{g}, \_), [!\_]) \rightsquigarrow ((\overline{f}, \_), [!\_]) \rightsquigarrow^* ((concl_l, Q), \_.\exists)$$

  Let us notice that the formula associated to l contains a label equal to s, so $t \geq s$ and $t = s$. By Lemma 163, $\beta(\overline{g})$ is of the shape $!_sH[\forall X_{s,q'}.A']$ so $\beta(e) = ?_sH^{\perp}[\exists X_{s,\perp}.\_]$. Thus $\beta(concl_l) = \exists X_{s,\perp}\_$. The formula associated to l can not contain a label equal to s.

□

Lemma 186. *Let (B, P, t) and (B, P, u) in MaxCanCop$_G$, then* $\neg\left((B, P, t) \ddot{\rightsquigarrow}^+ (B, P, u)\right)$

Proof. We prove the lemma by contradiction. Let us suppose that $(B_0, P_0, t_0) \ddot{\rightsquigarrow} (B_2, P_2, t_2) \cdots (B_k, P_k, t_k)$ and $(B_0, P_0) = (B_k, P_k)$. By Lemma 122, we know that either $((\sigma(B_0), P_0), [!_{t_0}]) \rightsquigarrow^* ((\sigma(B_0), P_0), [!\_]@T.!_{t_k})$ or $((\sigma(B_0), P_0), [!_{t_0}]) \rightsquigarrow^* ((\overline{\sigma(B_0)}, P_0), T.?_{t_k})$. The first alternative is ruled out because of Lemma 90. So there exists a path of the shape $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(B)}, P), U.?_u)$. This is in contradiction with Lemma 185. □

Lemma 187. *QPLL is sound for primitive recursion.*

Proof. Immediate from Theorem 152. □

$$B\ \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdash a_l : X^\perp, a_r : X \qquad \vdash b_l : X^\perp, b_r : X}{\vdash a_l : X^\perp, b_l : X^\perp, t : X \otimes X}}{\vdash \sigma_2(B) : ?_1X^\perp, \sigma_1(B) : ?_0X^\perp, \sigma(B) : !_0(X \otimes X)}}{\vdash \sigma_2(B) : ?_1X^\perp, \sigma_1(B) : ?_1X^\perp, \sigma(B) : !_0(X \otimes X)}}{\vdash c : ?_1X^\perp, \sigma(B) : !_0(X \otimes X)}}{\vdash p : (?_1X^\perp) \parr !_0(X \otimes X)}$$

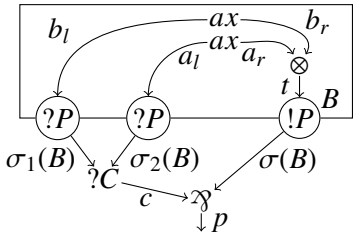


Figure 5.13: Correspondence between proof derivation and proof-net.

5.3 Sweetened Linear Logic

The idea of this section is to take advantage of the more general criteria defined in Section 3.5 to define a more expressive decidable subsystem of Linear Logic. The type system is called "*Sw*eetened" *L*inear *L*ogic (*SwLL*) by analogy with the food industry. Linear Logic as a whole can lead to very high time complexity, in the same way as eating food without restriction leads to obesity. To prevent this, one may enfore a strict discipline (only considering the proof-nets of *LLL* or sugar-free food). With these restrictions, programming becomes tedious and meals might become boring. To get some expressivity (sugar taste) back, one may relax the restrictions (allow sweeteners). However, as in the food industry, there is a new flaw. Sweeteners are allegedly associated with cancer. Here, our cancer is the complexity of the system. The number of labels carried by ! and ? connectives in the formulae of *SwLL* is very high and the rules of *SwLL* are quite complex. This is the reason why we will present *SwLL* as an intersection of several systems. Each system enforces some property on the labels. The conjunction of those properties entails the existence of a weak stratification (Definition 108) and a bound on $|\dot{\prec}|$, implying a polynomial bound (Theorem 119).

In the definition of weak stratifications, a subset S of B_G is distinguished. For several of the subsystems defined below, we suppose given S (this does not break decidability because B_G is a finite set and one may try to infer types for every $S \subseteq B_G$). We also suppose given a subset $S_{\boxdot\to}$ of B_G. This set will have the following meaning: if $(B, P, t) \boxdot\to (C, Q, u)$ then $C \in S_{\boxdot\to}$.

We present every system as proof derivation systems as this presentation is much more compact. Let us notice that boxes of proof-nets correspond to promotion rules of proof derivation. Thus, the sets S and $S_{\boxdot\to}$ mentioned in the previous paragraph will be sets of instances of promotion rules. The instances will be identified by their names, written on their left.

We also suppose defined sets $E_\circ$, and $E_{\boxdot\to}$ of "edges". In proof derivation systems, edges correspond to a set of formulae in judgements. The correspondence is illustrated by Figure 5.13. It represents a proof derivation in $SwLL_{dc}$ and the corresponding proof-net. Let us observe that the edge a_l correspond to two occurrences of the formula $X^\perp$. Similarly, $\sigma_1(B)$ corresponds to occurrences of $?_0X^\perp$ and $?_1X^\perp$. In the general case, the formulae corresponding to an edge can differ only on their labels (the underlying *LL* formula is the same). In fact, for every proof derivation system $SwLL_z$, we will define a subtyping relation $\leq_z$ such that, if the edge e corresponds to occurrences $e_1, ..., e_k$ of formulae (from top to bottom), then $e_1 \leq_z e_2 \leq_z \cdots \leq_z e_k$.

5.3.1 Definition of $SwLL_{dc}$

The role of the first subsystem is to enforce the conditions on $\mathfrak{d}(\_)$ required by the second part of Definition 108. In $SwLL_{dc}$, the ! and ? connectives carry a label which plays a role quite similar to the second label (usually denoted d) of *SDNLL*. There are three differences:

- For every box B, if $((\sigma(B), P), [!_t]) \mapsto^* ((\sigma(C), \_), [!\_])$ implies that every box containing C is in S, then B does not need to abide to the same rules. This allows to have $B \not\succ B$ when $|Can(B)|$ does not depend on the argument. A formula e is labelled by an element of $\{0, 1\}$ which is equal to 1 only if every box containing e is in S.

- Whenever $(B, P, t) \smile^* (C, Q, u)$ and $(B, P, t') \smile^* \boxdot\!\!\mapsto^+ (C, Q', u')$ we must have $\mathfrak{d}(B) > \mathfrak{d}(C)$. We will enforce a stronger property: for every box $C \in S_{\boxdot\mapsto}$, we require the labels corresponding to *every* auxiliary door (even the leftmost one) of C to be strictly greater than the label corresponding to its principal door. Thus, if $(B, P, t) \smile^* (C, Q, u)$ and $(B', P, t') \smile^* \boxdot\!\!\mapsto^+ (C, Q', u')$ we have $\mathfrak{d}(B') > \mathfrak{d}(C)$.

- Finally, we fuse two previously defined way to enforce dependence control. Let us suppose that $B \not\succ B$ because of paths of the following shape (with e the conclusion of a $?C$ node and $i \neq j$)

$$((\sigma(B), P), [!_t]) \mapsto^* ((\overline{e}, Q), [!_{1_{(\_)}}]) \mapsto ((\overline{e_l}, Q), [!\_]) \mapsto^* ((\overline{\sigma_i(B)}, Q), [!\_]) \hookrightarrow ((\sigma(B), Q), [!\_])$$
$$((\sigma(B), P), [!_{t'}]) \mapsto^* ((\overline{e}, Q), [!_{\mathfrak{r}(\_)}]) \mapsto ((\overline{e_r}, Q), [!\_]) \mapsto^* ((\overline{\sigma_j(B)}, Q'), [!\_]) \hookrightarrow ((\sigma(B), Q'), [!\_])$$

  Let $!_{d_B}$, $?_{d_e}\_$, $?_{d_l}\_$, $?_{d_r}\_$, $?_{d_i}\_$ and $?_{d_j}\_$ be the formulae labelling $\sigma(B)$, e, e_l, e_r, $\overline{\sigma_i(B)}$ and $\overline{\sigma_j(B)}$. To forbid such paths, we require that the label controlling dependence decreases along those paths and it strictly decreases in at least one of these steps: $d_B \geq d_e \geq d_l \geq d_i \geq d_B$, $d_B \geq d_e \geq d_r \geq d_j \geq d_B$ and at least one of those inequalities is strict. The choice we made in $SDNLL$ is to require the label to strictly decrease along one of the $\hookrightarrow$ step: either $d_i > d_B$ or $d_j > d_B$. On the contrary, in [58], Mazza requires the label to strictly decrease along one of the step crossing e upwards[6]: either $d_e > d_l$ or $d_e > d_r$. In [64], Roversi and Vercelli defined a framework MS where the step where the label must decrease can depend on d_B. For instance, we may define a system such that ($d_e = 3$ implies that either $d_l < 3$ or $d_r < 3$) and ($d_B = 5$ implies that either $d_i > 5$ or $d_j > 5$). Here, we require the even labels to strictly decrease on $?C$, and the odds labels to strictly decrease on $\hookrightarrow$.

In the definition of weak stratifications (Definition 108), we defined $\mathfrak{d}(B)$ (for $B \in B_G$) as an element of $\mathbb{N} - \{0\}$ because the bounds were easier to state this way, and the proofs were simpler. Here, to give a simpler and more intuitive definition of the system, the labels are elements of $\mathbb{N} \cup \{\bot\}$.

- If $\sigma(B)$ is labelled by $!_{\bot}\_$ then we will set $\mathfrak{d}(B) = 1$.

- If $\sigma(B)$ is labelled by $!_d\_$ with $d \in \mathbb{N}$, then we will set $\mathfrak{d}(B) = d + 2$.

For every set E, we define $E_\bot$ (resp. $E^\top$) as the set $E \cup \{\bot\}$ (resp. $E \cup \{\top\}$). If E admits an order $\leq$, the order is extended to $E_\bot$ (resp. $E^\top$) by setting, for every $e \in E$, $\bot \leq e \leq \top$. For every binary operation f on E and $e \in E$, we extend f by $f(\bot, e) = f(e, \bot) = \bot$ and $f(\top, e) = f(e, \top) = \top$. Similarly, we will represent booleans by the set $\{\bot, \top\} = \{\bot, \top\}$ and we define an order $\leq$ by: $a \leq b$ iff $a = \bot$ or $b = \top$.

We define $\mathcal{F}_{dc}$ by the following grammar (with X ranging over a countable set of variables and $d \in \mathbb{N}_\bot$).

$$\mathcal{F}_{dc} := X \mid X^\bot \mid \mathcal{F}_{dc} \otimes \mathcal{F}_{dc} \mid \mathcal{F}_{dc} \parr \mathcal{F}_{dc} \mid \forall X.\mathcal{F}_{dc} \mid \exists X.\mathcal{F}_{dc} \mid !_d\mathcal{F}_{dc} \mid ?_d\mathcal{F}_{dc}$$

In the first four subsystems ($SwLL_{dc}$, $SwLL_{nest}$, $SwLL_{used}$ and $SwLL_{last}$), to define the subsystem $SwLL_z$, we define a set of labels $L_z^?$ (here $L_{dc}^? = \mathbb{N}_\bot$). Then, $\mathcal{F}_z$ is defined by the following grammar (with X ranging over a countable set of variables, and $l \in L_z^?$):

$$\mathcal{F}_z := X \mid X^\bot \mid \mathcal{F}_z \otimes \mathcal{F}_z \mid \mathcal{F}_z \parr \mathcal{F}_z \mid \forall X.\mathcal{F}_z \mid \exists X.\mathcal{F}_z \mid !_l\mathcal{F}_z \mid ?_l\mathcal{F}_z$$

[6] $!A$ and $\S A$ can be viewed as $!_2A$ and $!_1A$

$$\frac{\vdash_a \Gamma, A \qquad A \leq_{dc} B \qquad a \leq b}{\vdash_b \Gamma, B} \qquad \frac{A \in \mathcal{F}_{dc} - \{?_\perp B\} - \{!_\perp B\}}{\vdash_\perp A, A^\perp} \qquad \frac{A \in \mathcal{F}_{dc}}{\vdash_\top !_\top A, ?_\top A^\perp} \qquad \frac{\vdash_a \Gamma, A \qquad \vdash_a \Delta, A^\perp}{\vdash_a \Gamma, \Delta}$$

$$\frac{\vdash_a \Gamma, A \qquad \vdash_a \Delta, B}{\vdash_a \Gamma, \Delta, A \otimes B} \qquad \frac{\vdash_a \Gamma, A, B}{\vdash_a \Gamma, A \parr B} \qquad \frac{\vdash_a \Gamma, A \qquad X \notin FV(\Gamma)}{\vdash_a \Gamma, \forall X.A} \qquad \frac{\vdash_a \Gamma, A[B/X] \qquad B \in \mathcal{F}_{dc}}{\vdash_a \Gamma, \exists X.A}$$

$$\frac{\vdash_a \Gamma, ?_{2d+1}A, ?_{2d+1}A}{\vdash_a \Gamma, ?_{2d+1}A} \qquad \frac{\vdash_a \Gamma, ?_{2d+2}A, ?_{2d+1}A}{\vdash_a \Gamma, ?_{2d+2}A} \qquad \frac{\vdash_a \Gamma, ?_\perp A, ?_\perp A}{\vdash_\top \Gamma, ?_\perp A} \qquad \frac{\vdash_a \Gamma, A}{\vdash_\top \Gamma, ?_\perp A} \qquad \frac{\vdash_a \Gamma}{\vdash_\top \Gamma, ?_\perp A}$$

$$\frac{\vdash_a \Gamma, ?_{2d+1}?_{2d}A}{\vdash_a \Gamma, ?_{2d+1}A} \qquad \frac{\vdash_a \Gamma, ?_{2d+2}?_{2d+1}A}{\vdash_a \Gamma, ?_{2d+2}A} \qquad \frac{\vdash_a \Gamma, ?_\perp ?_\perp A}{\vdash_\top \Gamma, ?_\perp A} \qquad \frac{\vdash_a A_1, \cdots, A_n, A}{\vdash_\top ?_\perp A_1, \cdots, ?_\perp A_n, !_\perp A}$$

$$B\,\frac{\vdash_a A_1, \cdots, A_n, A \qquad (a = \perp) \vee (B \in S)}{\vdash_a ?_{2d}A_1, \cdots, ?_{2d}A_n, !_{2d}A} \qquad B\,\frac{\vdash_a A_1, A_2 \cdots, A_n, A \qquad (a = \perp) \vee (B \in S)}{\vdash_a ?_{2d+1}A_1, ?_{2d+2}A_2, \cdots, ?_{2d+2}A_n, !_{2d+1}A}$$

Figure 5.14: Definition of $SwLL_{dc}$, d ranges over $\mathbb{N}$. In promotion rules, if $B \in S_{\boxdot\rightarrow}$, the indices on the ? must be strictly greater than the !.

Then, we define a subtyping order $\leq_z$. This order will be based on an order $\preccurlyeq_z^?$ on elements of $L_z^?$. Then $\leq_z$ by induction on formulae by:

- For every formula A, we have $A \leq_z A$.
- Let us suppose that $A \leq_z A'$ and $B \leq_z B'$ then $(A \otimes B) \leq_z (A' \otimes B')$, $(A \parr B) \leq_z (A' \parr B')$, $\forall X.A \leq_z \forall X.A'$ and $\exists X.A \leq_z \exists X.A'$.
- Let us suppose that $A \leq_z A'$ and $a, b \in L_z^?$ such that $a \preccurlyeq_z b$ then $?_a A \leq_z ?_b A'$ and $!_b A \leq_z !_a A'$.

Definition 188. *The proofs of $SwLL_{dc}$ are defined inductively by the rules of Figure 5.14. The judgements $\vdash_a \Gamma$ are composed of a non-empty list Γ of formulae (as usual) and an element a of $\{\perp, \top\}$. If $\vdash_\top \Gamma$ then every promotion rule below this judgement is in S.*

We also require that, if $\sigma_i(B) \in E_{\boxdot\rightarrow}$ then there are formulae $?_n A$ and $!_{n'}A'$ corresponding to $\sigma_i(B)$ and $\sigma(B)$ such that $n > n'$.

Definition 189. *We define a mapping from B_G to $\mathbb{N}_\perp$ by $\mathfrak{d}_{sw}(B) = d$ iff the highest $\sigma(B)$ is labelled by a formula of the shape $!_{d\_}$. We also define a partial mapping $\mathfrak{d}_{sw}(\_)$ on contexts by: $\mathfrak{d}_{sw}(((e, P), [!_t]@T)) = d$ iff $\boldsymbol{\beta}((e, P), [!_t]@T) = !_{d\_}$ (considering the lowest e).*

Lemma 190. *If $((e, P), T) \mapsto^* ((f, Q), U)$ then $\mathfrak{d}_{sw}(((e, P), T))$ is defined iff $\mathfrak{d}_{sw}(((f, Q), U))$ is defined. If they are defined $\mathfrak{d}_{sw}(((e, P), T)) \geq \mathfrak{d}_{sw}(((f, Q), U))$.*

Proof. Let us consider the steps where $\mathfrak{d}_{sw}(\_)$ is modified.

- It can be modified by the subtyping relation. In this case $P = Q$ and $T = U$. For $\mathfrak{d}_{sw}(((e, P), T))$ to differ from $\mathfrak{d}_{sw}(((f, P), T))$, the trace T must point to a ? or ! modality (not inside a variable).
  - Either e is a downward edge, f is below e (so $\beta(e) \leq_{dc} \beta(f)$) and T points to a ! modality. In this case, the label is smaller in the formula associated to f. Thus, $\mathfrak{d}_{sw}(((e, P), T)) \geq \mathfrak{d}_{sw}(((f, Q), U))$.
  - Or e is an upward edge, f is above e (so $\beta(f) \leq_{dc} \beta(e)$) and T points to a ? modality. In this case, the label is smaller in the formula associated to f. Thus, $\mathfrak{d}_{sw}(((e, P), T)) \geq \mathfrak{d}_{sw}(((f, Q), U))$.

- Or $T = [!_t]$ and the $\mapsto$ step from $((e, P), [!_t])$ to $((f, Q), U)$ is crossing a $?C$, $?N$ upwards or a $\hookrightarrow$ step. We can verify that in every case it decreases the labels: the labels above the contraction and digging rules are smaller or equal than below them, the label on the ! of a promotion rule is smaller (or equal) than on the ? modalities.

□

Lemma 191. *Let B be a box, if $\mathfrak{d}_{sw}(((e, P), [!_t])) = \bot$ then every box containing e is in S*

Proof. Because $\mathfrak{d}_{sw}(((e, P), [!_t]) = \bot, \beta(e)$ is of the shape $!_\bot A$ or $?_\bot A$. The conclusion of every possible rule creating e (axiom, contraction, digging, weakening, dereliction, and promotion) is of the shape $\vdash_\top \_$. Thus, every judgement below this rule is of the shape $\vdash_\top \Gamma$. Thus for every promotion rule below this judgement, the box is in S. □

Lemma 192. *If $\mathfrak{d}_{sw}(B) = \bot$ and $((\sigma(B), P), [!_t]) \mapsto^* (((e, Q), [!_u])$ then every box containing e is in S.*

Proof. We have $\mathfrak{d}_{sw}(((\sigma(B), P), [!_t])) \leq \mathfrak{d}_{sw}(B) = \bot$ so $\mathfrak{d}_{sw}(((\sigma(B), P), [!_t])) = \bot$. Because $((\sigma(B), P), [!_t]) \mapsto^* ((e, Q), [!_u])$, by Lemma 190, $\mathfrak{d}_{sw}(((e, Q), [!_u])) \leq \bot$ so $\mathfrak{d}_{sw}(((e, Q), [!_u])) = \bot$. By Lemma 191, every box containing e is in S. □

Lemma 193. *If $((\sigma(B), P), [!_t]) \mapsto^* ((\overline{\sigma_i(B')}, P'), [!_{t'}])$ $((\sigma(B), P), [!_{t'}]) \mapsto^* ((\overline{\sigma_{i'}(B')}, P''), [!_{t''}])$ with $i \neq i'$, and $\mathfrak{d}_{sw}(B) \in \mathbb{N}$ then $\mathfrak{d}_{sw}(B) > \mathfrak{d}_{sw}(B')$.*

Proof. The proof depends on whether $\mathfrak{d}_{sw}(B) = 2 \cdot k$ or $\mathfrak{d}_{sw}(B) = 2 \cdot k + 1$.

- If $\mathfrak{d}_{sw}(B) = 2{\cdot}k$, then let us consider the first i such that $((\sigma(B), P), [!_t]) \mapsto^i C_i \mapsto C_{i+1} = ((e, P), [!_u]@T)$, $((\sigma(B), P), [!_t]) \mapsto^i C'_i \mapsto C'_{i+1}$ and C'_{i+1} is not of the shape $((e, P), [!_{u'}]@T)$. By Lemma 190, $2 \cdot k \geq \mathfrak{d}_{sw}(C_i)$, $2 \cdot k \geq \mathfrak{d}_{sw}(C'_i)$, $\mathfrak{d}_{sw}(C_{i+1}) \geq \mathfrak{d}_{sw}(B')$ and $\mathfrak{d}_{sw}(C'_{i+1}) \geq \mathfrak{d}_{sw}(B')$. Concerning the $C_i \mapsto C_{i+1}$ and $C'_i \mapsto C'_{i+1}$ steps:
  - Either these steps cross a $?C$ node upwards with the trace of C_i and C'_i being of the shape $[!_{1(\_)}]$ and $[!_{\mathrm{r}(\_)}]$. Then, either $\mathfrak{d}_{sw}(C_i) > \mathfrak{d}_{sw}(C_{i+1})$ or $\mathfrak{d}_{sw}(C'_i) > \mathfrak{d}_{sw}(C'_{i+1})$.
  - Or these steps cross a $?N$ node upwards with the trace of C_i and C'_i being of the shape $[!_{\mathrm{n}(\_,v)}]$ and $[!_{\mathrm{n}(\_,v')}]$ with $v \neq v'$. In this case $\mathfrak{d}_{sw}(C_i) > \mathfrak{d}_{sw}(C_{i+1})$ and $\mathfrak{d}_{sw}(C'_i) > \mathfrak{d}_{sw}(C'_{i+1})$.

  In each case, it implies that $\mathfrak{d}_{sw}(B) = 2 \cdot k > \mathfrak{d}_{sw}(B')$ so $\mathfrak{d}_{sw}(B) \geq 1 + \mathfrak{d}_{sw}(B')$.
- If $\mathfrak{d}_{sw}(B) = 2 \cdot k + 1$, we know that $\mathfrak{d}_{sw}(B') \leq 2 \cdot k + 1$ by Lemma 190. If $\mathfrak{d}_{sw}(B') = 2 \cdot k + 1$ then let us notice that either $i \neq 1$ or $j \neq 1$. This would mean that either $\mathfrak{d}_{sw}(((\overline{\sigma_i(B')}, P'), [!_{t'}])) = 2 \cdot k + 2$ or $\mathfrak{d}_{sw}(((\overline{\sigma_{i'}(B')}, P''), [!_{t''}])) = 2 \cdot k + 2$. Each possibility would contradict Lemma 190.

□

5.3.2 Definition of $SwLL_{nest}$

The role of this subsystem is to control nesting by bounding the depth of $\dashv\!\!\lhd$. We will do so as in the premises of Theorem 119: we require that $(B, P) \dashv\!\!\lhd (B, Q)$ is only possible if every box containing B is in S. This is why, as in $SwLL_{dc}$ judgements, the judgements of $SwLL_{dc}$ will carry a boolean a whose meaning is "every promotion rule below this judgement is in S". The ! and ? connectives will carry a label (n, c) in $\mathbb{N}^\top_\bot \times \{\bot, \top\}$.

$$\frac{\vdash_a \Gamma, A \qquad A \leq_{nest} B \qquad a \leq b}{\vdash_b \Gamma, B} \qquad \frac{a \in \{\bot, \top\} \qquad A \in \mathcal{F}_{nest}}{\vdash_a A, A^\bot} \qquad \frac{\vdash_a \Gamma, A \qquad \vdash_a \Delta, A^\bot}{\vdash_a \Gamma, \Delta}$$

$$\frac{\vdash_a \Gamma, A \qquad \vdash_a \Delta, B}{\vdash_a \Gamma, \Delta, A \otimes B} \qquad \frac{\vdash_a \Gamma, A, B}{\vdash_a \Gamma, A \parr B} \qquad \frac{\vdash_a \Gamma, A \qquad X \notin FV(\Gamma)}{\vdash_a \Gamma, \forall X.A} \qquad \frac{\vdash_a \Gamma, A[B/X] \qquad B \in \mathcal{F}_{nest}}{\vdash_a \Gamma, \exists X.A}$$

$$\frac{\vdash_a \Gamma, ?_{n,\top}A, ?_{n,\top}A}{\vdash_a \Gamma, ?_{n,\top}A} \qquad \frac{\vdash_a \Gamma, A}{\vdash_a \Gamma, ?_{\bot,\bot}A} \qquad \frac{\vdash_a \Gamma}{\vdash_a \Gamma, ?_{\bot,\bot}A} \qquad \frac{\vdash_a \Gamma, ?_{n,\top}?_{n+1,\top}A}{\vdash_a \Gamma, ?_{n+1,\top}A} \qquad \frac{\vdash_a \Gamma, ?_{n,\top}?_{n,\bot}A}{\vdash_a \Gamma, ?_{n,\top}A}$$

$$\frac{\vdash_a A_1, \cdots, A_k, A \qquad (a = \bot) \vee (B \in S)}{\vdash_\top ?_{\bot,c}A_1, \cdots, ?_{\bot,c}A_k, !_{\bot,c}A} \qquad \frac{\vdash_a A_1, \cdots, A_k, A \qquad (a = \bot) \vee (B \in S)}{\vdash_\top ?_{\top,c}A_1, \cdots, ?_{\top,c}A_k, !_{\top,c}A}$$

$$B\ \frac{\vdash_a A_1, \cdots, A_k, A \qquad (a = \bot) \vee (B \in S) \qquad n \in \mathbb{N}}{\vdash_a ?_{n,c}A_1, \cdots, ?_{n,c}A_k, !_{n,c}A}$$

Figure 5.15: Definition of $SwLL_{nest}$

- The $n \in \mathbb{N}^\top_\bot$ plays a role quite similar to the third label (also usually denoted n) of $SDNLL$. Let us consider a box B such that $\beta(\sigma(B)) = !_{n,c\_}$. Provided that every box containing the box B is in S, n may be equal to $\bot$ or $\top$. In those cases, it is possible to have $(B, P) \dashv\!\lhd (B, Q)$. If $n = \bot$ then, as in $SwLL_{dc}$, for every box B such that $B \smile C$, $\beta(\sigma(C))$ is of the shape $!_{\bot,\_\_}$ (so every box containing such a C is in S). Similarly, if $n = \top$ and $C \smile B$ then $\beta(\sigma(C))$ is of the shape $!_{\top,\_\_}$(so every box containing such a C is in S).

- Intuitively, the difference between $\lhd\!\dashv$ and $\lhd$ is that, whenever $((\sigma(B), P), [!_{\mathtt{p}(u)}]) \mapsto^* ((\sigma(C), Q), [!_{\mathtt{e}}])$ it always implies that $B \lhd\!\dashv C$. This is why, when m is a $?N$ node, we have $\beta(concl_m) = ?_{n,\_}A$ and $\beta(prem_m) = ?_{n',\_}?_{\_,c}A$ we require that $n > n'$.

  On the contrary, $((\sigma(B), P), [!_{\mathtt{p}(u)}]) \mapsto^* ((\sigma(C), Q), [!_{\mathtt{e}}])$ implies $B \lhd C$ only if there exist two distinct copies $\mathfrak{n}(t_1, u)$ and $\mathfrak{n}(t_2, u)$ of (B, P). This is why, when m is a $?N$ node, $\beta(concl_m) = ?_{n,\_}A$ and $\beta(prem_m) = ?_{n',\_}?_{\_,c}A$, we may have $n = n'$ provided that there is no $\mapsto$-paths of the shape $((\overline{concl_m}, P), [!_{\mathfrak{n}(t,u)}]) \mapsto ((\overline{prem_m}, P), [!_t; !_u]) \mapsto^* ((\overline{concl_{m'}}, Q), [!\_])$ with m' a $?C$ node. To ensure this property, we will require (when $n = n'$) that $c = \bot$. And conclusions of $?C$ nodes can not be labelled by formulae of the shape $?_{\_,\bot}B$.

We define $L^?_{nest}$ as the set $\mathbb{N}^\top_\bot \times \{\bot, \top\}$. Thus, $\mathcal{F}_{nest}$ is defined by the following grammar (with X ranging over a countable set of variables, $n \in \mathbb{N}^\top_\bot$ and $c \in \{\bot, \top\}$).

$$\mathcal{F}_{nest} := X \mid X^\bot \mid \mathcal{F}_{nest} \otimes \mathcal{F}_{nest} \mid \mathcal{F}_{nest} \parr \mathcal{F}_{nest} \mid \forall X.\mathcal{F}_{nest} \mid \exists X.\mathcal{F}_{nest} \mid !_{n,c}\mathcal{F}_{nest} \mid ?_{n,c}\mathcal{F}_{nest}$$

We define $\preccurlyeq^?_{nest}$ by $(n, c) \preccurlyeq^?_{nest} (n', c')$ iff $(n < n')$ or $(n = n'$ and $c \leq c')$. Then the subtyping relation $\leq^?_{nest}$ is defined from $\preccurlyeq^?_{nest}$ as described in Section 5.3.1.

Definition 194. *The proofs of $SwLL_{nest}$ are defined inductively by the rules of Figure 5.15.*

Definition 195. *We define a mapping from B_G to $\mathbb{N}^\top_\bot$ by $\mathfrak{n}_{sw}(B) = n$ iff the highest $\sigma(B)$ is labelled by a formula of the shape $!_{n,\_}A$. We also define a partial mapping $\mathfrak{n}_{sw}(\_)$ on contexts by: $\mathfrak{n}_{sw}(((e, P), [!_t]@T)) = n$ iff $\boldsymbol{\beta}((e, P), [!_t]@T) = !_{n,\_\_}$ (considering the lowest e).*

Lemma 196. *If* $((e,P),T) \mapsto^* ((f,Q),U)$ *then* $\mathfrak{n}_{sw}(((e,P),T))$ *is defined iff* $\mathfrak{n}_{sw}(((f,Q),U))$ *is defined. If they are defined* $\mathfrak{n}_{sw}(((e,P),T)) \geq \mathfrak{n}_{sw}(((f,Q),U))$.

Proof. The proof is quite similar to the proof of Lemma 190. Let us consider the steps where $\mathfrak{n}_{sw}(\_)$ is modified.

- It can be modified by the subtyping relation. In this case $P = Q$ and $T = U$. For $\mathfrak{n}_{sw}(((e,P),T))$ to differ from $\mathfrak{n}_{sw}(((f,P),T))$, the trace T must point to a ? or ! modality (not inside a variable).
  - Either e is a downward edge, f is below e (so $\beta(e) \leq_{dc} \beta(f)$) and T points to a ! modality. In this case, the label is smaller in the formula associated to f. Thus, $\mathfrak{n}_{sw}(((e,P),T)) \geq \mathfrak{n}_{sw}(((f,Q),U))$.
  - Or e is an upward edge, f is above e (so $\beta(f) \leq_{dc} \beta(e)$) and T points to a ? modality. In this case, the label is smaller in the formula associated to f. Thus, $\mathfrak{n}_{sw}(((e,P),T)) \geq \mathfrak{n}_{sw}(((f,Q),U))$.
- Or $T = [!_t]$ and the $\mapsto$ step from $((e,P),[!_t])$ to $((f,Q),U)$ is crossing a $?N$ node upwards. This step decreases the labels: the labels above the digging rules are smaller or equal than below them.

□

Lemma 197. *If* $\boldsymbol{\beta}(C) = !_{n,c\_}$ *and* $C \mapsto^* D$ *then* $\boldsymbol{\beta}(D) = !_{n',c'\_}$ *and* $(n,c) \geqslant_{nest} (n',c')$.

Proof. Let us consider the steps where this label is modified.

- It can be modified by the subtyping relation. As in the proof of Lemma 190, the label can only decrease this way.
- If $C = ((\overline{e},P),[!_{\mathfrak{n}(t,u)}]) \mapsto ((\overline{f},P),[!_t;!_u]) = D$ (crossing a $?N$ node upwards) with e and f labelled by $?_{n,\top}A$ and $?_{n,\top}?_{n,\bot}A$, then $\boldsymbol{\beta}(C) = !_{n,\top\_}$ and $\boldsymbol{\beta}(D) = !_{n,\bot\_}$. By definition, we have $(n,\top) \geqslant_{nest} (n,\bot)$.
- If $C = ((\overline{e},P),[!_{\mathfrak{p}(u)}]) \mapsto ((\overline{f},P),[!_u]) = D$ (crossing a $?N$ node upwards) with e and f labelled by $?_{n+1,\top}A$ and $?_{n,\top}?_{n+1,\top}A$, then $\boldsymbol{\beta}(C) = !_{n+1,\top\_}$ and $\boldsymbol{\beta}(D) = !_{n,\top\_}$. By definition, we have $(n+1,\top) \geqslant_{nest} (n,\top)$.

□

Lemma 198. *Let B be a box, if* $\mathfrak{n}_{sw}(B) = \bot$ *or* $\mathfrak{n}_{sw}(B) = \top$ *then every box containing B is in S.*

Proof. The proof is the same as the proof of Lemma 191. □

Lemma 199. *If* $\mathfrak{n}_{sw}(B) \in \mathbb{N}$ *and* $(B,P) \dot{\lhd} (B',P')$ *then* $\mathfrak{n}_{sw}(B) > \mathfrak{n}_{sw}(B')$.

Proof. By definition of $\dot{\lhd}$ (Definition 112), there exist $t \neq u \in Cop(B,P)$ and a simplification v of t and u, and $((\sigma(B),P),[!_v]) \mapsto^* ((\sigma(B'),P'),[!_x])$ with x a standard signature. Let us consider the lowest i such that: $((\sigma(B),P),[!_t]) \mapsto^{i+1} ((e_{i+1},P_{i+1}),[!_{t_{i+1}}]@T_{i+1})$, $((\sigma(B),P),[!_u]) \mapsto^i C$ and C is not of the shape $((e_{i+1},P_{i+1}),[!_{u_{i+1}}]@T_{i+1})$ with $u_{i+1} \sqsupseteq t_{i+1}$. The only possibility is that the last step crosses a $?N$ node upwards. Thus, we have paths of the following shape (with $v' \sqsupseteq w$)

$$((\sigma(B),P),[!_t]) \mapsto^i ((\overline{e},Q),[!_{\mathfrak{n}(t',w)}]) \mapsto ((\overline{f},Q),[!_{t'};!_w])$$
$$((\sigma(B),P),[!_u]) \mapsto^i ((\overline{e},Q),[!_{\mathfrak{n}(u',w)}]) \mapsto ((\overline{f},Q),[!_{u'};!_w])$$
$$((\sigma(B),P),[!_v]) \mapsto^i ((\overline{e},Q),[!_{\mathfrak{p}(v')}]) \mapsto ((\overline{f},Q),[!_{v'}])$$

$$\frac{\vdash \Gamma, A \qquad A \leq_{last} B}{\vdash \Gamma, B} \qquad \frac{A \in \mathcal{F}_{last}}{\vdash A, A^{\perp}} \qquad \frac{\vdash \Gamma, A \qquad \vdash \Delta, A^{\perp}}{\vdash \Gamma, \Delta} \qquad \frac{\vdash \Gamma, A \qquad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B} \qquad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}$$

$$\frac{\vdash \Gamma, A \qquad X_x \notin FV(\Gamma)}{\vdash \Gamma, \forall X_x.A} \qquad \frac{\vdash \Gamma, A[B/X_{\perp}] \qquad B \in \mathcal{F}_{last}}{\vdash \Gamma, \exists X_{\perp}.A} \qquad \frac{\vdash \Gamma, A[B/X_{\top}] \qquad B \in \mathcal{F}^{\top}_{last}}{\vdash \Gamma, \exists X_{\top}.A}$$

$$\frac{\vdash \Gamma, ?_{\perp}?_{\perp}A \qquad A \in \mathcal{F}^{\top}_{last}}{\vdash \Gamma, ?_{\top}A} \qquad \frac{\vdash \Gamma \qquad A \in \mathcal{F}^{\top}_{last}}{\vdash \Gamma, ?_{\top}A} \qquad \frac{\vdash \Gamma, ?_{\perp}A, ?_{\perp}A \qquad A \in \mathcal{F}^{\top}_{last}}{\vdash \Gamma, ?_{\top}A}$$

$$\frac{\vdash \Gamma, A}{\vdash \Gamma, ?_{\top}A} \qquad \frac{\vdash \Gamma, e : ?_{\top}A \qquad e \in E_{\circ}}{\vdash \Gamma, e : ?_{\perp}A} \qquad \frac{\vdash A_1, \cdots, A_k, A \qquad f_1 \in E_{\circ}, \ldots f_k \in E_{\circ}}{\vdash f_1 : ?_{\top}A_1, \cdots, f_k : ?_{\top}A_k, !_{\top}A}$$

Figure 5.16: Definition of $SwLL_{last}$.

- Either e and f are labelled by formulae of the shape $?_{n,\top}?_{n+1,\top}A$ and $?_{n+1,\top}A$. By Lemma 196, $\mathfrak{n}_{sw}(B) \geq \mathfrak{n}_{sw}(((\sigma(B), P), [!_v])) \geq n+1$ and $n \geq \mathfrak{n}_{sw}(B')$. Thus $\mathfrak{n}_{sw}(B) > \mathfrak{n}_{sw}(B')$.
- Or e and f are labelled by formulae of the shape $?_{n,\top}?_{n,\perp}A$ and $?_{n,\top}A$. Because $t \neq u$, we have $t' \neq u'$. So either $t' \neq \mathtt{e}$ or $u' \neq \mathtt{e}$. Thus, there exists a path of the shape $((\overline{f}, Q), [!_{t'}; !_w]) \mapsto^* ((\overline{g}, R), [!\_])$ with g the conclusion of a $?C$ node or $?N$ node. Let $?_{n'',\top}\_$ be the label of g. By Lemma 196, we have $\mathfrak{n}_{sw}(B) \geq n$, $(n, \perp) \geqslant_{nest} (n'', \top)$ and $n'' \geq \mathfrak{n}_{sw}(B')$. By definition of $\geqslant_{nest}$, $(n, \perp) \geqslant_{nest} (n'', \top)$ implies that $n > n''$. Thus, we have $\mathfrak{n}_{sw}(B) > \mathfrak{n}_{sw}(B')$.

□

5.3.3 Definition of $SwLL_{last}$

The system $SwLL_{last}$ restricts the contexts of the shape $((e, P), [!_u])$ such that there exists $(B, P, t) \in BoxSig_G$ such that $C_{B,P,t} = ((e, P), [!_u])$. For $(B, P) \in Pot(B_G)$ and $t \in Cop(B, P)$ we define (B, P, t) as the last[7] context of the $\rightsquigarrow$ path beginning by $((\sigma(B), P), [!_t])$ of the shape $((e, P), [!_u])$ with $e \in E_{\circ}$. In $SwLL_{last}$, the ! and ? modalities carry a label l in $\{\perp, \top\}$. Let us consider an edge f labelled by $!_{\top}A$, for every path of the shape $((\overline{e}, P), [!_t]) \mapsto^* ((f, Q), [!_u])$ with e the conclusion of a $?C$ or $?N$ node, there exists a context of the shape $((g, R), [!\_])$ in the path with $g \in E_{\circ}$ (Lemma 201).

We define $L^{?}_{last}$ as the set $L^{?}_{used} = \{\perp, \top\}$ and $\leqslant^{?}_{last} = \leqslant^{?}_{used}$. In this subsystem, variables are also labelled by elements of $L^{X}_{last} = \{\perp, \top\}$. Then $\mathcal{F}_{last}$ is defined by the following grammar (with X ranging over a countable set of variables, $l \in L^{?}_{last}$ and $x \in L^{X}_{last}$):

$$\mathcal{F}_{last} := X_x \mid X_x^{\perp} \mid \mathcal{F}_{last} \otimes \mathcal{F}_{last} \mid \mathcal{F}_{last} \parr \mathcal{F}_{last} \mid \forall X_x.\mathcal{F}_{last} \mid \exists X_x.\mathcal{F}_{last} \mid !_l\mathcal{F}_{last} \mid ?_l\mathcal{F}_{last}$$

Then, $\mathcal{F}^{\top}_{last}$ refers to the formulae of $\mathcal{F}_{last}$ whose labels (both on exponential modalities and on variables) are all equal to $\top$.

Definition 200. *The proofs of $SwLL_{last}$ are defined inductively by the rules of Figure 5.16.*

Lemma 201. *Let us suppose that e is the conclusion of a $?C$ or $?N$ node, f is labelled by $!_{\top}A$ and $((\overline{e}, P), [!_t]) \mapsto^* ((f, Q), [!_u])$. Then, there exists a context of the shape $((\overline{g}, R), [!\_])$ in the path with $g \in E_{\circ}$.*

[7] The name of the system comes from this definition.

Proof. Let C be the context such that $((\overline{e}, P), [!_t]) \mapsto C$. By definition of $SwLL_{last}$, $\boldsymbol{\beta}(C)$ is of the shape $!_\bot A$. And $\boldsymbol{\beta}((f, Q), [!_u])$ is of the shape $!_\top A'$. Let us consider the first context of the path where the label changes from $\bot$ to $\top$. Most of the steps decrease the labels (so we can only go from $\top$ to $\bot$) as in the proofs of Lemmas 190, 196 and 206. The only rule changing the label the other way is the second of the last line. Thus there exists an edge $g \in E_\circ$ such that the path goes through the context $((\overline{g}, \_), [!\_])$. □

Lemma 202. *Let $((e, P), [!_t]@T)$ be a context such that $\boldsymbol{\beta}((e, P), [!_t]@T)$ is defined. If $\beta(e) \in \mathcal{F}^\top_{last}$, then $\boldsymbol{\beta}((e, P), [!_t]@T) \in \mathcal{F}^\top_{last}$.*

Proof. First we can notice that, for every (f, P) in $Pot(\mathrm{N}^\forall_)$ such that $\beta concl_f$ is of the shape $\forall X_\top.A$ and $\phi_G(f, P) = (B, (e, Q))$ then $B \in \mathcal{F}^\top_{last}$. Indeed, because the labels on variables can not be modified by subtyping relation, for every context C of the path $((concl_f, P), [\forall]) \mapsto^* ((\overline{concl_e}, Q), [\forall])$ is of the shape $\forall X_\top.\_$. In particular, $\beta(concl_e)$ is of the shape $\exists X_\top.A'$. Thus, by Definition of $SwLL_{last}$, the formula B associated with e is in $\mathcal{F}^\top_{last}$.

Then, we can deduce that for every (f, P) in $Pot(\mathrm{N}^\forall_G)$ such that $\beta(concl_f)$ is of the shape $\forall X_\top.A$, $\phi^{lim}_G(f, P) \in \mathcal{F}^\top_{last}$. We prove it by induction on $<_\forall$. By definition of ϕ^{lim}_G, either $\phi_G(f, P)$ is undefined and $\phi^{lim}_G(f, P) = X_\top \in \mathcal{F}^\top_{last}$. Or $\phi_G(f, P) = (B, (e, [q_1; \cdots; q_{\partial(m)}]))$, $Z^1_{l_1}, \cdots, Z^k_{l_k}$ are respectively associated to $\forall$ nodes $f_1, \cdots, f_k$ and

$$\phi^{lim}_G(f, P) = B[\phi^{lim}_G(f_1, [q_1; \cdots; q_{\partial(f_1)}])/Z_1; \cdots; \phi^{lim}_G(f_k, [q_1; \cdots; q_{\partial(f_k)}])/Z_k]$$

According to the previous paragraph, B is in $\mathcal{F}^\top_{last}$ so $l_1 = \cdots = l_k = \top$. Thus, by induction hypothesis, $\phi^{lim}_G(f_1, [q_1; \cdots; q_{\partial(f_1)}]), \cdots, \phi^{lim}_G(f_k, [q_1; \cdots; q_{\partial(f_k)}]) \in \mathcal{F}^\top_{last}$. By definition of $\mathcal{F}^\top_{last}$, $\phi^{lim}_G(f, P) \in \mathcal{F}^\top_{last}$.

Finally, by definition, of $\boldsymbol{\beta}((e, P), [!_t]@T) = \beta(\beta(e)[\theta(e, P)], e, P, T, [])_{|T}$. By supposition, $\beta(e) \in \mathcal{F}^\top_{last}$ so every eigenvariable of $\beta(e)$ is labelled by $\top$. Thus, according to the previous paragraph, $\beta(e)[\theta(e, P)] \in \mathcal{F}^\top_{last}$. In the computation of $\beta(\beta(e)[\theta(e, P)], e, P, T, [])$, for every step of the shape $\beta(\forall X_x.A, e, P, U.\forall, V)$ with $((concl_f, Q), [\forall] \rightsquigarrow^* ((e, P), [\forall]@V)$, we have $x \in \top$ so $\beta(concl_f)$ is of the shape $\forall X_\top.\_$ and $\phi^{lim}_{f,Q} \in \mathcal{F}^\top_{last}$. The steps $\beta(\exists X_x.A, e, P, U.\exists, V)$ are similar. So, $\boldsymbol{\beta}((e, P), T)$ is in $\mathcal{F}^\top_{last}$. □

Lemma 203. *Let us consider a $SwLL_{last}$ proof-net G such that no $\bot$ appears in its conclusions. For every $(B, P, t) \in CanCop_G$, there exists a context $C_{B,P,t} = ((e, Q), [!_u])$ such that $((\sigma(B), P), [!_t]) \rightsquigarrow^* C_{B,P,t}$ and,*

- *Either $e \in E_\circ$ or $C_{B,P,t} = ((\sigma(B), P), [!_t])$.*
- *If $((e, Q), [!_u]) \rightsquigarrow^k ((\overline{\sigma(\_)}, \_), [!\_])$ then we have $k = 0$.*
- *If e is not an auxiliary door, then $u = \mathsf{e}$.*

Proof. By definition of $CanCop_G$, there exists a path of the shape $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((f, R), [!_v]@T) \not\rightsquigarrow$. Let $!_l A = \boldsymbol{\beta}((f, R), [!_v]@T)$, let us prove that $l = \top$. Because $((f, R), [!_v]@T) \not\vdash$, we are in one of the following cases:

- $\overline{f}$ is the conclusion of a $?W$ node, a $?N$ node or a $?C$ node. In those cases, let us notice that $\beta(f) \in \mathcal{F}^\top_{last}$ (by definition of $SwLL_{last}$). Thus, by Lemma 202, $\boldsymbol{\beta}((f, R), [!_v]@T)$ is in $\mathcal{F}^\top_{last}$.
- $\overline{f}$ is the conclusion of a $?D$ or $?P$ node and $T = []$. In these cases, $\beta(\overline{f})$ is of the shape $?_\top A'$ and $\boldsymbol{\beta}((f, R), [!_v]) = \beta(f)[\theta(f, R)]$. So $l = \top$.
- f is a conclusion of G. In this case, by assumption $\beta(f) \in \mathcal{F}^\top_{last}$. Thus, by Lemma 202, $\boldsymbol{\beta}((f, R), [!_v]@T)$ is in $\mathcal{F}^\top_{last}$.

Let us consider the last context of the path $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((f, R), [!_v]@T)$ of the shape $((e, Q), [!_u])$ with $e \in E_\circ$. If such a context does not exist, we set $((e, Q), [!_u]) = ((\sigma(B), P), [!_t])$. By definition, $((\sigma(B), P), [!_t]) \rightsquigarrow^* C_{B,P,t}$ and (either $e \in E_\circ$ or $C_{B,P,t} = ((\sigma(B), P), [!_t])$)

- If $((e, Q), [!_u]) \rightsquigarrow^k ((\overline{\sigma_i(B)}, \_), [!\_])$ then this context is $((f, R), [!_v]@T)$. By definition of $SwLL_{last}$, $\sigma_i(B)$ is in $E_\circ$. So $((e, Q), [!_u]) = ((f, R), [!_v]@T)$ and $k = 0$.

- Let us suppose that e is not an auxiliary door, then (according to the previous item) $((f, R), [!_v]@T)$ is not of the shape $((\overline{\sigma(\_)}, \_), [!\_])$ and $((f, R), [!_v]@T) \not\rightarrow$. Thus, $((f, R), [!_v]@T) \not\rightarrow$ and by definition of copies it implies that $v = \mathtt{e}$. If we had $u \neq \mathtt{e}$ then, in the path $((e, Q), [!_u]) \rightsquigarrow^* ((f, R), [!_\mathtt{e}]@T)$, there would be a step of the shape $((concl_n, P'), [!\_]) \rightsquigarrow C$ with n a $?C$ node or $?N$ node. Let us notice that in every case $\boldsymbol{\beta}(C)$ is of the shape $!_{\perp\_}$. Because there is no context of the shape $((e', Q'), [!_{u'}])$ with $e' \in E_\circ$ in the path from C to $((f, R), [!_v]@T)$ it implies that $\boldsymbol{\beta}((f, R), [!_v]@T)$ is of the shape $!_{\perp\_}$. However, we proved that $l = \top$. So $u \neq \mathtt{e}$ leads to a contradiction, we have $u = \mathtt{e}$.

□

5.3.4 Definition of $SwLL_{strat}$

The subsystem $SwLL_{strat}$ enforces stratification. The ! and ? modalities are decorated with three labels:

- A label $s \in \mathbb{N}_\perp$ corresponding to the mapping $\mathfrak{s}(\_)$ of weak stratifications. If $\sigma(B)$ is labelled by $!_{\perp,\_,\_}A$ then $B \in S$. If $\sigma(B)$ is labelled by $!_{n,\_,\_}A$ with $n \in \mathbb{N}$ then we will set $\mathfrak{s}(B) = n + 1$.

- A label $w \in \{\blacklozenge, \lozenge, \top\}$ helping us to capture the notion of "entering definitely a box". Let us suppose that $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(B')}, P'), T'.?_{t'}]) \rightsquigarrow^* ((e, Q), [!_u])$, with $\sigma(B)$, $\sigma(B')$ and e being labelled respectively by $!_{s,\_,\_}A$ and $!_{s',\_,\_}A'$ and $!_{t,w,\_}B$. If $((\overline{\sigma(B')}, P'), T'.?_{t'}])$ is a used context, we are in one of the following conditions:

  - $w = \top$ and $s \geq 1 + s'$. So $\mathfrak{s}(B) \geq \mathfrak{s}(B') + 1$. This is what happens in $SDNLL$.
  - $w = \blacklozenge$ and the path $((e, Q), [!_u]) \rightsquigarrow^* C_{B,P,t}$ leaves (B', P', t').
  - $w = \lozenge$ and $(B, P, t) \boxdot\!\!\rightarrow (B', P', t')$.

- A label $u \in \{\perp, \top\}$ helping us to capture the notion of used contexts. The idea is that, if $((e, P), [!_t])$ is a used context, then e is labelled by a formula of the shape $!_{\_,\_,\perp}A$ (Lemma 208).

Let $L^?_{strat} = \mathbb{N}_\perp \times \{\blacklozenge, \lozenge, \top\} \times \{\perp, \top\}$ and $L^X_{strat} = \mathbb{N}_\perp \times \{\blacklozenge, \lozenge, \top\}$. Then we define $\mathcal{F}_{strat}$ by the following grammar (with X ranging over a countable set of variables, $l \in L^?_{strat}$ and $x \in L^X_{strat}$)

$$\mathcal{F}_{strat} := X_x \mid X^\perp_x \mid \mathcal{F}_{strat} \otimes \mathcal{F}_{strat} \mid \mathcal{F}_{strat} \parr \mathcal{F}_{strat} \mid \forall X.\mathcal{F}_{strat} \mid \exists X.\mathcal{F}_{strat} \mid !_l\mathcal{F}_{strat} \mid ?_l\mathcal{F}_{strat}$$

We define an order $\leq$ on $\{\blacklozenge, \lozenge, \top\}$ by $\top \geq \lozenge \geq \blacklozenge$. We also define $\preccurlyeq_{strat}$ on $L^?_{strat}$ by $(s, w, u) \leq (s', w', u')$ iff ($s \leq s'$, $w \leq w'$ and $u \leq u'$). Then, we define a subtyping order $\leq_{strat}$ from this order similarly to the definitions of subtyping orders in previous sections.

- For every formula A, we have $A \leq_{strat} A$.

- Let us suppose that $A \leq_{strat} A'$ and $B \leq_{strat} B'$ then $(A \otimes B) \leq_{strat} (A' \otimes B')$, $(A \parr B) \leq_{strat} (A' \parr B')$, $\forall X.A \leq_{strat} \forall X.A'$ and $\exists X.A \leq_{strat} \exists X.A'$.

$$\frac{\vdash \Gamma, A \qquad A \leq_{strat} B}{\vdash \Gamma, B} \qquad \frac{A \in \mathcal{F}_{strat}}{\vdash A, A^{\perp}} \qquad \frac{\vdash \Gamma, A \qquad \vdash \Delta, A^{\perp}}{\vdash \Gamma, \Delta} \qquad \frac{\vdash \Gamma, A \qquad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B} \qquad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}$$

$$\frac{\vdash \Gamma, A \qquad X_{s,\top}, X_{s,\blacklozenge}, X_{s,\lozenge} \notin FV(\Gamma)}{\vdash \Gamma, \forall X.A} \qquad \frac{\vdash \Gamma, A\{B/X_s\} \qquad B \in \mathcal{F}_{s,\lozenge} \cap \overline{\mathcal{F}_{s,\lozenge}}}{\vdash \Gamma, \exists X.A}$$

$$B\,\frac{\vdash A_1, \cdots, A_k, A \qquad A_1, \cdots, A_k \in \mathcal{F}_{\perp,\top} \qquad A \in \mathcal{F}_{s,\lozenge} \qquad B \in S_{\boxdot\mapsto}}{\vdash e_1 : ?_{s_1,w_1,u}A_1, \cdots, e_k : ?_{s_k,w_k,u}A_k, !_{s,\top,u}A_{\top}} \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, ?_{0,\lozenge,\perp}A} \qquad \frac{\vdash \Gamma}{\vdash \Gamma, ?_{0,\lozenge,\perp}A}$$

$$B\,\frac{\vdash A_1, \cdots, A_k, A \qquad A_1, \cdots, A_k \in \mathcal{F}_{\perp,\top} \qquad A \in \mathcal{F}_{s,\blacklozenge}}{\vdash e_1 : ?_{s_1,w_1,u}A_1, \cdots, e_k : ?_{s_k,w_k,u}A_k, !_{s,\top,u}A_{\top}} \qquad \frac{\vdash \Gamma, ?_{s,w_1,\perp}A_1, ?_{s,w_2,\perp}A_2}{\vdash \Gamma, ?_{s,w_1+w_2,\perp}(A_1 + A_2)} \qquad \frac{\vdash \Gamma, ?_{s,w_1,\perp}?_{s,w_2,\perp}A}{\vdash \Gamma, ?_{s,w_1+w_2,\perp}A}$$

Figure 5.17: Definition of $SwLL_{strat}$

- If $A \leq_{strat} A'$ and $a, b \in L^{?}_{strat}$ such that $a \preccurlyeq_{strat} b$ then $?_a A \leq_{strat} ?_b A'$ and $!_b A \leq_{strat} !_a A'$.

We also define an operation + on $\{\blacklozenge, \lozenge, \top\}$ by $\blacklozenge + \blacklozenge = \blacklozenge$, $\blacklozenge + \lozenge = \lozenge + \blacklozenge = \lozenge$, $\top + \top = \top$ and $w_1 + w_2$ is otherwise undefined. Let us notice that, whenever $w_1 + w_2 = w$, we have $w_1 \leq w$ and $w_2 \leq w$. The operation + is extended on formula by $X_{s,w_1} + X_{s,w_2} = X_{s,w_1+w_2}$, $X^{\perp}_{s,w_1} + X^{\perp}_{s,w_2} = X^{\perp}_{s,w_1+w_2}$, $(A \otimes B) + (A' \otimes B') = (A + A') \otimes (B + B')$, $(A \parr B) + (A' \parr B') = (A + A') \parr (B + B')$, $(\forall X.A) + (\forall X.A') = \forall X.(A + A')$, $(\exists X.A) + (\exists X.A') = \exists X.(A + A')$, $(!_{s,w,u}A) + (!_{s,w,u}A') = !_{s,w,u}(A + A')$ and finally $(?_{s,w_1,u}A) + (?_{s,w_2,u}A') = ?_{s,w_1+w_2,u}(A + A')$. In the other cases, $A + A'$ is undefined.

We wrote above that for every path of the form $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(B')}, P'), T'.?_{t'}]) \rightsquigarrow^* ((e, Q), [!_u])$ with $((\overline{\sigma(B')}, P'), T'.?_{t'}])$ a used context, this path satisfies one of three possible conditions. To enforce those conditions, we require $\sigma(B')$ to be of the shape $!_{s',\top,\_}A$ and $A \in \mathcal{F}_{s,\lozenge}$ with $\mathcal{F}_{s,\lozenge}$ a set of formulae defined below. Intuitively, in this case there exists a hole-formula H such that $A = H[?_{s,w,u}A']$ and this ? corresponds to $((\sigma(B), P), [!_t])$: it is to say $\beta(\sigma(B'))_{T'} = ?_{s,w,u}A'$. Then, by definition of $\mathcal{F}_{s,\lozenge}$, we are in one of the following situations: Either ($w = \top$ and $s \geq 1 + s'$), or $w = \blacklozenge$, or ($w = \lozenge$ and $s \geq s'$).

For $s \in \mathbb{N}_{\top}$, we define sets of formulae $\mathcal{F}_{s,\top}$, $\mathcal{F}_{s,\blacklozenge}$ and $\mathcal{F}_{s,\lozenge}$. In the formulae of $\mathcal{F}_{s,\top}$ neither $\blacklozenge$ nor $\lozenge$ appear. In the formulae of $\mathcal{F}_{s,\blacklozenge}$, $\lozenge$ does not appear. In the formulae of $\mathcal{F}_{\lozenge}$, the label $\lozenge$ appears at most once. Moreover, if $A \in \mathcal{F}_{s,\blacklozenge}$ or $A \in \mathcal{F}_{s,\lozenge}$, for every $?_{t,\_,\perp}$ and $X_{t,\blacklozenge}$ in A, we have $t \geq s + 1$. In the following grammars, X ranges over a countable set of variables $s, t, s' \in \mathbb{N}_{\perp}$ with $t \geq s + 1$, $w \in \{\blacklozenge, \top\}$, $w' \in \{\blacklozenge, \lozenge, \top\}$ and $u \in \{\perp, \top\}$.

$$\begin{aligned}
\mathcal{F}_{s,\top} :=& \mathcal{F}_{t,\top} \mid X_{s,\top} \mid X^{\perp}_{s,\top} \mid \mathcal{F}_{s,\top} \otimes \mathcal{F}_{s,\top} \mid \mathcal{F}_{s,\top} \parr \mathcal{F}_{s,\top} \mid \forall X.\mathcal{F}_{s,\top} \mid \exists X.\mathcal{F}_{s,\top} \mid !_{s',\top,u}\mathcal{F}_{s,\top} \mid ?_{t,\top,u}\mathcal{F}_{s,\top} \\
\mathcal{F}_{s,\blacklozenge} :=& \mathcal{F}_{t,\blacklozenge} \mid \mathcal{F}_{s,\top} \mid X_{s,\blacklozenge} \mid X^{\perp}_{s,\blacklozenge} \mid \mathcal{F}_{s,\blacklozenge} \otimes \mathcal{F}_{s,\blacklozenge} \mid \mathcal{F}_{s,\blacklozenge} \parr \mathcal{F}_{s,\blacklozenge} \mid \forall X.\mathcal{F}_{s,\blacklozenge} \mid \exists X.\mathcal{F}_{s,\blacklozenge} \mid !_{s',w,u}\mathcal{F}_{s,\blacklozenge} \mid \\
& ?_{t,\top,u}\mathcal{F}_{s,\blacklozenge} \mid ?_{s',\blacklozenge,u}\mathcal{F}_{s,\blacklozenge} \mid ?_{s',w,\top}\mathcal{F}_{s,\blacklozenge} \\
\mathcal{F}_{s,\lozenge} :=& \mathcal{F}_{t,\lozenge} \mid \mathcal{F}_{s,\blacklozenge} \mid X_{s,\lozenge} \mid X^{\perp}_{s,\lozenge} \mid \mathcal{F}_{s,\lozenge} \otimes \mathcal{F}_{s,\blacklozenge} \mid \mathcal{F}_{s,\blacklozenge} \otimes \mathcal{F}_{s,\lozenge} \mid \mathcal{F}_{s,\lozenge} \parr \mathcal{F}_{s,\blacklozenge} \mid \mathcal{F}_{s,\blacklozenge} \parr \mathcal{F}_{s,\lozenge} \mid \forall X.\mathcal{F}_{s,\lozenge} \mid \exists X.\mathcal{F}_{s,\lozenge} \mid \\
& !_{s',w',u}\mathcal{F}_{s,\lozenge} \mid ?_{t,\top,u}\mathcal{F}_{s,\lozenge} \mid ?_{s',\blacklozenge,u}\mathcal{F}_{s,\lozenge} \mid ?_{s',w',\top}\mathcal{F}_{s,\lozenge} \mid ?_{s,\lozenge,u}\mathcal{F}_{s,\blacklozenge}
\end{aligned}$$

We can observe that for every $s \in \mathbb{N}_{\perp}$, $\mathcal{F}_{s,\top} \subseteq \mathcal{F}_{s,\blacklozenge} \subseteq \mathcal{F}_{s,\lozenge}$. Moreover, if $s \leq t \in \mathbb{N}_{\perp}$, we have $\mathcal{F}_{s,\top} \supseteq \mathcal{F}_{t,\top}$, $\mathcal{F}_{s,\blacklozenge} \supseteq \mathcal{F}_{t,\blacklozenge}$ and $\mathcal{F}_{s,\lozenge} \supseteq \mathcal{F}_{t,\lozenge}$.

Then, we define projections $(\,)_{\blacklozenge}$ and $(\,)_{\top}$ of $\mathcal{F}_{s,\lozenge}$ on $\mathcal{F}_{s,\blacklozenge}$ and $\mathcal{F}_{\perp,\top}$. The formula $A_{\blacklozenge}$ is obtained from A by replacing the possible $X_{s,\lozenge}$, $X^{\perp}_{s,\lozenge}$ or $?_{s,\lozenge,u}$ with (respectively) $X_{s,\blacklozenge}$, $X^{\perp}_{s,\blacklozenge}$ and $?_{s,\blacklozenge,u}$. The formula $A_{\top}$ is obtained

from A by replacing the $X_{s,w}$, $X^{\perp}_{s,w}$, and $?_{s,w,u}$ with (respectively) $X_{s,\top}$, $X^{\perp}_{s,\top}$ and $?_{s,\top,u}$. Because $\mathcal{F}_{s,\blacklozenge} \subseteq \mathcal{F}_{s,\Diamond}$, $(\,)_\top$ can be seen as a projection of $\mathcal{F}_{s,\blacklozenge}$ on $\mathcal{F}_{s,\top}$.

Then we define a refinement of usual substitution as follows. For $A, B \in \mathcal{F}_{strat}$ and variable X, we write $A\{B/X_s\}$ for $A[B/X_{s,\Diamond}; B_\blacklozenge/X_{s,\blacklozenge}; B_\top/X_{s,\top}]$.

Lemma 204. *If $A \in \mathcal{F}_{s,\Diamond}$ (resp. $\mathcal{F}_{s,\blacklozenge}$, $\mathcal{F}_{s,\top}$) and $B \in \mathcal{F}_{s,\Diamond} \cap \overline{\mathcal{F}_{s,\Diamond}}$ then $A\{B/X_s\} \in \mathcal{F}_{s,\Diamond}$ (resp. $\mathcal{F}_{s,\blacklozenge}$, $\mathcal{F}_{s,\top}$).*

Proof. We prove it by induction on A. We only prove the result for $A \in \mathcal{F}_{s,\Diamond}$ (the proofs for $A \in \mathcal{F}_{s,\blacklozenge}$ and $A \in \mathcal{F}_{s,\top}$ are similar). For every formula A (with possible indices), we write A^X for $A[B/X_{s,\Diamond}; B_\blacklozenge/X_{s,\blacklozenge}]$. If A does not contain X, then $A^X = A \in \mathcal{F}_{s,\Diamond}$. If $A \in \mathcal{F}_{t,\Diamond}$ with $t \geq s+1$ then by induction hypothesis $A^X \in \mathcal{F}_{t,\Diamond} \subseteq \mathcal{F}_{s,\Diamond}$. If $A \in \mathcal{F}_{s,\blacklozenge}$ then $A^X \in \mathcal{F}_{s,\blacklozenge} \subseteq \mathcal{F}_{s,\Diamond}$. If $A \in X_{s,\Diamond}$ then $A^X = A \in \mathcal{F}_{s,\Diamond} \cap \overline{\mathcal{F}_{s,\Diamond}} \subseteq \mathcal{F}_{s,\Diamond}$. If $A \in X^{\perp}_{s,\Diamond}$ then $A^X = A^\perp \in \overline{\mathcal{F}_{s,\Diamond}} \cap \mathcal{F}_{s,\Diamond} \subseteq \mathcal{F}_{s,\Diamond}$. If $A = A_1 \otimes A_2$ with $A_1 \in \mathcal{F}_{s,\Diamond}$ and $A_2 \in \mathcal{F}_{s,\blacklozenge}$ then $A^X = A_1^X \otimes A_2^X$. By induction hypothesis, $A_1^X \in \mathcal{F}_{s,\Diamond}$ and $A_2^X \in \mathcal{F}_{s,\blacklozenge}$ so $A^X \in \mathcal{F}_{s,\Diamond}$. The other cases are solved similarly by using the induction hypothesis. □

Definition 205. *The proofs of $SwLL_{strat}$ are defined inductively by the rules of Figure 5.17. Moreover, we require that:*

- *If $\sigma(B)$ is labelled by $!_{\perp,\_,\_}$ then $B \in S$.*
- *For every edge $\sigma_i(B)$ labelled by $?_{s_i,w_i}$, either $w_i = \top$ or ($w_i = \Diamond$ and $e_i \in E_{\boxdot\to}$).*
- *If $e \in E_\circ$ then e is labelled by a formula of the shape $?_{\_,w,\_}A$ with $w \in \{\Diamond, \top\}$.*

Lemma 206. *If $C \mapsto^* C'$, $\boldsymbol{\beta}(C) = !_{s,\_,u}A$ and $\boldsymbol{\beta}(C') = !_{s',\_,u'}A'$ then $s \geq s'$ and $u \geq u'$.*

Proof. The proof is exactly the same as the proofs of Lemmas 190 and 196. □

Definition 207. *We define a mapping from B_G to $\mathbb{N}_\perp$ by $\mathfrak{s}_{sw}(B) = s$ iff $\sigma(B)$ (considering the highest occurrence of $\sigma(B)$ in the proof derivation) is labelled by a formula of the shape $!_{s,\_}$. We also define a partial mapping $\mathfrak{s}_{sw}(\_)$ on contexts by: $\mathfrak{s}_{sw}(((e,P),[!_t]@T)) = s$ iff $\boldsymbol{\beta}((e,P),[!_t]@T) = !_{s,\_}$ (considering the lowest e).*

Lemma 208. *If $((e,Q),[!_{t_1}]@T)$ is used and $\beta(e)_{|T} = !_{\_,\_,u}A$ then $u = \perp$.*

Proof. By definition of used contexts (Definition 105), there exists a path of the shape $((\sigma(B),P),[!_t]) \mapsto^* ((e,Q'),[!_{t'_1}]@T')$ with $t' \neq t'_1$ and $((e,Q),[!_{t_1}]@T)^\varnothing = ((e,Q'),[!_{t'_1}]@T')^\varnothing$. Because $t' \neq t'_1$, there exists a context in the path from $((\sigma(B),P),[!_t])$ to $((e,Q'),[!_{t'_1}]@T')$ of the shape $((\overline{f},R),[!_v])$ (with f the premise of a $?C$ or $?N$ node).

By definition of $\_,\beta(e)_{|T'}$ is of the shape $!_{\_,\_,u'}A'$. By definition of $\boldsymbol{\beta}(\_)$, there exists a substitution θ such that $\boldsymbol{\beta}((e,Q'),[!_{t'_1}]@T') = !_{u'}A'[\theta]$. By Lemma 206, $\boldsymbol{\beta}((\overline{f},R),[!_v])$ is of the shape $!_u B$ with $u \geq u'$. Because f is the premise of a $?C$ or $?N$ node, we have $u = \perp$ so $u' = \perp$. □

Lemma 209. *If $\beta(e) \in \mathcal{F}_{s,\Diamond}$ (resp. $\mathcal{F}_{s,\blacklozenge}$, $\mathcal{F}_{s,\top}$), then $\beta(\beta(e)[\theta(e,P)],e,P,V@T,[]) \in \mathcal{F}_{s,\Diamond}$ (resp. $\mathcal{F}_{s,\blacklozenge}$, $\mathcal{F}_{s,\top}$).*

Proof. The proof is similar to the proof of Lemma 202. The important point to notice is that, if $A \in \mathcal{F}_{s,\Diamond}$ and $X_{t,w}$ or $X^{\perp}_{t,w}$ appears in A then $t \geq s$. If this variable is replaced by a formula B in a $\exists$ node, then $B \in \mathcal{F}_{t,\Diamond} \subseteq \mathcal{F}_{s,\Diamond}$.

If we also suppose that $A \in \mathcal{F}_{s,\blacklozenge}$ then for every variable $X_{t,w}$ or $X^{\perp}_{t,w}$ appearing in A we have $w \in \{\blacklozenge, \top\}$. Thus, this variable can only be replaced by formulae of the shape $B_\blacklozenge$ with $B \in \mathcal{F}_{t,\Diamond}$. Let us notice that in this case, $B_\blacklozenge \in \mathcal{F}_{t,\blacklozenge}$. □

Lemma 210. *Let us suppose that* $((e, P.p), T) \rightsquigarrow^* C', \boldsymbol{\beta}((e, P.p), T) = !_{\_w,\_}A$ *with* $w \in \{\Diamond, \blacklozenge\}$*, B is the deepest box containing e and the path contains no contex of the shape* $((\sigma(B), P), U.!_p)$*.*

Then C' *is of the shape* $((e', P.p), T')$ *with* $e' \in B$ *and* $\boldsymbol{\beta}(C')$ *is of the shape* $!_{\_w',\_}A'$ *with* $w \geq w'$*.*

Proof. We can observe that it is enough to prove the result when $C = ((e, P.p), T) \rightsquigarrow C'$ (the length of the path is 1), then the general result is obtained by induction on the length of the path.

By supposition, $\boldsymbol{\beta}((e, P.p), T) \notin \mathcal{F}_{\bot,\top}$ and $\boldsymbol{\beta}((\overline{e}, P.p), \overline{T}) \notin \mathcal{F}_{\bot,\top}$. By Lemma 209, $\beta(e) \notin \mathcal{F}_{\bot,\top} \cup \overline{\mathcal{F}_{\bot,\top}}$. So $\overline{e}$ is not the conclusion of a $?P$ node or $!P$ node: the $C \rightsquigarrow C'$ step does not enter a box. And e is not the premise of a $?P$ node: the $C \rightsquigarrow C'$ step does not leave the box by an auxiliary door. By supposition, the $C \rightsquigarrow C'$ step does not leave B by its principal door.

This proves that C' is of the shape $((e', P.p), T')$. Moreover, the only $\mapsto$ step which increase the second label are the $\hookrightarrow$ step (this is not the case here because $C \rightsquigarrow C'$) and the step leaving a box by its principal door (this is not the case here by assumption). The other steps modifying the second label are: the subtyping rule (which can only decrease the label, similarly to Lemmas 206, 190 and 196), crossing a $?C$ or $?N$ node upwards (we go from a label $w_1 + w_2$ to w_1 so it decreases the label). □

5.3.5 Combining the previous systems into *SwLL*

In this section, we want to prove that, for every proof-net G belonging to $SwLL_{dc}$, $SwLL_{nest}$, $SwLL_{last}$ and $SwLL_{strat}$, there exists a weak stratification. In fact there is one property which we fail to prove: no criterion prevents sequences of the shape $(C, Q, u) \boxdot\!\!\mapsto^+ (C, Q', u')$ with $s = \mathfrak{s}(C)$ and $(C, Q)^{s-1} = (C, Q')^{s-1}$. Thus, we may have $(C, Q, u) \smile^0 (C, Q, u)$ and $(C, Q, u') \smile^0 (C, Q, u') \boxdot\!\!\mapsto^+ (C, Q', u'')$, $s = \mathfrak{s}(C)$ and $(C, Q')^{s-1} = (C, Q'')^{s-1}$ which contradicts the last condition of weak stratifications.

We think that such a property could be enforced by criteria close to *QPLL* (Section 5.2). However this part is still a work in progress. This is why, in the meantime, we set $S_{\boxdot\mapsto} = \varnothing$. Even with this constraint, *SwLL* is still expressive enough to embed *SDNLL* (so *LLL*, L^4 and MS_{max}), the *Poly*-sound systems of *MS*, and *SLL*.

Definition 211. *The proof-nets of SwLL are the proof-nets G such that there exist decorations of G:* G_{dc} *in* $SwLL_{dc}$*,* G_{nest} *in* $SwLL_{nest}$*,* G_{last} *in* $SwLL_{last}$ *and* G_{strat} *in* $SwLL_{strat}$*. And those decorations use the same subsets S of* B_G*, and subsets* $E_\circ$ *and* $E_{\boxdot\mapsto}$ *of* E_G*. The set* $S_{\boxdot\mapsto}$ *used by those systems is* $\varnothing$*.*

The sets S, $S_{\boxdot\mapsto}$, $E_\circ$ and $E_{\boxdot\mapsto}$ are only used to communicate between the 4 systems used. We could define *SwLL* as a single system. The ! and ? modalities of this system would be labelled by tuples (s, d, n, c, l, u, w) with $s \in \mathbb{N}_\bot$, $d \in \mathbb{N}_\bot$, $n \in \mathbb{N}_\bot^\top$, $c \in \{\bot, \top\}$, $l \in \{\bot, \top\}$, $u \in \{\bot, \top\}$ and $w \in \{\blacklozenge, \Diamond, \top\}$. The variables of this system would be labelled by tuples (s, l, w) with $s \in \mathbb{N}_\bot$, $l \in \{\bot, \top\}$, and $w \in \{\blacklozenge, \Diamond, \top\}$. We did not present the system this way because its rules are quite complex. Separating *SwLL* into four systems allowed us to prove step by step the properties enforced by *SwLL*.

Let us consider a *SwLL* proof-net G whose only conclusion is e. We suppose that, in G_{dc}, e is labelled by $A_{dc} \multimap B_{dc}$ and no $\bot$ appears in A_{dc}. We also suppose that, in G_{last}, e is labelled by $A_{last} \multimap B_{last}$ and no $\bot$ appears in B_{dc}. We consider a *SwLL* proof-net H in normal form with only one conclusion. Let us suppose that $(G)H$ is a *SwLL* proof-net (i.e. the formulae match in $SwLL_{dc}$, $SwLL_{nest}$, $SwLL_{last}$ and $SwLL_{strat}$). Let us notice that no $\bot$ appears in the conclusions of $(G_{last})H_{last}$. We define a weak stratification candidate of $(G)H$ as follows:

- If $B \in S$, we set $\mathfrak{s}(B) = 0$, else we set $\mathfrak{s}(B) = 1 + \mathfrak{s}_{sw}(B)$.
- If $(B, \_, \_) \boxdot\!\!\mapsto^* (B', \_, \_)$, and $B' \subset C' \notin S$ we set $\mathfrak{d}(B) = 1 + \mathfrak{d}_{sw}(B)$. Else we set $\mathfrak{d}(B) = 1$.

- For every box B, we set $(B,P,t) \boxdot\!\!\rightarrow (B',P',t')$ if and only if there exists a path of the shape $((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(B')},P'),T'.?_{t'}) \rightsquigarrow C \rightsquigarrow^* C_{B,P,t}$ (with $C_{B,P,t}$ the context of Lemma 203) and $\boldsymbol{\beta}(C)$ (in $(G_{strat})H_{strat}$) is of the shape $!_{\_,\Diamond,\_}$.

Lemma 212. *For every $B \in B_G - S$, we have $\mathfrak{s}(B) \geq 1$.*

Proof. By definition of $SwLL_{strat}$, $\mathfrak{s}_{sw}(B) \in \mathbb{N}$. So we defined $\mathfrak{s}(B)$ as $\mathfrak{s}_{sw}(B) + 1 \geq 1$. □

Lemma 213. *If $(B,P,t) \smile^* (B',P',t')$ and there is a box of $B_G - S$ containing B', $\mathfrak{d}(B) \geq 1$.*

Proof. By Lemma 192, $\mathfrak{d}_{sw}(B) \in \mathbb{N}$. So, we defined $\mathfrak{d}(B)$ as $\mathfrak{d}_{sw}(B) + 1 \geq 1$. □

Lemma 214. *Let us suppose that $((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(D)},R),V.?_v) \rightsquigarrow^* C_{B,P,t}$, the second part of the path does not leave (D,R,v) and $((\overline{\sigma(D)},R),V.?_v)$ is a used context.*

Then, we have $\mathfrak{s}(B) \geq \mathfrak{s}(D)$. Moreover, if $\mathfrak{s}(B) = \mathfrak{s}(D)$, $(B,P,t) \boxdot\!\!\rightarrow (D,R,v)$.

Proof. Let us consider the context C such that $((\overline{\sigma(D)},R),V.?_v) \rightsquigarrow ((e,R.v),V)$. Let us set $!_{s,w,u}A = \boldsymbol{\beta}((e,R.v),V)$ (in $(G_{strat})H_{strat}$). By Lemma 206, we have $\mathfrak{s}_{sw}(B) \geq s$. Let $!_{s',w',u'}A' = \beta(\sigma(D))$. Observing the promotion rule of $SwLL_{strat}$, either $A' \in \mathcal{F}_{s',\blacklozenge}$ or ($D \in S_{\boxdot\rightarrow}$ and $A' \in \mathcal{F}_{s',\Diamond}$). By definition of $\boldsymbol{\beta}(\_)$, $!_{s,w,u}A$ is a subformula of $A'' = \beta(\beta(e)[\theta(e,R.v)],e,R.v,V,[])$. By Lemma 209, $A'' \in \mathcal{F}_{s',\Diamond}$ and either $A'' \in \mathcal{F}_{s',\blacklozenge}$ or $D \in S_{\boxdot\rightarrow}$. Thus, we are in one of the following cases:

- Either $u = \top$. In this case, by Lemma 208, $((\overline{\sigma(D)},R),V.?_v)$ is not used. Which contradicts our assumption.
- Or $w = \blacklozenge$. In this case, by Lemma 210, $\boldsymbol{\beta}(C_{B,P,t})$ would be of the shape $!_{\_,\blacklozenge,\_}$. So there exists an edge f of $E_{\boxdot\rightarrow}$ with $\beta(f) = ?_{\_,\blacklozenge,\_}$. This is in contradiction with the definition of $SwLL_{strat}$.
- Or $s \geq s' + 1$. In this case, $\mathfrak{s}(B) > \mathfrak{s}(D)$.
- Or $s = s'$ (so $\mathfrak{s}(B) \geq \mathfrak{s}(D)$), $w = \Diamond$ and $D \in S_{\boxdot\rightarrow}$. By definition, we have $(B,P,t) \boxdot\!\!\rightarrow (D,R,v)$.

□

Lemma 215. *Let $(B,P,t),(B',P',t') \in CanCop_G$. If $(B,P,t) \boxdot\!\!\rightarrow (D,R,v)$, $(B',P',t') \boxdot\!\!\rightarrow (D,R',v')$ then there exist paths of the shape:*

$$((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(D)},R),V.?_v) \quad \rightsquigarrow ((e_1,P_1),[!_{t_1}]@T_1)\cdots \quad \rightsquigarrow ((e_k,P_k),[!_{t_k}]@T_k) = C_{B,P,t}$$
$$((\sigma(B'),P'),[!_{t'}]) \rightsquigarrow^* ((\overline{\sigma(D)},R'),V'.?_{v'}) \quad \rightsquigarrow ((e'_1,P'_1),[!_{t'_1}]@T'_1)\cdots \quad \rightsquigarrow ((e'_k,P'_k),[!_{t'_k}]@T'_k) = C_{B',P',t'}$$

For every $1 \leq i \leq k$, we have $e_i = e'_i$ and the paths do not enter boxes. Moreover, if e_k is of the shape $\overline{\sigma\_(D)}$, then $\mathfrak{d}(B) > \mathfrak{d}(D)$.

Proof. By definition of $\boxdot\!\!\rightarrow$, $\boldsymbol{\beta}((\overline{\sigma(D)},R),V.?_v) = !_{\_,\Diamond,\_}A$ and $\boldsymbol{\beta}((\overline{\sigma(D)},R'),V'.?_{v'}) = !_{\_,\Diamond,\_}A'$. Thus, because the paths to $C_{B,P,t}$ and $C_{B',P',t'}$ do not leave (D,R,v) and (D,R',v'), and $\boldsymbol{\beta}(C_{B,P,t})$ and $\boldsymbol{\beta}(C_{B',P',t'})$ can not be of the shape $!_{\_,\blacklozenge,\_}$, for every context C in one of the paths we have $\boldsymbol{\beta}(C) = !_{\_,\Diamond,\_}$.

Let us observe that $e_i = e'_i$ (they are the premise of the principal door of D) and, because the formulae of $\mathcal{F}_{\bot,\Diamond}$ have at most one $\Diamond$ label, T_1 and T'_1 "point" to the same connective of $\beta(e_1)$: either $\beta(e_1)_{T_1} = \beta(e_1)_{T'_1} = !_{\_,\Diamond,\_}$ and T_1 only differs from T'_1 on exponential trace elements, or there exist prefixes U_1 and U'_1 of T_1 and T'_1 such that $\beta(e_1)_{U_1} = \beta(e_1)_{U'_1} = X_{\_,\Diamond}$ (it can also be equal to $X^{\bot}_{\_,\Diamond}$ but this case is treated in the same way) and U_1 only differs from U'_1 on exponential trace elements.

We can prove by induction on i that this property is preserved for $((e_i,P_i),[!_{t_i}]@T_i)$ and $((e_i,P_i),[!_{t_i}]@T_i)$. The interesting cases are the following:

- When $\overline{e_i}$ is the conclusion of a $?C$ node. Then, there are two choices for e_{i+1}, however there is only one choice which preserves $\Diamond$ as the second label (indeed $\Diamond + \Diamond$ is undefined). So $e_{i+1} = e'_{i+1}$.
- When $\overline{e_i}$ is the conclusion of a $?N$ node. Then, there are two choices for $|T_{i+1}|$, however there is only one choice which preserves $\Diamond$ as the second label (indeed $\Diamond + \Diamond$ is undefined). So $|T_{i+1}| = |T'_{i+1}|$.
- When $\overline{e_i}$ is the conclusion of an $\exists$ node whose associated variable is X and whose associated formula is A''. Let us suppose that there exist prefixes U_1 and U'_1 of T_1 and T'_1 such that $\beta(e_1)_{U_1} = \beta(e_1)_{U'_1} = X_{\_,\Diamond}$. Because A'' must be in $\mathcal{F}_{\perp,\Diamond} \cup \overline{\mathcal{F}_{\perp,\Diamond}}$, there is only one possibility to extend U_i and U'_i to traces pointing to a connective carrying a $\Diamond$ label.

□

Lemma 216. *If* $\mathfrak{d}(B) \geq 1$*, and* $(B, P, t) \smile (C, Q, u)$ *then* $\mathfrak{d}(B) \geq \mathfrak{d}(C)$.

Proof. Immediate from Lemma 190. □

Lemma 217. *If* $\mathfrak{d}(B) \geq 1$*, and* $(B, P, t) \boxdot\!\!\rightarrow (C, Q, u)$ *then* $\mathfrak{d}(B) \geq \mathfrak{d}(C)$.

Proof. By Definition of the weak stratification candidate and the proof of Lemma 210. □

Lemma 218. *For* $s \in \mathbb{N}$*, if* $((\sigma(B), P), [!_t]) \mapsto^*_{S_s} ((\overline{\sigma_i(C)}, Q), [!_u])$ *and* $((\sigma(B), P), [!_{t'}]) \mapsto^*_{S_s} ((\overline{\sigma_{i'}(C)}, Q'), [!_{u'}])$ *with* $(C, Q)^{\mapsto_{S_s}} = (C, Q')^{\mapsto_{S_s}}$ *then either* $i = i'$ *or* $\mathfrak{d}(B) > \mathfrak{d}(C)$.

Proof. It is a weak version of Lemma 199. □

Lemma 219. *Let us suppose that* $(B, P, t) \smile^* (C, Q, u)$ *and* $(B, P, t') \smile^* \boxdot\!\!\rightarrow^+ (C, Q', u')$*. Then,* $\mathfrak{d}(B) > \mathfrak{d}(C)$.

Proof. Because we supposed $S_{\boxdot\rightarrow} = \emptyset$, $\boxdot\!\!\rightarrow = \emptyset$ so such a situation never happens. □

Lemma 220. *The tuple* $(S, \boxdot\!\!\rightarrow, \mathfrak{s}(\_), \mathfrak{d}(\_))$ *defined above is a weak stratification.*

Proof. Let us examine the definition of weak stratification:

- By Lemma 212, "For every $B \in S$, we have $\mathfrak{s}(B) = 0$. And for every $B \in B_G - S$, we have $\mathfrak{s}(B) \geq 1$."
- By Lemma 203, "$((\sigma(B), P), [!_t]) \mapsto^* C_{B,P,t}$. For every $((\sigma(D), R), [!_v])$ in the path we have $\mathfrak{d}(B) = \mathfrak{d}(D)$ and $\mathfrak{s}(B) \geq \mathfrak{s}(D)$. If $\overline{e}$ is not an auxiliary edge then $u = \mathtt{e}$ and there is no path of the shape $C_{B,P,t} \rightsquigarrow^+ ((\overline{\sigma(\_)}, \_), \_)$. If $\boxdot\!\!\rightarrow (B, P, t)$, then the path $((\sigma(B), P), [!_t]) \mapsto^* C_{B,P,t}$ is a $\rightsquigarrow$-path."
- By Lemma 214, we have "Let us suppose that $((\sigma(B), P), [!_t]) \mapsto^* ((\overline{\sigma(D)}, R), V.?_v) \mapsto^* C_{B,P,t}$ and the second part of the path does not leave (D, R, v) then we are in one of the following situations:
  - $\mathfrak{s}(B) > \mathfrak{s}(D)$
  - $\mathfrak{s}(B) = \mathfrak{s}(D)$, $(B, P, t) \boxdot\!\!\rightarrow (D, R, v)$ and the first part of the path is a $\rightsquigarrow$-path (it is to say that $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(D)}, R), V.?_v)$).
  - $((\overline{\sigma(D)}, R), V.?_v)$ is not used and $\not\boxdot\!\!\rightarrow (B, P, t)$.

  "

- By Lemma 215, we satisfy the condition "Let $(B,P,t),(B',P',t') \in CanCop_G$. If $(B,P,t) \boxdot\!\!\rightarrow (D,R,v)$, $(B',P',t') \boxdot\!\!\rightarrow (D,R',v')$ then there exist paths of the shape (notice that the edges are the same in the second part of the path):

$$\begin{array}{lll} ((\sigma(B),P),[!_t]) \rightsquigarrow^* ((\overline{\sigma(D)},R),V.?_v) & \rightsquigarrow ((e_1,\_),\_)\cdots & \rightsquigarrow ((e_k,\_),\_) = C_{B,P,t} \\ ((\sigma(B'),P'),[!_{t'}]) \rightsquigarrow^* ((\overline{\sigma(D)},R'),V'.?_{v'}) & \rightsquigarrow ((e_1,\_),\_)\cdots & \rightsquigarrow ((e_k,\_),\_) = C_{B',P',t'} \end{array}$$

  and the paths do not enter boxes. Moreover, if $\overline{e_k}$ is an auxiliary edge of a box C, we have $\mathfrak{d}(B) > \mathfrak{d}(C)$ and $\mathfrak{s}(B) \leq \mathfrak{s}(C)$."

- By Lemma 213, we have "Let us consider a box $B \in B_G$. Either $\mathfrak{d}(B) = 0$ and $((\sigma(B),P),[!_t]) \mapsto^* ((e,Q),[!_u])$ implies that every box containing C is in S. Or $\mathfrak{d}(B) \geq 1$".

- By Lemmas 216 and 217 we satisfy "Let us suppose that $\mathfrak{d}(B) \geq 1$. If $(B,P,t) \smile (C,Q,u)$ or $(B,P,t) \boxdot\!\!\rightarrow (C,Q,u)$ then $\mathfrak{d}(B) \geq \mathfrak{d}(C)$."

- By Lemma 218 we have "For $s \in \mathbb{N}$, if $((\sigma(B),P),[!_t]) \mapsto^*_{S_s} ((\overline{\sigma_i(C)},Q),[!_u])$ and $((\sigma(B),P),[!_{t'}]) \mapsto^*_{S_s} ((\overline{\sigma_{i'}(C)},Q'),[!_{u'}])$ with $(C,Q)^{\mapsto s_s} = (C,Q')^{\mapsto s_s}$ then either $i = i'$ or $\mathfrak{d}(B) > \mathfrak{d}(C)$."

- By Lemma 219, we satisfy the last condition: "Let us suppose that we have $(B,P,t) \smile^* (C,Q,u)$, $(B,P,t') \smile^* (D,R,v) \boxdot\!\!\rightarrow^+ (C,Q',u')$ and $s = \mathfrak{s}(D)$ then we have $(C,Q)^{\mapsto s_{s-1}} \neq (C,Q')^{\mapsto s_{s-1}}$."

□

Theorem 221. *S wLL is sound for Poly.*

Proof. By Lemmas 220, 198 and 199, we can use Lemma 119. □

Let us sketch the encoding of *S DNLL* in *S wLL*. For every proof-net G of *S DNLL* we define a proof-net G' of *S DNLL* as follows. We set $S = \emptyset$, and the judgements in $S\,wLL_{dc}$ and $S\,wLL_{nest}$ are always of the shape $\vdash_\perp$. Thus, we will not use the indices $\perp$ in $S\,wLL_{dc}$ and $S\,wLL_{nest}$. For every modality $!_{s,d,n}$ of *S DNLL*, we consider the labels $!_{s,\top,\perp\_}$ in $S\,wLL_{strat}$, $!_{2d+1}$ in $S\,wLL_{dc}$, $S\,wLL_{n,\top}$ in $S\,wLL_{nest}$. Finally, we set $E_\circ$ as the set of every edge, so in $S\,wLL_{last}$ we can consider the labels $!_\top$.

Encoding the systems of *MS* is straightforward: we can replace every $!_d$ by $!_{2d}$. in $S\,wLL_{dc}$. The labels in the other systems are defined as in the encoding of *S DNLL*.

To encode *S LL*, we can notice that we may label every ? (except on the conclusion of $?D$ nodes) by $?_{\multimap\multimap\top}$, thus all the other conditions of $S\,wLL_{strat}$ become trivial. Because there is no $?N$ node, we can set all the labels to 0 in $S\,wLL_{nest}$. In $S\,wLL_{dc}$, we can consider the label 2 on every connective, except on the premises of $?C$ nodes where we consider the label 1. Finally, we set $E_\circ$ as the set of every edge, so in $S\,wLL_{last}$ we can label every connective by $\top$.

Chapter 6

Interaction nets

In the previous sections we push the boundaries of intensional expressivity for subsystems of linear logic characterizing polynomial time. There is still room for improvement, but we think that the most promising direction to increase the intensional expressivity of our systems is to extend linear logic with more primitives: built-in arithmetic, inductive types, pattern matching, less constrained recursion, side-effects, multi-threading, exceptions,... As a simple limitation of linear logic we observed, in $SDNLL_\lambda$ we did not allow weakening on linear variables (if x does not appear free in t, the type of $\lambda x.t$ must be of the shape $!A \multimap B$). Indeed, if we allowed such linear weakening, there would be no canonical encoding of typing derivations into proof-nets. Because the set of features we want in our language is not fixed, a general framework of systems would be preferred to a single system. This way, we would need to define the context semantics and prove the general theorems only once, and they will stand for any system of the framework.

The framework we chose is interaction nets: a well-behaved class of graph rewriting systems [45]. Interaction nets are a model of asynchronous deterministic computation. A net is a graph-like structure whose nodes are called *cells*. Each cell is labelled by a symbol. A library defines the set of symbols and the rewriting rules for the symbols. Thus, a library corresponds to a programming language. Interaction nets as a whole, correspond to a set of programming languages. Interaction nets present several major advantages:

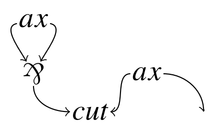
- The definition of interaction nets was inspired by the proof-nets of linear logic. Because they have many similarities, it may be relatively easy to transpose the methods of the present thesis to interaction nets (compared to other equally expressive frameworks).
- Interaction nets have been used to encode several systems. Proof-nets [53] and λ-calculus [51] but also functional programming languages containing pattern-matching and built-in recursion [30]. A non-deterministic extension is powerful enough to encode the full π-calculus [55]. This could be used to control ressources of processes as in the SHO_π calculus [48].

Because context semantics is our main tool, our first step will be to extend context semantics to every library of interaction nets. As context semantics is a model of geometry of interaction, the most relevant work is the definition of a geometry of interaction for an arbitrary library by De Falco [27]. De Falco defines a notion of paths in nets and a notion of reduction of those paths. Then, he defines a geometry of interaction of a library as a weighing of paths by elements of a semi-group such that the weights are stable along reduction. However, he exhibits such a semi-group only for some particular libraries (based on linear logic). Thus, there is no complete geometry of interaction model of interaction nets yet.

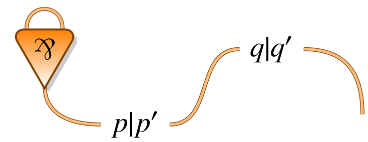
The most difficult part to define a context semantics on interaction net was to define a notion of tokens. Then, defining the rules of the paths, proving that these paths are stable along reduction, defining a weight

$W_G \in \mathbb{N} \cup \{\infty\}$, and proving that it decreases along reduction is relatively easy and natural. For instance, the paths of context semantics for interaction nets are governed by only 5 rules, compared to the 29 of context semantics for linear logic. However, it seems harder to use it than the linear logic context semantics. We think we still need to define additional tools.

We study another application for this context semantics: a denotational semantics for a large class of interaction net systems. We define a notion of observational equivalence for each library. Then we define a denotational semantics which is, on a class of libraries named *crossing libraries*, sound and fully abstract with respect to our equivalence. We previously presented those works in [61].



(a) Proof-net G



(b) Interaction net corresponding to G

Figure 6.1: Intuition behind the merging ports

6.1 Definition of interaction nets

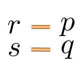
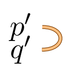
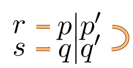
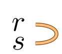
6.1.1 Statics

Interaction nets have been defined in many ways. Here, to define properly the paths of context semantics, we had to use a formal definition. Our presentation is inspired by De Falco's [26].

Definition 222. *We fix a* symbol set $S = (\mathcal{S}, \alpha)$ *with* $\mathcal{S}$ *a countable set whose elements will be called* symbols *and* α *a mapping from* $\mathcal{S}$ *to* $\mathbb{N}$ *associating an* arity *to each symbol.*

For instance in the case of linear logic, we set:

$$\begin{aligned} S_{LL} &= \{⅋, \otimes, \forall, \exists, ?C, ?D, ?N, ?W\} \cup \{Box_G \mid G \text{ is a proof-net}\} \\ \alpha_{LL} &= \{⅋ \mapsto 2, \otimes \mapsto 2, \forall \mapsto 1, \exists \mapsto 1, ?C \mapsto 2, ?D \mapsto 1, ?N \mapsto 1, ?W \mapsto 0, \\ &\qquad Box_G \mapsto \text{number of conclusions of } G \text{ minus } 1\} \end{aligned}$$

A net is a set of cells joined by wires. Wires may have one (or both) ends unattached. We will often connect nets, those connections are made by those unattached ends. Formally, the ends of wires will be represented by a set P^N of *ports*. There are three types of ports: ports attached to a cell (the set P^N_c), *free ports* (the set P^N_f) and *merging ports* (the set P^N_m). This latest group is used for technical reasons, when connecting the nets  and  the result will be  , where the ports p, p', q and q' are merging ports. This net will be considered equivalent to  . The merging ports correspond to *ax* and *cut* nodes. For instance, the proof-net of Figure 6.1a corresponds to the interaction net of Figure 6.1b.

Definition 223. *A* net N *is a tuple* $(P^N, C^N, l^N, \sigma^N_w, \sigma^N_m, \sigma^N_c)$ *with:*

- $P^N = P^N_c \uplus P^N_f \uplus P^N_m$ *is a finite set called set of* ports.
- C^N *is a finite set whose elements will be called* cells
- $l^N : C^N \mapsto \mathcal{S}$ *labels each cell with a symbol.*
- σ^N_w *is an involution on* P^N *with no fixpoint. We also write* $\overline{p}$ *for* $\sigma^N_w(p)$. σ^N_w *represents the wires: if there is a wire between the ports* p *and* p'*, then* $\overline{p} = p'$ *and* $\overline{p'} = p$.
- σ^N_m *is an involution on* P^N_m *with no fixpoint. This mapping associates two merging ports.*
- σ^N_c *is a bijection from* P^N_c *to* $\{(c, i) \mid c \in C^N, 0 \leq i \leq \alpha(l^N(c))\}$. σ^N_c *represents the cells. For instance,* $\sigma^N_c(p) = (c, 2)$ *if* p *is the second auxiliary port of* c *and* $\sigma^N_c(p) = (c, 0)$ *if* p *is the principal port of* c.

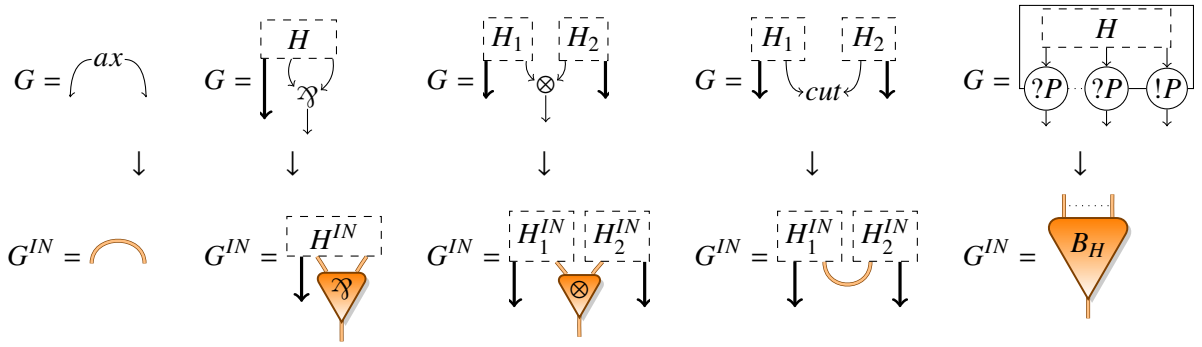


Figure 6.2: A simple encoding of linear logic in interaction nets.

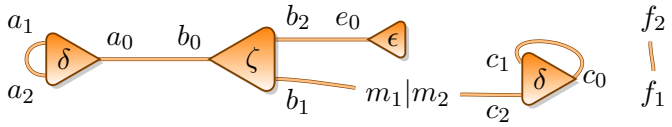


Figure 6.3: Net N. Names of ports and labels of cells are represented while names of cells are not.

For example, let $S_{comb} = \{\zeta, \delta, \epsilon\}$ be symbols with $\alpha(\zeta) = \alpha(\delta) = 2$ and $\alpha(\epsilon) = 0$. Then, Figure 6.3 represents the net N with:

- $P^N = \{a_0, a_1, a_2, b_0, b_1, b_2, c_0, c_1, c_2, e_0, f_1, f_2, m_1, m_2\}$, $C^N = \{A, B, C, E\}$
- $l^N = \{A \mapsto \delta, B \mapsto \zeta, C \mapsto \delta, E \mapsto \epsilon\}$
- $\sigma_w^N = \{a_1 \leftrightarrow a_2, a_0 \leftrightarrow b_0, b_2 \leftrightarrow e_0, b_1 \leftrightarrow m_1, m_2 \leftrightarrow c_2, c_1 \leftrightarrow c_0, f_1 \leftrightarrow f_2\}$ and $\sigma_m^M = \{m_1 \leftrightarrow m_2\}$
- $\sigma_c^N = \{a_0 \mapsto (A, 0), a_1 \mapsto (A, 1), a_2 \mapsto (A, 2), b_0 \mapsto (B, 0), b_1 \mapsto (B, 1), b_2 \mapsto (B, 2), c_0 \mapsto (C, 0), c_1 \mapsto (C, 1), c_2 \mapsto (C, 2), e_0 \mapsto (E, 0)\}$.

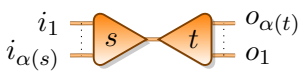
For every proof-net G, we define an encoding G^{IN} of G in nets (with the symbol set S_{LL}). The main cases of the encoding are presented in Figure 6.2.

The merging ports are introduced for technical reasons but are not essential. Let p, q be merging ports of a net N such that $p \neq \overline{q}$. Let N' be the net equal to N where $\overline{p} - p|q - \overline{q}$ is replaced by $\overline{p} = \overline{q}$, then we write $N \rightarrow_m N'$. We define the equivalence relation $\leftrightarrows_m$ as the reflexive symmetric transitive closure of $\rightarrow_m$. The nets will be considered up to $\leftrightarrows_m$ equivalence and α-equivalence (renaming of the ports and cells). Notice that $\rightarrow_m$ is confluent and strongly normalizing, we will usually represent a net by its $\rightarrow_m$ normal form (the only merging ports are the cycles of shape $p|q$).

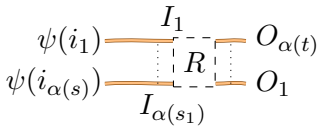
Let c be a cell of N. We write $p^i(c)$ the port p such that $\sigma_c^N(p) = (c, i)$. The *principal port* of c denotes $p^0(c)$. If $i \geq 1$, $p^i(c)$ is called the *i-th auxiliary port* of c.

6.1.2 Dynamics

The interaction between two nets is done by merging some of their free ports. This operation is called *gluing* and will be the main tool to define the dynamics of nets.



(a) The net $\mathfrak{R}_{s,t}$



(b) The reduct $N_{s,t}$ of s/t

Figure 6.4: Interaction rule with explicit bijection ($O_k = \psi(o_k)$).

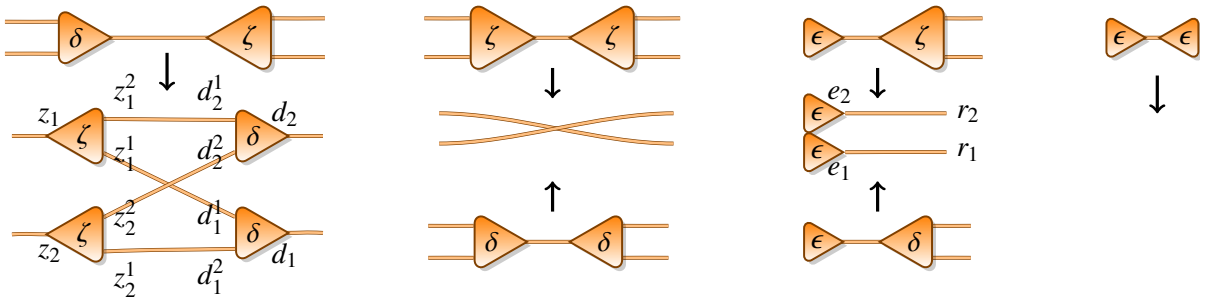


Figure 6.5: The symmetric combinators, library L_{comb} for S_{comb}

Definition 224. *Let M and N be nets and ϕ be a partial injection from P_f^M to P_f^N, then $M \bowtie_\phi N$ is the net whose ports and cells are those of M and N, the free ports in the domain and codomain of ϕ become merging nodes with $\sigma_m^{M\bowtie_\phi N}(p) = \phi(p)$ and $\sigma_m^{M\bowtie_\phi N}(\phi(p)) = p$.*

As an example, if we set $M =$  , $N =$  $f_1 = f_2$ and $\phi = \{m_1 \mapsto m_2\}$, then $M \bowtie_\phi N$ is the net of Figure 6.3.

The computation in interaction nets is done by reduction of *active pairs*. An active pair is a set of two cells linked by their principal ports. *Libraries* will define which pairs of symbols can interact. When an active pair is labelled by symbols which can interact together, we may reduce it: those cells are replaced by a net $N_{s,t}$ which only depends on the symbols of the active pair. The rest of the interaction net is left untouched.

Definition 225. *Let $s, t \in \mathcal{S}$, $\mathfrak{R}_{s,t}$ is the net of Figure 6.4a.*

An interaction rule for (s,t) is a tuple (R,ψ) where R is a net and ψ is a bijection from $P_f^{\mathfrak{R}_{s,t}}$ to P_f^R. For $1 \le j \le \alpha(s)$, we name I_j the edge $\overline{\psi(i_j)}$ of R. For $1 \le j \le \alpha(t)$, we name O_j the edge $\psi(o_j)$ of R, as in Figure 6.4b.

In practice, we will describe interaction rules by displaying an active pair and the reduct linked by an arrrow as in Figure 6.5. The bijection is given implicitly by the position of the ports.

Definition 226 (library)**.** *A* library *for the symbol set $(\mathcal{S}, \alpha)$ is a partial mapping L on $\mathcal{S}\times\mathcal{S}$. To each (s_1, s_2) in the domain of L, L associates an interaction rule for (s_1, s_2).*

Let us suppose that $L(s_1, s_2) = (R, \psi)$. Then we require that $L(s_2, s_1)$ is defined and equal to the symmetric *of $L(s_1, s_2)$ where inputs and outputs are switched, i.e. $L(s_2, s_1) = (R, \psi \circ \{i_k \leftrightarrow o_k\})$.*

Once we fixed a library, the reduction $\to$ is defined by $N \bowtie_\phi \mathfrak{R}_{s_1,s_2} \to (N \bowtie_{\psi\circ\phi} R)$.

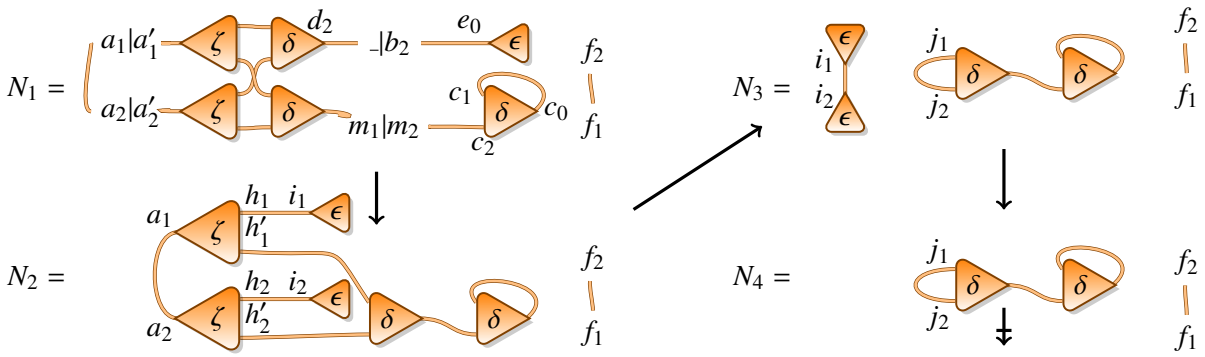


Figure 6.6: Example of reduction with the library L_{comb}.

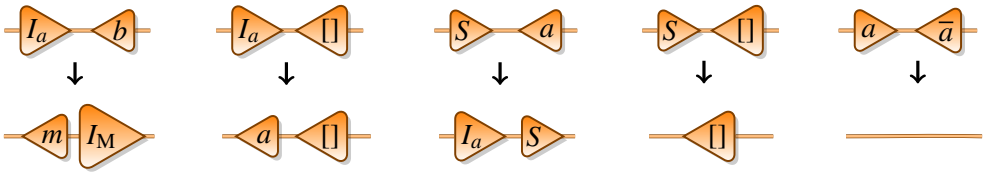
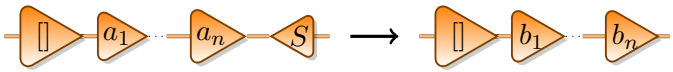


Figure 6.7: L_{sort} library. The rules stand for any $a, b \in A$, $m = \min(a, b)$ and $M = \max(a, b)$

Because of the symmetry condition, the rules shown in Figure 6.5 are enough to describe the whole library L_{comb} of symmetric combinators. The net of Figure 6.3 successively reduces to the nets of Figure 6.6.

As another example, let us consider an ordered set $(A, \leq)$, and the symbols $\{S, []\} \cup A \cup \{I_a \mid a \in A\} \cup \{\overline{a} \mid a \in A\}$. The arities and the library L_{sort} are defined by Figure 6.7. Then,



with $[b_1; \cdots; b_n]$ the sorted list corresponding to $[a_1; \cdots; a_n]$. More precisely, it is an implementation of insertion sort.

Finally, in Figure 6.8, we define a library for the encoding of proof-nets we defined in Figure 6.2. We can notice that this does not correspond exactly to the *cut*-elimination of linear logic because: we lack the $!P/?P$ rule, and we can not reduce inside a box. However, the only purpose of this encoding is to make a comparison with the linear logic case to guide intuition. So this slight mismatch will have little importance.

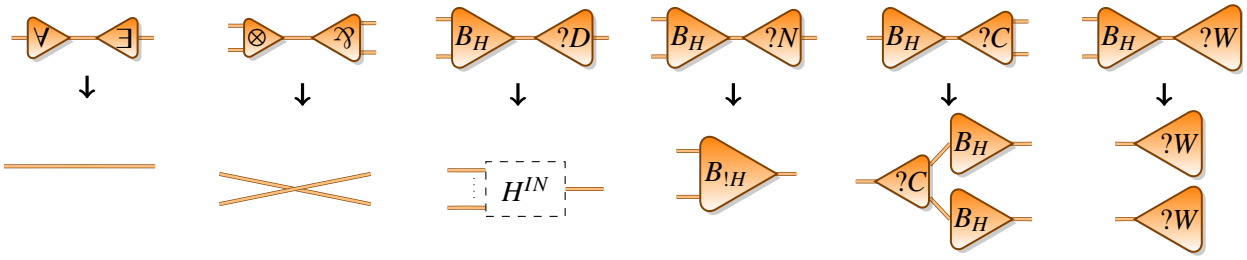


Figure 6.8: Library for the encoding of *LL* in interaction net. For the $?C$ rule, we consider a box with only one auxiliary door to simplify the figure. In the other rules, we represent two doors.

a)	$(P.(p_k(c), N)$	$,T$	$) \mapsto$	$(P.(\overline{p_0(c)}, N)$	$,T.(s,k))$
b)	$(P.(p_0(c), N)$	$,T.\overline{(s,k)}$	$) \mapsto$	$(P.(\overline{p_k(c)}, N)$	$,T)$
c)	$(P.(p_0(c'), N)$	$,T.(s,k)$	$) \mapsto$	$(P.(p_0(c'), N).(I_k, N_{s,s'})$	$,T)$
d)	$(P.(p_0(c'), N).(O_{k'}, N_{s,s'})$	$,T$	$) \mapsto$	$(P.(\overline{p_{k'}(c)}, N)$	$,T)$
e)	$(P.(p_0(c'), N).(\overline{I_k}, N_{s,s'})$	$,T$	$) \mapsto$	$(P.(\overline{p_0(c)}, N)$	$,T.\overline{(s,k)})$
f)	$(P.(m, N)$	$,T$	$) \mapsto$	$(P.(\overline{m'}, N)$	$,T)$

Figure 6.9: Rules of context-semantics

6.2 Context semantics

6.2.1 Motivation and definition of the paths

In this section, we fix a library L. For any (s_1, s_2) in the domain of L, we write $(N_{s_1,s_2}, \phi_{s_1,s_2}) = L(s_1, s_2)$.

Let us consider, in the net N_3 of Figure 6.6, the wire from i_1 to i_2. The wire appears in N_3 because the interaction of the two ζ cells in N_2 creates a wire between the second auxiliary ports of the cells. So, intuitively, the path $P_3 = i_1, i_2$ in N_3 comes from $P_2 = i_1, h_1, a_1, a_2, h_2, i_2$ in N_2. The path P_1 in N_1 corresponding to P_2 seems harder to define because the extremities i_1 and i_2 are not in N_1, they correspond to the ports e_1 and e_2 of $N_{\delta,\epsilon}$. We could represent the port i_1 in N_1 by the stack $[(e_0, N); (e_2, N_{\delta,\epsilon})]$, remembering what kind of port is created (e_2) and by which interaction it is created (e_0).

It is exactly the idea of our context semantics. The ports which will appear along the reduction are characterized by stacks called *potential ports* (e.g. $[(e_0, N); (e_2, N_{\delta,\epsilon})]$). The reduction is simulated by paths, which are defined by *contexts* travelling across the net according to a relation $\mapsto$. The contexts are pairs of a potential port P and a *trace* T. The trace T_i represents information about the beginning of the path.

The set Pot^N of *potential ports* of net N is the set of lists $[(p_0, N); (p_1, N_{s_1,t_1}); \cdots ; (p_k, N_{s_k,t_k})]$ such that for each i: p_i is a port of N_{s_i,t_i} and p_{i-1} is the principal port of a cell labelled by t_i. For $P \in Pot^N$, we set $\overline{P.(p, N')} = P.(\overline{p}, N')$.

A *positive trace element* is (s, i) with $s \in \mathcal{S}$ and $1 \leq i \leq \alpha(s)$. The meaning of (s, i) is "I have crossed a cell of symbol s, from its i-th auxiliary port to the principal port". A positive trace is a list of positive trace elements. The set of positive traces is written Tra^+.

A *negative trace element* is $\overline{(s, i)}$ with $s \in \mathcal{S}$ and $1 \leq i \leq \alpha(s)$. The meaning of $\overline{(s, i)}$ is "I will arrive at the principal port of a cell of symbol s. When this happens I will choose to leave it by its i-th auxiliary port". A *trace element* is either a positive trace element or a negative trace element. A *trace* is a list of trace elements. The set of traces is written Tra.

We define the set of contexts of N_1 by $Cont^{N_1} = Pot^{N_1} \times Tra$. The intuitive meaning of a context $([(p_1, N_1); \cdots ; (p_k, N_k)], [(t_1, i_1); \cdots ; (t_l, i_l)])$ is: "N_1 reduces to a net of the shape of Figure 6.10 where for $1 \leq j \leq l$, q_j is the i_j-th auxiliary port of the cell labelled by t_j".

Thus, a potential port is a nesting of interaction nets: in the net N_1 there is a cell which will be part of an active pair and will produce the interaction net N_2 when reducing. Inside this N_2, there is a cell which will be part of an active pair and will produce N_3 when reducing, and so on.

Let us notice that the trace is transformed into a net consisting of a line of cells labelled by the symbols of the trace, the wire linking the cells according to the indices of the trace. We will use this construction

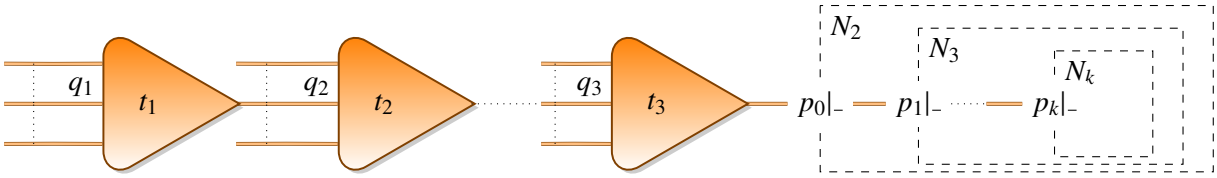


Figure 6.10: Intuition behind the definition of contexts

again, representing the net corresponding to T by  .

Definition 227. *For any net N, we define a relation $\mapsto$ on $Cont^N$ by the rules of Figure 6.9. In those rules, we suppose $s, s' \in \mathcal{S}$, $c, c' \in C^N$, $l^N(c) = s$, $l^N(c') = s'$, $1 \leq k \leq \alpha(s)$, $1 \leq k' \leq \alpha(s')$ and $m, m' \in P_m^N$ with $\sigma_m^N(m) = m'$.*

This relation simulates reduction, in the sense that if $([(p, N)], []) \mapsto^* (P, T)$ and Figure 6.10 is the net intuitively representing (P, T), then $q_1 = p$. The $\mapsto$ relation is deterministic and incomplete (there are contexts C such that $C \not\mapsto$, i.e. $\forall D \in Cont^N, \neg(C \mapsto D)$). Let $C = (P, T) \in Cont^N$, the possible context D such that $C \mapsto D$ is defined depending on the right-most port p of P.

If p is an auxiliary port, we cross the cell and add the information on the trace (rule a).

If p is a principal port, the behaviour depends on whether the right-most trace element t is positive or negative (if the trace is empty, $C \not\mapsto_L$): if t is positive (rule c), then $t = (s, k)$ it corresponds to an active pair $\{c, c'\}$ of symbols $\{s, s'\}$. During reduction $\underline{N_{s,s'}}$ will be glued to N, here we jump to $N_{s,s'}$ to simulate reduction. Else if t is negative (rule b), then $t = \overline{(s, k)}$. It means that at some moment in the path we crossed the cell c from its auxiliary port to its principal port. Then, we found a cell c' such that $\{c, c'\}$ will be an active pair. The $\mapsto$-path lead us to a $\overline{I_k}$ free port. This port will correspond to the k-th auxiliary port of c during reduction, so we pushed $\overline{(s, k)}$ on the trace and took the path backward from c' to c so that we can reach the k-th auxiliary port of c.

If p is free, we are in the net $N_{s,s'}$ corresponding to the interaction of the future active pair $\{c, c'\}$. The behaviour depends on whether p is a $O_{k'}$ (rule d) or a $\overline{I_k}$ (rule e). In the first case, we must go to the k'-th auxiliary port of c' and it is local, in the second we have to go back to the cell c.

If p is a merging port, we cross the merging port (rule f).

Let us recall that we consider the interaction nets up to α-conversion and port merging. So we need to verify that the relation is the same on equivalent nets. The names of the ports have no importance, so the $\mapsto$ relation is the same on α-equivalent nets (up to the renaming of the ports in the potential ports of the contexts). We can verify that whenever $([(p_1, M_1); \cdots; (p_k, M_k)], T) \mapsto^* ([(q_1, N_1); \cdots; (q_l, N_l)], U)$, with:

- For all $1 \leq i \leq k$, p_i is not a merging port and there exists a net M_i' such that $M_i \leftrightarrows_m M_i'$.
- For all $1 \leq j \leq l$, q_j is not a merging port and there exists a net N_j' such that $N_j \leftrightarrows_m N_j'$.

Then, the path $([(p_1, M_1'); \cdots; (p_k, M_k')], T) \mapsto^* ([(q_1, N_1'); \cdots; (q_l, N_l')], U)$ exists.

As an example, the following $\mapsto$-path in the net N of Figure 6.3, goes from the principal port of E to the δ cell which will form an active pair with E. Notice that this δ cell does not exist yet (it will be created by the ζ/δ reduction): $([(b_2, N)], []) \mapsto ([(a_0, N)], [(\zeta, 2)]) \mapsto ([(a_0, N); (I_2, R_{\zeta,\delta})], [])$.

As a more involved example, we will study in the net N, the path between the two ϵ cells created during

the δ/ϵ step of reduction (first step of Figure 6.6).

$$
\begin{aligned}
&([(e_0, N); (r_1, R_{\delta,\epsilon})], []) \mapsto ([(b_2, N)], [\overline{(\delta, 1)}]) \\
&\mapsto ([(a_0, N)], [\overline{(\delta, 1)}; (\zeta, 2)]) \mapsto ([(a_0, N); (d_2, R_{\zeta,\delta})], [\overline{(\delta, 1)}]) \\
&\mapsto ([(a_0, N); (z_1^2, R_{\zeta,\delta})], []) \mapsto ([(a_0, N); (O_1, R_{\zeta,\delta})], [(\zeta, 2)]) \\
&\mapsto ([(\overline{a_1} = a_2, N)], [(\zeta, 2)]) \mapsto ([(b_0, N)], [(\zeta, 2); (\delta, 2)]) \\
&\mapsto ([(b_0, N); (z_2, R_{\delta,\zeta})], [(\zeta, 2)]) \\
&\mapsto ([(b_0, N); (z_2, R_{\delta,\zeta}); (I_2 = O_2, R_{\zeta,\zeta})], []) \\
&\mapsto ([(b_0, N); (d_2^2, R_{\delta,\zeta})], []) \mapsto ([(b_0, N); (\overline{d_2}, R_{\delta,\zeta})], [(\delta, 2)]) \\
&\mapsto ([(e_0, N)], [(\delta, 2)]) \mapsto ([(e_0, N); (e_2, N)], [])
\end{aligned}
$$

Let us consider the net of Figure 6.11 which corresponds[1] to the proof-net of Figure 2.7 (through the encoding defined in Figure 6.2). In the proof-net of Figure 2.7, we had $((e, [\mathtt{r(e)}; \mathtt{l(e)}]), [⅋_r]) \mapsto^9 ((i, [\mathtt{e}]), [⅋_r; \otimes_r])$. Here, one of the corresponding path (on the leftmost net) is:

$$
\begin{aligned}
&([(b, N_1); (b_r, R_{?C,B_\mathbf{B}}); (c, R_{?D,B_\mathbf{B}}); (c_l, R_{?C,B_\mathbf{C}}); (\overline{p}, R_{?D,B_\mathbf{C}})], []) \\
\mapsto&([(b, N_1); (b_r, R_{?C,B_\mathbf{B}}); (c, R_{?D,B_\mathbf{B}}); (\overline{c_l}, R_{?C,B_\mathbf{C}})], [\overline{(?D, 1)}]) \\
\mapsto&([(b, N_1); (b_r, R_{?C,B_\mathbf{B}}); (\overline{c}, R_{?D,B_\mathbf{B}})], [\overline{(?D, 1)}; \overline{(?C, 2)}]) \mapsto ([(b, N_1); (\overline{b_r}, R_{?C,B_\mathbf{B}})], [\overline{(?D, 1)}; \overline{(?C, 2)}; \overline{(?D, 1)}]) \\
\mapsto&([(\overline{b}, N_1)], [\overline{(?D, 1)}; \overline{(?C, 2)}; \overline{(?D, 1)}; \overline{(?C, 1)}]) \mapsto ([(\overline{d_B}, N_1)], [\overline{(?D, 1)}; \overline{(?C, 2)}; \overline{(?D, 1)}]) \\
\mapsto&([(\overline{c_C}, N_1)], [\overline{(?D, 1)}; \overline{(?C, 2)}]) \mapsto ([(\overline{d_C}, N_1)], [\overline{(?D, 1)}]) \mapsto^2 ([(f, N_1)], [(\otimes, 2)])
\end{aligned}
$$

On the rightmost net N_5, this path simply becomes $([(\overline{p}, N)], []) \mapsto ([(f, N_1)], [(\otimes, 2)])$.

Let us notice that, not only the path in interaction net is harder to follow than in linear logic, but it also seems less structured: there is no obvious difference in the potential of the first context between $(c_l, R_{?C,B_\mathbf{C}})$ (which represents a residue, or copy, of the cell represented by the prefix $[(b, N_1); (b_r, R_{?C,B_\mathbf{B}}); (c, R_{?D,B_\mathbf{B}})]$) and $(\overline{p}, R_{?D,B_\mathbf{C}})$ (which is not a residue of the box represented by $[(b, N_1); (b_r, R_{?C,B_\mathbf{B}}); (c, R_{?D,B_\mathbf{B}}); (c_l, R_{?C,B_\mathbf{C}})]$ but a cell inside it). Because of this lack of distinction between "signatures" and "potentials", it is not clear how Lemma 21 would be formulated (let alone proved) in interaction nets. Similarly, most of our definitions and results on the linear logic context semantics rely on the notions of copies, and can not be easily translated into the interaction net context semantics. This is why we defined our criteria on linear logic and leave their extensions to interaction nets for future works.

6.2.2 Soundness of the context semantics

We wrote that $\mapsto$ simulates the reduction of the net. We will prove that the $\mapsto$-paths are stable by reduction. Formally, if $N \to N'$, we will define a projection Π from the potential ports of N to potential ports of N' so that $(P, T) \mapsto^* (Q, U) \Leftrightarrow (\Pi(P), T) \mapsto^* (\Pi(Q), U)$.

In this section, we suppose that $N \to N'$ by reducing the active pair $\{c_1, c_2\}$ labelled by s_1, s_2. We set $(R_1, \phi_1) = L(s_2, s_1)$ and $(R_2, \phi_2) = L(s_1, s_2)$. So $N = N_0 \bowtie_{\phi_2} \Re_{s_1,s_2}$ and $N' = N_0 \bowtie_{\psi \circ \phi_2} R_2$.

The mapping Π defined below loosely correspond to $\pi_{N \to N'}(\_)^{-1}$. Let us notice that this definition is inversed: Π is a mapping from the potentials of the original net to the potentials of the reduced nets. On the contrary, $\pi_{G \to H}(\_)$ is a mapping from the contexts of the reduced net to the contexts of the original net.

[1] To be precise, because of the mismatch between proof-nets and interaction nets, we replaced the right box by a dereliction

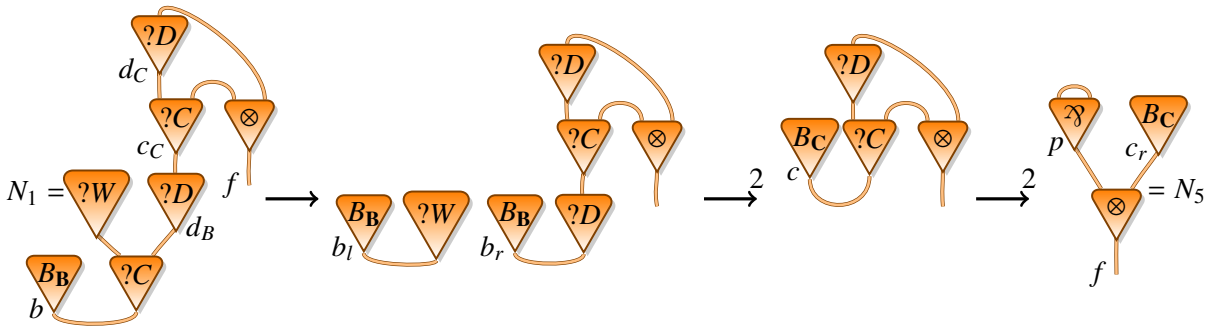


Figure 6.11: Reduction of the encoding of a proof-net in interaction nets.

Definition 228. *We define a mapping* Π *from* Pot^N *to* $Pot^{N'}$ *which depends on the left-most port* p*:*

- *If* $p \in P^{N_0}$*, we set* $\Pi([(p, N)]@P) = [(p, N')]@P$*.*
- *If* $p = p_0(c_i)$ *for* $i \in \{1, 2\}$*, we set:*

$$\Pi([(p_0(c_i), N); (r, R_i)]@P) = [(r, N')]@P$$

- *Otherwise,* Π *is undefined.*

The two next propositions show that the projection behaves as expected. Lemma 231 shows that the paths are preserved along reduction. It requires the potentials P and Q to be in the domain of the projection, this condition is the counterpart of the "long enough" condition on paths in GoI settings [27].

Proposition 229. *Let* $P' \in Pot^{N'}$*, then there exists* $P \in Pot^N$ *such that* $\Pi(P) = P'$*.*

Proof. Either P' was already a potential of N (it represents a part of the net which is not modified by the reduction $N \to N'$). In this case $\Pi(P') = P'$. Else, P' is a potential of the shape $[(p, N')]@Q'$ with p a port of R_{s_1,s_2} in this case $\Pi([(p^0(c_2), N); (p, R_{s_1,s_2})]@Q') = \Pi([(p^0(c_1), N); (p, R_{s_2,s_1})]@Q') = P'$. To be more precise, let us notice that those alternatives are not mutually exclusive: because we consider the nets up to merging, we have $\Pi([(p^0(c_2), N); (\overline{I_i}, R_{s_1,s_2})]) = \Pi([(p^0(c_1), N); (O_i, R_{s_2,s_1})]) = [(\overline{p^i(c_1)}, N)]$. □

Proposition 230. *Let* $P \in Pot^N$ *such that* $\Pi(P)$ *is defined, then* $\Pi(\overline{P})$ *is defined and* $\overline{\Pi(P)} = \Pi(\overline{P})$*.*

Lemma 231. *If* $T, U \in Tra$*,* $P, Q \in Pot^N$*,* $\Pi(P) = P'$ *and* $\Pi(Q) = Q'$ *then* $(P, T) \mapsto^* (Q, U) \Rightarrow (P', T) \mapsto^* (Q', U)$

If $T, U \in Tra$*,* $P', Q' \in Pot^{N'}$ *and* $(P', T) \mapsto^+ (Q', U)$ *then there exists* P, Q *such that* $\Pi(P) = P'$*,* $\Pi(Q) = Q'$ *and* $(P, T) \mapsto^+ (Q, U)$

Proof. We will only prove the first statement. The proof of the second is quite similar.

We will prove it for minimal such paths: let us suppose that $(P, T) \mapsto^* (Q, U)$ and that for every other context (R, V) in the path, $\Pi(R)$ is undefined. We will show that $(P', T) \mapsto^* (Q', U)$. Then, the lemma is straightforward because any $\mapsto^*$ path between potentials in the domain of Π can be decomposed in such smaller paths.

We set $[(p, N)]@P_1 = P$ and $[(q, N)]@Q_1 = Q$.

- If $p, q \in P^{N_0}$, then $(P, T) \mapsto (Q, U)$ (the path has length 1), $P' = [(p, N')]@P_1$ and $Q' = [(q, N')]@Q_1$. Given that all $\mapsto$ rules are local, and that ports of N_0 are unaffected by the reduction, $(P', T) \mapsto (Q', U)$.

- If p and q belong respectively to P^{R_i} and P^{R_j} with $i, j \in \{1, 2\}$, then a careful observation of the $\mapsto$ rules shows that the only possibility is $i = j$ and $(P, T) \mapsto (Q, U)$. Because $\Pi(P)$ and $\Pi(Q)$ are defined, we have $P = [(p_0(c_i), N); (r, R_i)]@P_1$ and $Q = [(p_0(c_i), N); (s, R_i)]@Q_1$ for some $r, s \in P^{R_i}$. So $P' = [(r, N')]@P_1$ and $Q' = [(s, N')]@Q_1$. Considering that $\mapsto$ is local, $(P', T) \mapsto (Q', U)$.

- If $p \in P^{N_0}$ and $q \in P^{R_i}$ (with $i \in \{1, 2\}$, we will write $j = 3 - i$ to refer to the other cell), then the only possibility is that p is a free port of N_0 which, in N, is merged with the free port i_k of $\mathfrak{R}_{s_1,s_2}$ and, in N', is merged with the free port $\psi(i_k)$ of R_i. So, we have $(P, T) = ([(p, N)], T) \mapsto ([(p_k(c_j), N)], T) \mapsto ([(p_0(c_i), N)], T.(s_j, k)) \mapsto ([(p_0(c_i), N); (I_k, R_i)], T) = (Q, U)$. We can notice that $P' = [(p, N')]$ and $Q' = [(I_k, N')]$. We get $(P', T) \mapsto ([(\overline{\sigma_m^{N'}(p)}, N')], T) = (Q', U)$.

- If $p \in P^{R_i}$ (with $i \in \{1, 2\}$, we will write $j = 3 - i$) and $q \in P^{N_0}$, then $\overline{q}$ is a free port of N_0 which, in N, is merged with a free port of $\mathfrak{R}_{s_1,s_2}$ and, in N', is merged with a free port $\psi(i_k)$ of R_i. And, either $p = \overline{I_k} = \psi(i_k)$ (and $1 \leq k \leq \alpha(s_j)$) or $p = O_k$ (and $1 \leq k \leq \alpha(s_i)$).

  - If $p = \overline{I_k}$, $(P, T) = ([(p_0(c_i), N); (\overline{I_k}, R_i)], T) \mapsto ([(p_0(c_j), N)], T.\overline{(s_j, k)}) \mapsto ([(\overline{p_k(c_j)}, N)], T) \mapsto ([(q, N)], T) = (Q, U)$. We can notice that $P' = [(\psi(i_k), N')]$ and $Q' = [(q, N')]$. In N', $\psi(i_k)$ is merged with $\overline{q}$ so $(P', T) \mapsto (Q', T) = (Q', U)$.
  - If $p = O_k$, $(P, T) = ([(p_0(c_i), N); (O_k, R_i)], T) \mapsto ([(\overline{p_k(c_i)}, N)], T) \mapsto ([(q, N)], T) = (Q, U)$. We can notice that $P' = [(O_k, N')]$ and $Q' = [(q, N')]$. In N', O_k is merged with $\overline{q}$ so $(P', T) \mapsto (Q', U)$.

□

In particular, the successive projections of free ports of a net will always be defined along a reduction sequence. So a path between two free ports of a net will always be stable along reduction, as stated by Corollary 232.

Corollary 232. *If $M \to^* N$, $p, q \in P_f^M$ and $T, U \in Tra$, then*

$$([(\overline{p}, M)], T) \mapsto^* ([(q, M)], U) \Leftrightarrow ([(\overline{p}, N)], T) \mapsto^* ([(q, N)], U)$$

Let Π_1, Π_2, Π_3 and Π_4 be the projections corresponding to the reduction steps of Figures 6.3 and 6.6. We have $\Pi_1([(e_0, N); (r_1, R_{\delta,\epsilon})]) = [(e_0, N_1); (r_1, R_{\delta,\epsilon})]$, then $\Pi_2([(e_0, N_1); (r_1, R_{\delta,\epsilon})]) = [(h_1, N_2)]$, next $\Pi_3([(h_1, N_2)]) = [(i_2, N_3)]$ and $\Pi_4([(i_2, N_3)])$ is not defined.

We can observe the reductions of the path of N $([(e_0, N); (r_1, R_{\delta,\epsilon})], []) \mapsto^{13} ([(e_0, N); (e_2, N)], [])$ which becomes $([(e_0, N_1); (r_1, R_{\delta,\epsilon})], []) \mapsto^2 ([(d_2, N_1)], [\overline{(\delta, 1)}]) \mapsto^4 ([(\overline{a_2'}, N_1)], [(\zeta, 2)]) \mapsto^5 ([(e_0, N); (e_2, N)], [])$ in N_1, then $([(h_1, N_2)], []) \mapsto^3 ([(i_2, N_2)], [])$ in N_2 and finally $([(i_2, N_3)], []) \mapsto^0 ([(i_2, N_3)], [])$ in N_3.

6.3 Complexity bounds

In this section, we define *canonical cells*, which are the potential ports which correspond to cells that will really appear during reduction. Then we use the canonical cells to define a weight $W_N \in \mathbb{N} \cup \{\infty\}$ for any net N such that, if $M \to N$, then $W_M \geq W_N + 1$. It follows that the length of any reduction sequence from M is bounded by W_M. Notice that it is not true that $W_M > W_N$ because if $W_M = \infty$, then $W_N = \infty$.

The approach is inspired by Dal Lago's context semantics for linear logic [20]. First, Dal Lago's weight allowed to show that every proof-net of some linear logic subsystems verified complexity properties (e.g. every proof-net of *LLL* reduce in polynomial time w.r.t the size of the argument, whatever the reduction strategy). These bounds were previously known, but Dal Lago's proofs were much shorter. Then, his tool was used to prove strong bounds which were previously unknown [3]. We hope that our tool will lead to similar results.

We want to capture the "cells which will appear during reductions beginning by N". Such a cell is either a cell of N, or appears during the reduction of two cells c_1 and c_2 such that: c_1 and c_2 both appear during reductions beginning by N, and $\{c_1, c_2\}$ will form an active pair. This is the intuition behind the following definition of canonical cells.

Definition 233. *We define the set Can^N of canonical cells of N by induction:*

- *For every cell c of N, $[(p_0(c), N)]$ is a canonical cell*
- *If $P_1.(p_0(c_1), N_1) \in Can^N$, $(P_1.(\overline{p_0(c_1)}, N_1), []) \mapsto (P_2.(p_0(c_2), N_2), [])$, $l^N(c_1) = s_1$, $l^N(c_2) = s_2$ and $L(s_1, s_2)$ is defined. Then for every cell c of N_{s_2,s_1}, $P_1.(p_0(c_1), N_1).(p_0(c), N_{s_2,s_1}) \in Can^N$.*

Lemma 234. *Let us suppose that $N \to_L N'$ by reducing the active pair $\{c_1, c_2\}$ and Π is the associated projection.*

If $P \in Can^N$, then either $\Pi(P)$ is defined and $\Pi(P) \in Can^{N'}$ or P corresponds to one of the ports of the active pair: $P \in \{[(p_0(c_1), N)], [(p_0(c_2), N)]\}$.

If $\Pi(P)$ exists and is in $Can^{N'}$, then $P \in Can^N$.

As an example, let us consider the net N of Figure 6.3. We can show that $C_1 = [(e_0, N); (e_1, R_{\delta,\epsilon})]$ is a canonical cell. Indeed, b_0 is a principal port of N so $[(b_0, N)]$ is a canonical cell. We know that $([(\overline{b_0}, N)], []) \mapsto^0 ([(a_0, N)], [])$ and $L(\zeta, \delta)$ is defined so $[(b_0, N); (d_2, N_{\delta,\zeta})]$ is a canonical cell. Finally, $([(b_0, N); (\overline{d_2}, N_{\delta,\zeta})], []) \mapsto^1 ([(e_0, N)], [])$ and $L(\delta, \epsilon)$ is defined so $[(b_0, N); (d_2, N_{\delta,\zeta}); (e_1, R_{\epsilon,\delta})]$ is canonical.

Similarly, $C_2 = [(a_0, N); (d_2, N_{\zeta,\delta}); (e_1, N_{\epsilon,\delta})]$ and $C_3 = [(e_0, N); (e_1, N_{\delta,\epsilon})]$ are canonical. Let Π_1, Π_2, be the projections corresponding to $N \to N_1$ and $N_1 \to N_2$ (Figures 6.3 and 6.6). We can observe that $\Pi_2 \circ \Pi_1(C_1) = \Pi_2 \circ \Pi_1(C_2) = \Pi_2 \circ \Pi_1(C_3)$ so, intuitively, there are three canonical cells corresponding to the same future cell.

The following theorem corresponds to the main result of [20]. The intuition behind it is that each reduction step erases two canonical potentials: the ones corresponding to the active pair.

Theorem 235. *For every interaction-net N, the length of any interaction sequence beginning by N is bounded by :*

$$T_N = \sum_{P \in C^N} \frac{1}{2^{|P|}}$$

Proof. We suppose that N reduces to N' by reducing the active pair $\{c, d\}$ labelled by s, t, Π is the associated projection and D its domain. For any $P' \in Can^{N'}$,

- Either $P' = [(p', N')]@Q$ with p' a port of $R_{s,t}$ then p' is also a port of $R_{t,s}$ (or vice-versa). So $\Pi^{-1}(P') = \{[(p_0(c), N); (p', R_{t,s})]@Q, [(p_0(d), N); (p', R_{s,t})]@Q\}$.
- Or $P' = [(p', N')]@Q$ and $\Pi^{-1}(P') = \{[(p', N)]@Q\}$.

So, for any $P' \in Can^{N'}$, we have:

$$\sum_{P \in \Pi^{-1}(P')} \frac{1}{2^{|P|}} = \frac{1}{2^{|P'|}}$$

This gives the following equations:

$$\begin{aligned} T_N &= \sum_{P \in C^N \cap D} \frac{1}{2^{|P|}} + \sum_{P \in C^N - D} \frac{1}{2^{|P|}} \\ T_N &= \sum_{P' \in C^{N'}} \frac{1}{2^{|P'|}} + \frac{1}{2^{|[(p_0(c), N)]|}} + \frac{1}{2^{|[(p_0(c), N)]|}} \\ T_N &= T_{N'} + 1 \end{aligned}$$

□

However we were unable to use this context semantics to prove complexity bounds in a way similar to our proofs on subsystems of linear logic, even for trivial libraries. It seems we need further tools (corresponding to the notion of copies, acyclicity of proof-nets and subtree properties in [20]) to ease the use of Theorem 235 to prove bounds for interaction nets system. With the current definitions, it is not even clear what a formulation (let alone the proof) of Lemma 21 in interaction nets would be.

The most obvious way to adapt the notion of copies to interaction nets would be to define the copies of the canonical cell P as the canonical cells of the shape $P.(q, N)$. However, we would have a mismatch with the definition of copies in the case of linear logic. For instance, every cell (even in the case of non-terminating nets) would have a finite number of copies: either 0 (if the cell is not part of a redex) or the number of cells in the net $N_{s,t}$ corresponding to its redex. Although we have some ideas to fix this mismatch, this seems to be a complex issue which is beyond the scope of this thesis.

6.4 Denotational semantics

6.4.1 Observational equivalence

Corollary 232 shows us that the paths from a free port to a free port are stable along the reduction. Hence, it seems natural to define a denotational semantics based on those paths. We would like our semantics to enjoy a *full abstraction* property, i.e. a theorem stating that two proof-nets have the same semantics if and only if they are *observationally equivalent*.

Let us recall that, in general, two programs P and Q are said observationally equivalent if for all context $C[\ ]$, such that the execution of $C[P]$ outputs some value v, the execution of $C[N]$ outputs the same value v[2]. In a framework as general as interaction nets, there are several possible notions of "outputting a value", each gives a different observational equivalence. The observational equivalence $\approx$ we will consider is based on an observational equivalence $\simeq$ defined by Mazza [56]. We modified a bit the equivalence, because in some farfetched libraries, $\begin{smallmatrix} a & b \\ c & d \end{smallmatrix} \simeq \begin{smallmatrix} a & b \\ c & d \end{smallmatrix}$. In our point of view, interaction nets are about "what can interact with what". So, if in a net a can only interact with b, it can not be equivalent to a net where a can only interact with d. In every system studied by Mazza in [56], the property $(N_1 \approx N_2) \Leftrightarrow (N_1 \simeq N_2)$ holds. Both observational equivalences are based on *observable paths* defined below.

Definition 236. *Let N be an interaction net, an observable path of N is a sequence $p_0, p_1, \cdots, p_k$ of ports of N such that we do not cross active pairs (if p_i is an auxiliary port, for $i < j \leq k$, p_j is not a principal port) and for every $i < k$:*

- *If $p_i = p_j(c)$ (with $j > 0$), then $p_{i+1} = \overline{p_0(c)}$ (crossing a cell from an auxiliary port to the principal port).*
- *If $p_i = p_0(c)$, then either there exists $j > 0$ such that $p_{i+1} = \overline{p_j(c)}$ (crossing a cell from the principal port to an auxiliary port) or $p_{i+1} = \overline{p_0(c)}$ (bouncing on a principal port).*
- *If $p_i \in P_m^N$, $p_{i+1} = \overline{\sigma_m^N(p_i)}$ (crossing a merging port).*

The observable and $\mapsto$-path are closely linked. If $(P_1, T_1) \mapsto \cdots \mapsto (P_n, T_n)$, and $[(p_1, N)], \cdots, [(p_k, N)]$ is the subsequence of $P_1, \cdots, P_n$ of potentials of length 1, then $p_1, \cdots, p_k$ is an observable path. In fact the observable paths which can be obtained in this way are exactly the observable paths which can not be eliminated by reduction.

We wrote earlier that, according to us, interaction nets are about "what can interact with what". Thus, in interaction nets, an observation is a property of the shape "the ports p and q can communicate together". This communication is meaningful only it can be used by another process (the internal communications between cells are not taken into accounts) so, in the definition of observations, we only consider free ports p and q.

Definition 237. *Let p, q be free ports of N. If $N \to^* N'$ and there exists an observable path from $\sigma_w^{N'}(p)$ to q, then we write $N \Downarrow_q^p$.*

Definition 238 (observational equivalence)**.** *Let N_1, N_2 be nets with $P_f^{N_1} = P_f^{N_2}$, then we write $N_1 \approx N_2$ if for all nets N, ϕ partial injection from $P_f^{N_1}$ to P_f^N, and $p, q \in P_f^{N_1 \bowtie_\phi N}$:*

$$(N_1 \bowtie_\phi N) \Downarrow_q^p \quad \Leftrightarrow (N_2 \bowtie_\phi N) \Downarrow_q^p$$

[2]Notice that the word "context" is not used here in our meaning of "token travelling through the net", but in the usual meaning of a "program with a hole".

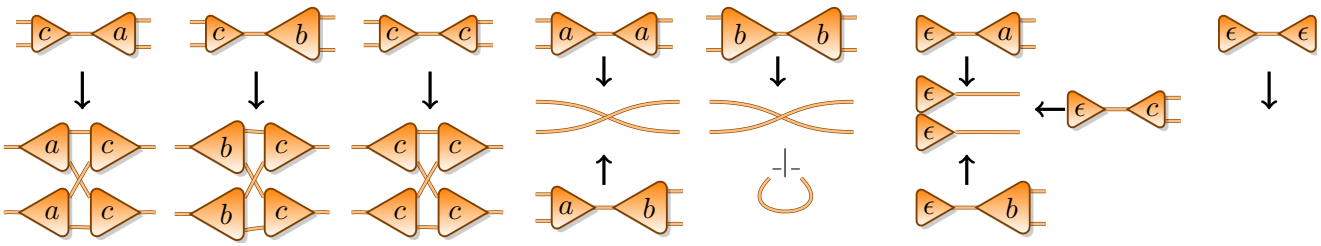


Figure 6.12: Reduction rules for libraries L and L_e

We wrote that our definition is inspired by Mazza's observational equivalence. The observation considered by Mazza is "there is a communication between two free ports". The difference is that, Mazza does not take into account which free port communicates with which free port.

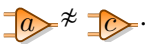
Definition 239. *[56] Let N_1, N_2 be nets with $P_f^{N_1} = P_f^{N_2}$, then we write $N_1 \simeq N_2$ if for all nets N and ϕ partial injection from $P_f^{N_1}$ to P_f^N,*

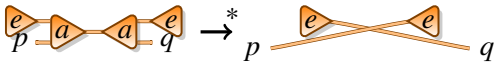
$$\exists p, q \in P_f^{N_1 \bowtie_\phi N}(N_1 \bowtie_\phi N)\Downarrow_q^p \Leftrightarrow \exists p, q \in P_f^{N_2 \bowtie_\phi N}(N_2 \bowtie_\phi N)\Downarrow_q^p$$

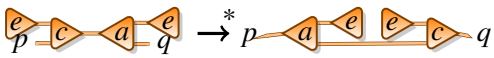
If the communications between free ports are the same in $N_1 \bowtie_\phi N$ and $N_2 \bowtie_\phi N$ then, in particular, there is a communication in $N_1 \bowtie_\phi N$ iff there is a communication in $N_2 \bowtie_\phi N$. This means that $(N_1 \approx N_2)$ implies $(N_1 \simeq N_2)$. However, the other implication is not true in general.

As an example, let us define the library L (resp. L_e) whose symbols are $\{a, b\}$ (resp. $\{a, b, c, e\}$), the reduction rules are given in Figure 6.12. One can observe that c duplicates every cell, e erases every cell, the other interactions (a/a, a/b and b/b) create wires between the free ports (b/b also creates a cycle).

In the library L, for any net N and $p \in P_f^N$, there exists $q \in P_f^N$ such that $N\Downarrow_q^p$. So, for any N_1 and N_2 with the same number of free ports, $N_1 \simeq N_2$. On the contrary, $N_1 =$  $\not\approx$  $= N_2$. Indeed, let us consider $N =$  , then $(N_1 \bowtie N)\Downarrow_q^p$ and $\neg((N_2 \bowtie N)\Downarrow_q^p)$.

In L_e, we can prove  and  . On the contrary,  . Indeed, let us consider $N =$  , then $\neg\left((N_1 \bowtie N)\Downarrow_q^p\right)$ and $(N_2 \bowtie N)\Downarrow_q^p$ as we can observe by reduction:

$N_1 \bowtie_\phi N =$ 

$N_2 \bowtie_\phi N =$ 

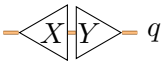
Finally, if we extended the library L_{sort} with another cell T performing sort in any way (for example merge sort), then we would have  . But  .

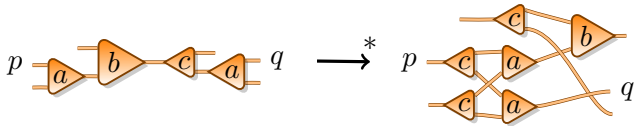
6.4.2 Definition of a denotational semantics

To define a denotational semantics matching our observational equivalence $\approx$, we need a mapping $|\_, \_, \_, \_|$ from $(Tra^+)^4$ to set of pairs of positive traces.

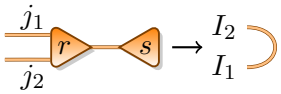
Definition 240. *Let $S, T, U, V \in Tra^+$, then if we set $N =$*  *, then we define $|S, T, U, V|$ as*

$$\left\{ (X, Y) \in (Tra^+)^2 \;\middle|\; \exists P \in Can^N, \begin{array}{c} (P, []) \mapsto^* ([(p, N)], X) \\ (\overline{P}, []) \mapsto^* ([(q, N)], X) \end{array} \right\}$$

We have $(X, Y) \in |S, T, U, V|$ iff  reduces to a net N' such that  is a subnet of N'. For example, in L_e, $|[(a, 1); (b, 2)], [(c, 1)], [], [(a, 2)]| = \{([(c, 1)], [(a, 1)])\}$ because



The interpretation $[N]$ of a net will be the set of unordered pairs $\{(p, S), (q, V)\}$ with p, q free ports of N and S, V positive traces such that, if we define M as the net  then $M \Downarrow_b^a$. So, $[N]$ corresponds to the observations of N when glued with a net consisting of only two lines of cells. Thus, the full abstraction of the semantics means "If for every net N, $N_1 \bowtie N$ and $N_2 \bowtie N$ have the same observations, then they have the same observations when N consists of two lines of cells." Thus, the proof of the full abstraction offers no real difficulty.

The soundness means that "If whenever N consists of two lines of cells, $N_1 \bowtie N$ and $N_2 \bowtie N$ have the same observations, then this is also true for an arbitrary net N". In fact, soundness is not true in the general case. However, we did prove soundness in the case of *crossing* libraries. A library is said *bouncing* if there is an interaction rule (R, ψ) and free ports $\overline{I_k}$, $\overline{I_l}$ of R such that $R \Downarrow_{\overline{I_l}}^{\overline{I_k}}$. A typical bouncing rule is  . A library is said *crossing* if it is not bouncing. For the rest of the paper, we consider that L is crossing.

Definition 241. *Let N be an interaction net, $[N]$ is the set*

$$\left\{ \begin{array}{l} \{(p, S), (q, V)\} \\ p, q \in P_f^N \\ S, V \in Tra^+ \end{array} \;\middle|\; \exists \begin{array}{l} P \in Pot_+^N \\ T, U \in Tra \end{array}, \begin{array}{c} (P, []) \mapsto^* ([(p, N)], T) \\ (\overline{P}, []) \mapsto^* ([(q, N)], U) \\ |S, T, U, V| \neq \varnothing \end{array} \right\}$$

Where $\{(p, S), (q, V)\}$ represents a multiset (unordered pair in this case).

6.4.3 Stability of [_] by reduction and gluing

Theorem 242. *If $N \to N'$, then $[N] = [N']$*

Proof. Follows from Lemma 231 and the definition of $[N]$. □

The proof of stability of [ ] by gluing is the most complex of this paper. It is necesary to prove the soundness of [ ] with respect to $\approx$. The proof requires the following lemmas.

Lemma 243. $\bigcup_{(Z,W) \in |[],[],V,Y|} |X, T, U, Z| \sim |X, T, U @ V, Y|$

Lemma 244. $\bigcup_{(X,Y) \in |[],[],U,V|} |S, X @ T, Y, V| \sim |S, T, U, Z @ V|$

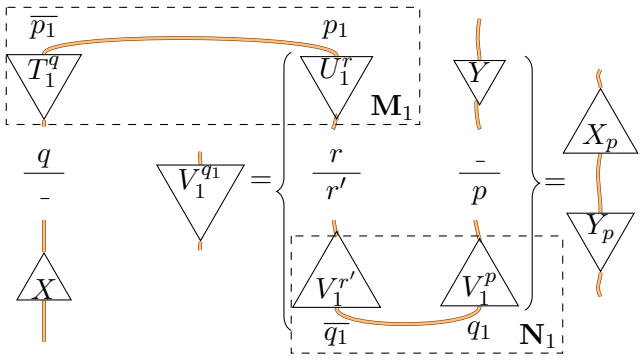


Figure 6.13: Sketch of the net G_1 in the proof of Theorem 246.

Lemma 245. *Let $P, Q \in Pot^N$ and $T, U \in Tra$,*

$$\left.\begin{array}{l}(P,[]) \mapsto (R,T)\\ (Q,[]) \mapsto (\overline{R},U)\\ |[],[],T,U| = (S,V)\end{array}\right\} \Rightarrow \exists R' \in Pot^N,\ \begin{array}{l}(R',[]) \mapsto (\overline{P},S)\\ (\overline{R'},[]) \mapsto (\overline{Q},V)\end{array}$$

Theorem 246. *Let M_1, N_1, M_2, N_2 be nets such that $[M_1] = [N_1]$, $[M_2] = [N_2]$ and ϕ an injection from $P_f^{M_1} = P_f^{M_2}$ to $P_f^{N_1} = P_f^{N_2}$, then $[M_1 \bowtie_\phi N_1] = [M_2 \bowtie_\phi N_2]$.*

Proof. For concision, we will write $G_1 = M_1 \bowtie_\phi N_1$ and $G_2 = M_2 \bowtie_\phi N_2$. We will not consider the $\mapsto_m$ normal versions of R_1 and R_2 but will leave the merging ports created on the connecting ports (the ports in the domain or codomain of ϕ) untouched. We need a notion of $\mapsto$-path with a bounded number of alternations between ports of M_1 and ports of N_1. For every $i \in \mathbb{N}$, we define a relation $\rightarrowtail_i$ on $Cont^{G_1}$ by: $(P,T) \rightarrowtail_i (Q,U)$ iff we are in one of those cases:

$$\begin{cases} i = 0 \text{ and } (P,T) = (Q,U)\\ (P,T) \mapsto ([(p,G_1)],V) \rightarrowtail_{i-1} (Q,U) \text{ with } p \in P_f^{M_1} \cup P_f^{N_1}\\ (P,T) \mapsto (R,V) \rightarrowtail_i (Q,U) \text{ with } R \notin [((P_f^{M_1} \cup P_f^{N_1}),G_1)]\end{cases}$$

We define the $\rightarrowtail_i$ relations on $Cont^{G_2}$ similarly. We will prove the following property $\mathcal{P}(i+j)$ by induction on $i+j$:

"Let $p, q \in P_f^{M_1} \cup P_f^{N_1}$ and $P_1 \in Pot^{G_1}$ such that $(P_1,[]) \rightarrowtail_i ([(p,G_1)],T_1)$, $(\overline{P_1},[]) \rightarrowtail_j ([(q,G_1)],U_1)$ and $|S,T_1,U_1,V|$ is defined, then there exists $P_2 \in Pot^{G_2}$ such that $(P_2,[]) \mapsto^* ([(p,G_2)],T_2)$, $(\overline{P_2},[]) \mapsto^* ([(q,G_2)],U_2)$ and $|S,T_2,U_2,V|$ is defined."

This directly implies that $[G_1] \subseteq [G_2]$ because $P_f^{G_1} \subseteq P_f^{M_1} \cup P_f^{N_1}$. Because the roles of (M_1,N_1) and (M_2,N_2) are symmetrical, it will imply $[G_2] \subseteq [G_1]$ so $[G_1] = [G_2]$.

Let us suppose that $\mathcal{P}(i+j-1)$ is true. Let $p,q,r \in P_f^{M_1} \cup P_f^{N_1}$ and $P_1 \in Pot^{G_1}$ such that $(P_1,[]) \rightarrowtail_i ([(r,G_1)],U_1^r) \rightarrowtail_1 ([(p,G_1)],T_1^p)$, $(\overline{P_1},[]) \rightarrowtail_j ([(q,G_1)],T_1^q)$ and $|X,T_1^q,T_1^p,Y|$ is defined.

Without loss of generality, we will suppose that $r \in P_f^{M_1}$. Thus, $r \in P_m^{G_1}$, let us write $r' = \phi(r) = \sigma_m^{G_1}(r)$. Then, $([(r,G_1)],U_1^r) \mapsto ([(\overline{r'},G_1)],U_1^r) \rightarrowtail_1 ([(p,G_1)],T_1^p)$.

We reduce M_1 and N_1 to nets M_1' and N_1' such that, if we write $G_1' = M_1 \bowtie_\phi N_1$, $\Pi(P_1)$ has shape $[(p_1,G_1')]$ and the paths $([(p_1',G_1')],[]) \rightarrowtail_i ([(r,G_1')],U_1^r)$, $([(\overline{p_1'})],[]) \rightarrowtail_j ([(q,G_1')],T_1^q)$ and $([(r,G_1')],U_1^r) \rightarrowtail_1 ([(p,G_1)],T_1^p)$ do not cross active pairs. The net G_1' is sketched in Figure 6.13.

The path $([(\overline{r'}, G'_1)], U_1^r) \rightarrowtail_1 ([(p, G'_1)], T_1^p)$ does not cross active pairs so the potential ports are first principal ports, then auxiliary ports and finally the free port $[(p, G'_1)]$. Let $[(q_1, G'_1)]$ be the first non-principal potential port of length 1 of the path. We have $([(p, G'_1)], []) \mapsto^* ([(q_1, G'_1)], V_1^{q_1}) \mapsto^* ([(p, G'_1)], V_1^{q_1} @ V_1^p)$ with $T_1^p = V_1^{q_1} @ V_1^p$.

As L is crossing, there exists $V_1^{r'} \in Tra^+$ such that $([(q_1,G'_1)],[]) \mapsto^* ([(r', G'_1)], V_1^{r'})$ and $|[],[],V_1^{r'}, U_1^r| = ([], V_1^{p_1})$.

By Lemma 243, there exists $(Y^p, X^p) \in |[], [], V_1^p, Y|$, such that $|X, T_1^q, U_1^r, Y^p @ V_1^{r'}|$ is not empty. By induction hypothesis, there exists $P_2 \in Pot^{G_2}$ such that $(P_2, []) \mapsto^* ([(r, G_2)], U_2^r)$, $(\overline{P_2}, []) \mapsto^* ([(q, G_2)], T_2^q)$ and $|X, T_2^q, U_2^r, Y^p @ V_1^{r'}|$ is defined.

By Lemma 243, there exists $(Y^q, X^q) \in |X, T_2^q, [], []|$, such that $|X^q, [], U_2^r, Y^p @ V_1^{r'}| \neq \emptyset$. But $|X^q, [], U_2^r, Y^p @ V_1^{r'}| = |X^q @ U_2^r, V_1^{r'}, [], Y^p|$ so, by Lemma 243 $|X^q @ U_2^r, V_1^{r'}, V_1^p, Y| \neq \emptyset$. We know that $[N'_1] = [N_1] = [N_2]$ so there exists some $Q_2 \in Can^{N_2}$ such that $(Q_2, []) \mapsto^* ([(p, N_2)], V_2^p)$, $(\overline{Q_2}, []) \mapsto^* ([(r', N_2)], V_2^{r'})$ and $|X^q @ U_2^r, V_2^{r'}, V_2^p, Y|$ is defined.

By Lemma 244, there exists $(W_2^p, W_2^q) \in |[], [], V_2^{r'}, U_2^r|$ such that $|X^q, W_2^q, W_2^p @ V_2^p, Y| \neq \emptyset$. So by Lemma 243, $|X, W_2^q @ T_2^q, W_2^p @ V_2^p, Y| \neq \emptyset$. From Lemma 245, there exists some potential port $R_2 \in Can^{G_2}$ such that $(R_2, []) \mapsto^* (\overline{\overline{Q_2}}, W_2^p) \mapsto^* ([(p, G_2)], W_2^p @ V_2^p)$ and $(\overline{R_2}, []) \mapsto^* (\overline{P_2}, W_2^q) \mapsto^* ([(q, G_2)], W_2^q @ T_2^q)$. □

6.4.4 Soundness and full abstraction

Lemma 247. *If $P, Q \in Can^N$ and $(P, T) \mapsto^* (Q, U)$, then we can reduce N to a net N' such that Π is the associated composition of projections, $\Pi(P)$ and $\Pi(Q)$ have shape $[(p, N')]$ and $[(q, N')]$, and the path $([(p, N')], T) \mapsto^* ([(q, N')], U)$ does not cross active pairs*

Proof. We prove it by induction on $|P| + |Q|$. If $|P| + |Q| = 2$ and the path crosses an active pair, then we can reduce the pair. Notice that the path $([(p, N')], T) \mapsto^* ([(q, N')], U)$ is strictly shorter than the path $(P, T) \mapsto^* (Q, U)$. So we get the result after finitely many such reductions.

Else, we have $P = P_1.(p_0(c), N_1).(r, R_{s,t})$ and $(P_1.(\overline{p_0(c)}, N_1), []) \mapsto^* ([P_2.(p_0(d), N_2)], [])$ (with $l^N(c) = s$ and $l^N(d) = t$). By induction hypothesis, we can reduce N so that this path does not cross active pairs. So this path has length 0, $\{c, d\}$ becomes an active pair that we can reduce. Then $|\Pi(P)| < |P|$ and $|\Pi(Q)| \leq Q$, so we can apply the induction hypothesis. □

Lemma states that if $[N_1] = [N_2]$ then the observations (the $(N_1)\Downarrow_q^p$) on N_1 and N_2 are the same. As we proved that [ ] is stable by context, we will get that if $[N_1] = [N_2]$, for any N, the observations on $N_1 \bowtie N$ and $N_2 \bowtie N$ are the same. This is exactly the soundness of [ ] with respect to $\approx$.

Lemma 248. *If $[N_1] = [N_2]$ and $p, q \in P_f^{N_1} = P_f^{N_2}$, then:*

$$N_1 \Downarrow_q^p \Leftrightarrow N_2 \Downarrow_q^p$$

Proof. We consider the $\rightarrow_m$-normal representations of N_1 and N_2. Notice that N_1 and N_2 play symmetric roles so we only need to prove one implication. Let us suppose that $N_1 \Downarrow_q^p$, then there exists some net N'_1 such that $N_1 \rightarrow^* N'_1$ and there exists an observable path in N'_1 from $\overline{p}$ to q.

By definition, the observable path is a (possibly empty) sequence of principal ports followed by a (possibly empty) sequence of auxiliary ports and the free port q. Let us consider r, the first port of the path which is not a principal port.

Then there is an observable path from r to q with only auxiliary ports (except q which is free), and there is an observable path from $\overline{r}$ to p with only auxiliary ports (except p which is free). Thus there exists $T_1, U_1 \in Tra^+$ such that $([(\overline{r}, N_1')], []) \mapsto^* ([(p, N_1')], T_1)$ and $([(r, N_1')], []) \mapsto^* ([(q, N_1')], U_1)$. We can notice that $|[], T_1, U_1, []|$ is defined.

We know that $(p, q, [], []) \in [N_1'] = [N_1] = [N_2]$. Thus, there exists $Q \in Can^{N_2}$, $T_2, U_2 \in Tra^+$ such that $(\overline{Q}, []) \mapsto^* ([(p, N)], T_2)$ and $(Q, []) \mapsto^* ([(q, N)], U_2)$. Thanks to Lemma 247, we know that we can reduce N_2 to a net N_2' such that the projection of Q has shape $[(s, N_2')]$ and the paths $([(\overline{s}, N_2')], []) \mapsto^* ([(p, N)], T_2)$ and $([(s, N_2')], []) \mapsto^* ([(q, N)], U_2)$ do not cross active pairs.

Thus, in N_2', there are observable paths from $\overline{s}$ to p and from s to q with only auxiliary ports. This means that there is an observable paths, in N_2', from $\overline{p}$ to q. So $N_2 \Downarrow_q^p$. □

Theorem 249 (soundness)**.** *If* $P_f^{N_1} = P_f^{N_2}$ *and* $[N_1] = [N_2]$*, then* $N_1 \approx N_2$

Proof. Let us consider a net N and ϕ a partial injection from $P_f^{N_1}$ to P_f^N and $p, q \in P_f^{N_1 \bowtie_\phi N}$, we need to prove that $(N_1 \bowtie_\phi N) \Downarrow_q^p \Leftrightarrow (N_2 \bowtie_\phi N) \Downarrow_q^p$.

By Theorem 246, we know that $[N_1 \bowtie_\phi N] = [N_2 \bowtie_\phi N]$. So, the result is given by Lemma 248. □

Theorem 250 (full abstraction)**.** *If* $P_f^{N_1} = P_f^{N_2}$ *and* $N_1 \approx N_2$*, then* $[N_1] = [N_2]$

Proof. Let us consider $\{(p, S), (q, V)\} \in [N_1]$, we will prove that $\{(p, S), (q, V)\} \in [N_2]$. We know that there exists $P \in Can^N$ and $T, U \in Tra^+$ such that $(P, []) \mapsto^* ([(p, N_1)], T)$, $(\overline{P}, []) \mapsto^* ([(q, N_1)], U)$ and $|S, T, U, V| \neq \varnothing$. We use Lemma 247 to reduce N_1 to a net N_1' such that the projection $|\Pi(P)| = 1$ and the paths $(\Pi(P), []) \mapsto^* ([(p, N_1)], T)$ and $(\overline{\Pi(P)}, []) \mapsto^* ([(q, N_1)], U)$ do not cross active pairs. So p —◁T U▷— q is a subterm of N_1'. We set $N = o$ —◁S▷— p' q' —◁V▷— r and $\phi = \{p \mapsto p', q \mapsto q'\}$. Then $N_1 \bowtie_\phi N$ reduces to a net which has o —◁S▷— —◁T U▷— —◁V▷— r as a subnet. We know that $|S, T, U, V| \neq \varnothing$ so $(N_1 \bowtie_\phi N) \Downarrow_r^o$. We know that $N_1 \approx N_2$ so $(N_2 \bowtie_\phi N) \Downarrow_r^o$. Thus, $N_2 \bowtie_\phi N$ reduces to a net N_2' with an observable path from o to r. We consider s the first port of the path which is not a principal port. Then $([(s, N_2')], []) \mapsto^* ([(o, N_2')], T_2)$ and $([(\overline{s}, N_2')], []) \mapsto^* ([(r, N_2')], U_2)$. By Lemma 231, we know that s is the projection of $Q \in Can^{N_2 \bowtie_\phi N}$ and that the paths exist in $N_2 \bowtie_\phi N$. Those paths begin in N_2 and end N, let us consider the traces T_2' and U_2' at the interfaces. Then, we have $(\overline{Q}, []) \mapsto^* ([(o, N_2 \bowtie_\phi N)], T_2')$, $(Q, []) \mapsto^* ([(r, N_2 \bowtie_\phi N)], U_2')$ and $|S, T_2', U_2', V'| \neq \varnothing$. □

6.5 Application on interaction combinators

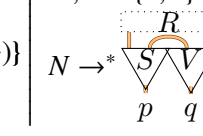
6.5.1 Comparison of $[\![\_]\!]$ and $[\_]$ on symmetric combinators

As we stated, our observational equivalence is strongly inspired by Mazza's equivalence [56]. If he defines it for any interaction net library, he only defines a sound and fully abstract semantics $[\![\_]\!]$ for symmetric combinators. The two equivalences coincide on symmetric combinators. In particular, $[\![N_1]\!] = [\![N_2]\!] \Leftrightarrow [N_1] = [N_2]$. Here, we will even see that the structures of those semantics are quite similar.

Mazza defines an arch for the interaction N as a multiset $\{(p, S_\delta, S_\zeta), (q, V_\delta, V_\zeta)\}$ where p, q are free ports of N, and $S_\delta, S_\zeta, V_\delta, V_\zeta \in \{1,2\}^{\mathbb{N}}$. We can notice that the shape is similar to our semantics, highlighted by the use of similar names for corresponding objects. One of the differences is that the information in the trace S is divided in a sequence S_δ corresponding to the δ cells and a sequence S_ζ corresponding to the ζ cells. The link is made more precise by the mappings $(\_)_\delta$ and $(\_)_\zeta$ from traces S to finite sequences on $\{1,2\}$, defined by induction on $|S|$: $[]_\delta = []_\zeta = []$, $(T.(\delta, i))_\zeta = T_\zeta$, $(T.(\zeta, i))_\delta = T_\delta$, $(T.(\delta, i))_\delta = T_\delta.i$ and $(T.(\zeta, i))_\zeta = T_\zeta.i$.

Let N be a net, the edifice of N is the set $\mathfrak{E}(N) =$

$$\left\{ \{(p, X@S_\delta, Y@S_\zeta), (q, X@V_\delta, Y@V_\zeta)\} \;\middle|\; \begin{array}{l} X, Y \in \{1,2\}^{\mathbb{N}} \\ N \to^* \end{array} \right\}$$



However, it is possible that nets are observationally equivalent but have different edifices. To be fully abstract, we will define a distance on arches and consider the metric completion of edifices. First, let us consider the usual distance on infinite sequences: if $S, V \in \{1,2\}^{\mathbb{N}}$, we define $d(S, V) = 2^{-k}$ where k is the length of the longest common prefix between S and V. On P_f^N, we will use the discrete topology: if $p = q$ then $d_{disc}(p, q) = 0$, else $d_{disc}(p, q) = 1$. We use those distances to define a distance on $P_f^N \times \{1,2\}^{\mathbb{N}} \times \{1,2\}^{\mathbb{N}}$:

$$d\left((p, S, S'), (q, V, V')\right) = \max\left\{d(S, V), d(S', V'), d_{disc}(p, q)\right\}$$

Finally, we can define a distance on arches. If $a = \{\mu, \mu'\}$ and $b = \{\nu, \nu'\}$, then

$$d(a, b) = \min\{d(\mu, \nu) + d(\mu', \nu'), d(\mu, \nu') + d(\mu', \nu)\}$$

In other words, as the pairs are unordered, we compare them in the two possible ways and we choose the best matching. Finally, we define $[\![N]\!]$ as the metric completion of $\mathfrak{E}(N)$.

Our semantics $[\_]$ is based on the $|\_, \_, \_, \_|$ function, we will study its behaviour on symmetric combinators. We denote the prefix order on sequences by $\leq$ (i.e. $T \leq U \Leftrightarrow \exists V, V@T = U$), and we define $\lesseqgtr$ as $\leq \cup \geq$. We also define $T - U$ as $T'@T - T = T'$ and otherwise $T - U = []$. Then, we can observe that for every $T, U \in Tra^+$,

$$|S, T, U, V| \neq \varnothing \Leftrightarrow \begin{cases} S_\delta \lesseqgtr T_\delta, U_\delta \lesseqgtr V_\delta, S_\delta - T_\delta \lesseqgtr V_\delta - U_\delta \\ S_\zeta \lesseqgtr T_\zeta, U_\zeta \lesseqgtr V_\zeta, S_\zeta - T_\zeta \lesseqgtr V_\zeta - U_\zeta \end{cases}$$

We can verify that $[N]$ is the set of prefixes of merging (meaning that the $\{1,2\}$ sequences for δ and ζ are merged into traces) of elements of $\mathfrak{E}(N)$:

$$[N] = \left\{ \{(p, S), (q, V)\} \;\middle|\; \begin{array}{l} \exists S', V' \in Tra^+, S \leq S', V \leq V' \\ \{(p, S'_\delta, S'_\zeta), (q, V'_\delta, V'_\zeta)\} \in \mathfrak{E}(N) \end{array} \right\}$$

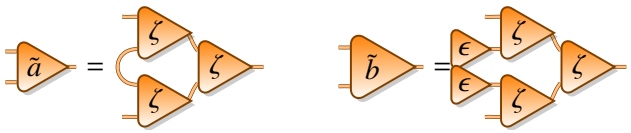
We wrote that one of the differences between $[N]$ and $[\![N]\!]$ is that in the first the information corresponding to δ and ζ are merged whereas they are separated in the second. On this point, Mazza's specialized semantics is better than our general semantics, because $[\![N]\!]$ is closer to full-completeness. Indeed, the structure of $[N]$ allows to have $\{(p,[(\delta,1);(\zeta,2)]),(q,[])\} \in [N]$ and $\{(p,[(\zeta,2);(\delta,1)]),(q,[])\} \notin [N]$ but the operational semantics of interaction combinators makes this impossible.

The second difference is that $[N]$ is defined by prefixes of archs, while $[\![N]\!]$ is defined by a metric completion. Thus, we noticed that if $E, F \subseteq \{1,2\}^{\mathbb{N}}$, then the completions of E and F are equal iff $\{S \mid \exists T \in E, S \leq T\} = \{S \mid \exists T \in F, S \leq T\}$. So we could interpret nets by the following semantics $\lfloor \_ \rfloor$ which is equivalent to $[\![\_]\!]$ but which we consider simpler to understand because it does not use metric completions.

$$\lfloor N \rfloor = \left\{ \{(p, S_1, S_2), (q, V_1, V_2)\} \;\middle|\; \begin{array}{c} X, Y \in \{1,2\}^{\mathbb{N}} \\ N \to^* \; R \;\; T \;\; U \;\; p \;\; q \\ S_1 \leq T_\delta @ X, S_2 \leq T_\zeta @ Y \\ V_1 \leq U_\delta @ X, V_2 \leq U_\zeta @ Y \end{array} \right\}$$

6.5.2 Comparison with semantics of encodings in symmetric combinators

As one can encode numerous libraries in symmetric combinators, one could try to define the semantics of a net N as $[\tilde{N}]$ with $\tilde{N}$ the encoding of N in interaction combinators. However, this semantics does not match $\approx$. Indeed, let us consider the following encoding of library L (of Figure 6.12) in interaction combinators.



We wrote that in the library L every pair of nets with the same number of free ports are equivalent. In particular, in L, $a \approx b$. However, $\{(p,[(\zeta,1)]),(p,[(\zeta,2)])\} \in [\tilde{a}]$ whereas $\{(p,[(\zeta,1)]),(p,[(\zeta,2)])\} \notin [\tilde{b}]$ so $\tilde{a} \not\approx \tilde{b}$ in L_{comb}. The difference is that, in L_{comb}, we can test nets with traces which do not exist in L.

Conclusion

This thesis is full of technical definitions and results, which sometimes hide the main ideas underlying them. This is why we thought it would be useful to sum up our main contributions, putting them in the perspective of our long term goal: a programming language certifying complexity bounds at compilation time. To go from the current characterizations of *Poly* to such a language, we have to solve several challenges:

1. Being more expressive: for most usual programs p terminating in polynomial time, the language should find a polynomial bounding the computation time of p.

2. Proving tighter bounds: bounding the complexity of the insertion sort by $|l|^{127}$ may be enough for a proof of *Ptime*-reduction. However, the concrete bound is not useful because it is too far from the actual bound.

3. Efficient type inferring: if the use of our language considerably slows down compilation time, it will only be used in applications where the certification of the bound is an important issue. It would rule out the third and fourth applications stated in the introduction (helping the programmer to find the origin of a slowness, and detecting errors/typos in the program).

4. Accomodate several complexity classes: inside a single software, one may need to know that the function f computes in logarithmic space, g computes in polynomial time and h computes in exponential time. Although one could imagine to use distinct type systems to test every complexity class, it would be preferable to have a single type-system characterizing several classes (for instance, with distinct class of types as in [13]).

Among those four challenges, we believe that the first is the most important. One argument is that, for some applications, the other challenges are secondary. If one wants to certify a proofs of *Ptime*-reduction (in computational complexity theory or cryptography, for instance) we only to infer a polynomial bound, the degree of the polynomial is not important. For such an application, one may tolerate a long compilation time, and characterizing only *Poly* is enough. With those first users, one could have feedback on our language and its limits.

Moreover, it seems that the 2nd, 3rd and 4th challenges heavily depend on the first. For instance, if we defined efficient type inferrence algorithms for $SDNLL$ and the expressive programming languages we obtain have nothing in common with $SDNLL$, those algorithms would be of no use. Thus, even if it is interesting to face those challenges in the case of landmark systems (such as *LLL*), we believe that spending too much time on the 2nd, 3rd and 4th challenges for the myriad of variations of subsystems would mainly be a loss of time.

Figure 6.14 sums up the method by which we hope to reach an expressive programming language characterizing polynomial time. We consider that this task can be divided in several subtasks: defining an expressive functional core (a linear logic subsystem or λ-calculus type system) and, in a second step, adding

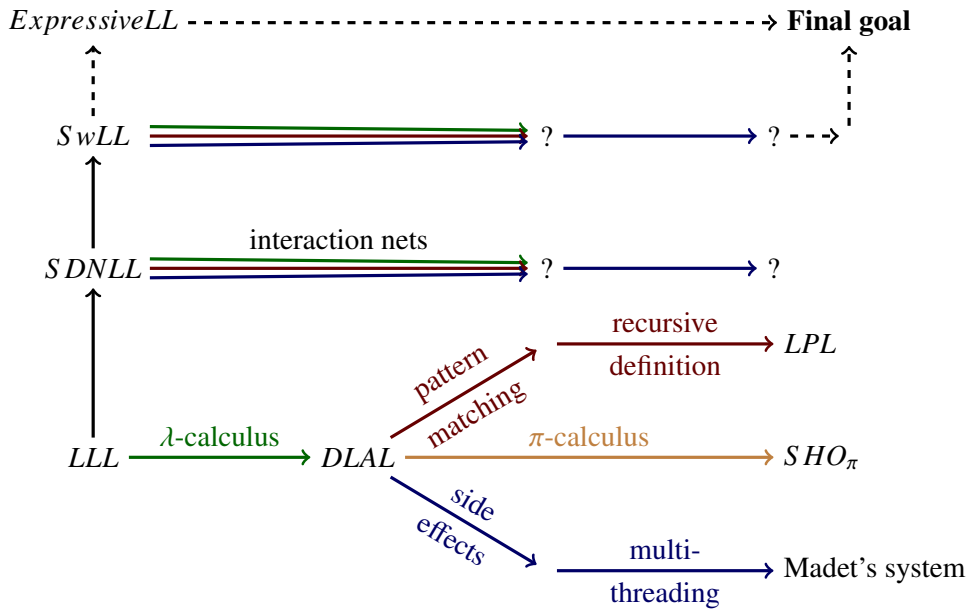


Figure 6.14: Path toward an expressive programming language

features. For example, previous works have added pattern-matching, recursive definitions, side-effects and concurrent features to *LLL*. However, even with those transformations, *LLL* seemed to lack expressive power. Thus, we believed that the most urgent direction was the "vertical" one: adding more expressive power to current linear logic subsystems.

With this thesis, not only did we define more expressive linear logic subsystems, but we defined abstract semantic criteria entailing polynomial bounds. Those criteria may be easier to transpose to another framework than the syntax of *SDNLL* and *SwLL* because they have a meaning which goes beyond linear logic itself: $B \twoheadrightarrow C$ means that the function C can be applied to a normal term containing B, $B \succ\!\!\!\rightarrow C$ means that C contains several occurrence of a variable which can be replaced by B along reduction, and $B \prec\!\!\!\triangleleft C$ means that a strict subterm of C uses B. Admittedly, those meanings are not very formal and, for instance, the difference between $\succ\!\!\!\rightarrow$ and $\succ\!\!\!\twoheadrightarrow$ may not have any correspondence in another framework. However, we think that the notions of stratification, dependence control and nesting may be relevent in other frameworks based on a local reduction. Thus rather than adapting the syntax of *SDNLL* and *SwLL* to richer languages, we think that we will adapt the criteria underlying them. And we believe this transformation to be easier in comparison.

We now think that the most urgent direction is the "horizontal" one (adding feature to our current criteria/subsystems).

- First, trying to type terms in our subsystems, we "feel"[3] that the main limit to expressivity is the lack of basic constructors (lists and natural numbers for instance). We hoped that, with our work on iterations of iterated functions (the w label in $SwLL_{strat}$), we could type most standard functions on lists (iter, fold, filter,...) with adequate type. However, we were limited by the representations of lists and natural numbers in linear logic. For instance 3 could be represented with 6 different proof-nets by reordering the contractions and the $\otimes$ nodes. Because of such possible reorderings, it seems to

[3] Admittedly, the justification is not based on formal arguments.

us that typing many standard functions on lists would require bijections between sets of copies of potential boxes. This technical difficulty only comes from the particular encoding of lists, which does not correspond to anything used in usual programming languages. Thus, rather than getting an even more expressive and complex type system of linear logic, we prefer to add basic constructors to our language.

- And, indeed, the system $SwLL$ is already quite complex. It contains a lot of different "features". But we are not even sure that all of them would be meaningful outside of linear logic. Trying to extend our language, we could get a better sense of which labels/rules are useful to be expressive in other languages, and which are not.

We are not sure that this functional core will be expressive enough to obtain (after transformation in a richer language) an expressive characterization of *Poly*. It might even be the case that, after discovering good ways to adapt linear logic subsystems into type systems for richer languages, we discover that *LLL* is enough as a functional core. In the absence of a good measure of intensional expressivity which could guide the research efforts, it seems that personal experience of typing terms in those languages is the best guide we have, to decide which direction is the most crucial for expressivity.

Bibliography

[1] K. Aehlig and H. Schwichtenberg. A syntactical analysis of non-size-increasing polynomial time computation. *ACM Transactions on Computational Logic (TOCL)*, 3(3), 2002.

[2] V. Atassi, P. Baillot, and K. Terui. Verification of ptime reducibility for system F terms: Type inference in dual light affine logic. *Logical Methods in Computer Science*, 3(4), 2007.

[3] P. Baillot, P. Coppola, and U. Dal Lago. Light logics and optimal reduction: Completeness and complexity. *Information and Computation*, 209(2), 2011.

[4] P. Baillot and U. Dal Lago. Higher-order interpretations and program complexity. In P. Cégielski and A. Durand, editors, *Computer Science Logic (CSL'12) - 26th International Workshop/21st Annual Conference of the EACSL, CSL 2012, September 3-6, 2012, Fontainebleau, France*, volume 16 of *LIPIcs*, pages 62–76. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.

[5] P. Baillot and U. Dal Lago. Higher-order interpretations and program complexity (long version). `http://hal.archives-ouvertes.fr/hal-00667816`, 2012.

[6] P. Baillot, U. Dal Lago, and J-Y. Moyen. On quasi-interpretations, blind abstractions and implicit complexity. *Mathematical Structures in Computer Science*, 22(04):549–580, 2012.

[7] P. Baillot, M. Gaboardi, and V. Mogbil. A polytime functional language from light linear logic. *Programming Languages and Systems*, 2010.

[8] P. Baillot and D. Mazza. Linear logic by levels and bounded time complexity. *Theoretical Computer Science*, 411(2), 2010.

[9] P. Baillot and V. Mogbil. Soft lambda-calculus: A language for polynomial time computation. In Igor Walukiewicz, editor, *Foundations of Software Science and Computation Structures, 7th International Conference, FOSSACS 2004, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2004, Barcelona, Spain, March 29 - April 2, 2004, Proceedings*, volume 2987 of *Lecture Notes in Computer Science*, pages 27–41. Springer, 2004.

[10] P. Baillot and M. Pedicini. Elementary complexity and geometry of interaction. *Fundamenta Informaticae*, 45(1-2), 2001.

[11] P. Baillot and K. Terui. A feasible algorithm for typing in elementary affine logic. In Pawel Urzyczyn, editor, *Typed Lambda Calculi and Applications, 7th International Conference, TLCA 2005, Nara, Japan, April 21-23, 2005, Proceedings*, volume 3461 of *Lecture Notes in Computer Science*, pages 55–70. Springer, 2005.

[12] P. Baillot and K. Terui. Light types for polynomial time computation in lambda calculus. *Information and Computation*, 207(1), 2009.

[13] Patrick Baillot, Erika De Benedetti, and Simona Ronchi Della Rocca. Characterizing polynomial and exponential complexity classes in elementary lambda-calculus. In Josep Diaz, Ivan Lanese, and Davide Sangiorgi, editors, *Theoretical Computer Science - 8th IFIP TC 1/WG 2.2 International Conference, TCS 2014, Rome, Italy, September 1-3, 2014. Proceedings*, volume 8705 of *Lecture Notes in Computer Science*, pages 151–163. Springer, 2014.

[14] Patrick Baillot and Kazushige Terui. Light types for polynomial time computation in lambda-calculus. *CoRR*, cs.LO/0402059, 2004.

[15] S. Bellantoni and S. Cook. A new recursion-theoretic characterization of the polytime functions. *Computational complexity*, 2(2), 1992.

[16] G. Bonfante, J-Y. Marion, and J-Y. Moyen. Quasi-interpretations a way to control resources. *Theor. Comput. Sci.*, 412(25):2776–2796, 2011.

[17] P. Coppola and S. Martini. Optimizing optimal reduction: A type inference algorithm for elementary affine logic. *ACM Trans. Comput. Log.*, 7(2):219–260, 2006.

[18] P. Coppola and S. Ronchi Della Rocca. Principal typing for lambda calculus in elementary affine logic. *Fundam. Inform.*, 65(1-2):87–112, 2005.

[19] U. Dal Lago. The geometry of linear higher-order recursion. In *Logic in Computer Science, 2005. LICS 2005. Proceedings. 20th Annual IEEE Symposium on*. IEEE, 2005.

[20] U. Dal Lago. Context semantics, linear logic, and computational complexity. *ACM Transactions on Computational Logic*, 10(4), 2009.

[21] U. Dal Lago and P. Di Giamberardino. Soft session types. In Bas Luttik and Frank Valencia, editors, *Proceedings 18th International Workshop on Expressiveness in Concurrency, EXPRESS 2011, Aachen, Germany, 5th September 2011.*, volume 64 of *EPTCS*, pages 59–73, 2011.

[22] U. Dal Lago and M. Gaboardi. Linear dependent types and relative completeness. In *Logic in Computer Science, 2011*. IEEE, 2011.

[23] U. Dal Lago, L. Roversi, and L. Vercelli. Taming modal impredicativity: Superlazy reduction. In Sergei N. Artëmov and Anil Nerode, editors, *Logical Foundations of Computer Science, International Symposium, LFCS 2009, Deerfield Beach, FL, USA, January 3-6, 2009. Proceedings*, volume 5407 of *Lecture Notes in Computer Science*, pages 137–151. Springer, 2009.

[24] V. Danos and J.B. Joinet. Linear logic and elementary time. *Information and Computation*, 183(1), 2003.

[25] V. Danos and L. Regnier. Proof-nets and the Hilbert space. *London Mathematical Society Lecture Note Series*, 1995.

[26] M. De Falco. An explicit framework for interaction nets. In *Rewriting Techniques and Applications*. Springer, 2009.

[27] M. De Falco. Géométrie de l'interaction et réseaux différentiels. *These de doctorat, Université Aix-Marseille*, 2, 2009.

[28] A. Dorman and D. Mazza. Linear logic by asymmetric levels. Unpublished note, 2009.

[29] M. Fernández and I. Mackie. Operational equivalence for interaction nets. *Theoretical Computer Science*, 297(1), 2003.

[30] M. Fernández, I. Mackie, S. Sato, and M. Walker. Recursive functions with pattern matching in interaction nets. *Electronic Notes in Theoretical Computer Science*, 253(4), 2009.

[31] M. Gaboardi, J-Y. Marion, and S. Ronchi Della Rocca. Soft linear logic and polynomial complexity classes. *Electr. Notes Theor. Comput. Sci.*, 205:67–87, 2008.

[32] M. Gaboardi, J-Y. Marion, and S. Ronchi Della Rocca. An implicit characterization of PSPACE. *ACM Trans. Comput. Log.*, 13(2):18, 2012.

[33] M. Gaboardi and S. Ronchi Della Rocca. A soft type assignment system for *lambda* -calculus. In Jacques Duparc and Thomas A. Henzinger, editors, *Computer Science Logic, 21st International Workshop, CSL 2007, 16th Annual Conference of the EACSL, Lausanne, Switzerland, September 11-15, 2007, Proceedings*, volume 4646 of *Lecture Notes in Computer Science*, pages 253–267. Springer, 2007.

[34] J-Y. Girard. Light linear logic. In Daniel Leivant, editor, *Logical and Computational Complexity. Selected Papers. Logic and Computational Complexity, International Workshop LCC '94, Indianapolis, Indiana, USA, 13-16 October 1994*, volume 960 of *Lecture Notes in Computer Science*, pages 145–176. Springer, 1994.

[35] J.Y. Girard. Une extension de l'interpretation de gödel a l'analyse, et son application a l'elimination des coupures dans l'analyse et la theorie des types. *Studies in Logic and the Foundations of Mathematics*, 63, 1971.

[36] J.Y. Girard. Linear logic. *Theoretical computer science*, 50(1), 1987.

[37] J.Y. Girard. Proof-nets: the parallel syntax for proof-theory. *Logic and Algebra*, 180, 1996.

[38] J.Y. Girard. Light linear logic. *Information and Computation*, 143(2), 1998.

[39] J.Y. Girard, A. Scedrov, and P.J. Scott. Bounded linear logic: a modular approach to polynomial-time computability. *Theoretical computer science*, 97(1), 1992.

[40] G. Gonthier, M. Abadi, and J.J. Lévy. The geometry of optimal lambda reduction. In *Proceedings of the 19th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*. ACM, 1992.

[41] G. Gonthier, M. Abadi, and J.J. Lévy. Linear logic without boxes. In *Logic in Computer Science, 1992. LICS'92., Proceedings of the Seventh Annual IEEE Symposium on*. IEEE, 1992.

[42] D. Hofbauer. Termination proofs by multiset path orderings imply primitive recursive derivation lengths. *Theoretical Computer Science*, 105(1):129–140, 1992.

[43] M. Hofmann. The strength of non-size increasing computation. In John Launchbury and John C. Mitchell, editors, *Conference Record of POPL 2002: The 29th SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Portland, OR, USA, January 16-18, 2002*, pages 260–269. ACM, 2002.

[44] M. Hofmann. Linear types and non-size-increasing polynomial time computation. *Information and Computation*, 183(1), 2003.

[45] Y Lafont. Interaction nets. In *Principles of programming languages, 17th ACM SIGPLAN-SIGACT symposium on*. ACM, 1989.

[46] Y. Lafont. Interaction combinators. *Information and Computation*, 137(1), 1997.

[47] Y. Lafont. Soft linear logic and polynomial time. *Theoretical Computer Science*, 318(1-2), 2004.

[48] U. Dal Lago, S. Martini, and D. Sangiorgi. Light logics and higher-order processes. In Sibylle B. Fröschle and Frank D. Valencia, editors, *Proceedings 17th International Workshop on Expressiveness in Concurrency, EXPRESS'10, Paris, France, August 30th, 2010.*, volume 41 of *EPTCS*, pages 46–60, 2010.

[49] O. Laurent. A token machine for full geometry of interaction. In *TLCA*, pages 283–297, 2001.

[50] D. Leivant. Ramified recurrence and computational complexity i: Word recurrence and poly-time. *Feasible Mathematics II*, 1994.

[51] S. Lippi. Encoding left reduction in the lambda-calculus with interaction nets. *Mathematical Structures in Computer Science*, 12(6), 2002.

[52] I. Mackie. The geometry of interaction machine. In Ron K. Cytron and Peter Lee, editors, *POPL*. ACM Press, 1995.

[53] I. Mackie and J.S. Pinto. Encoding linear logic with interaction combinators. *Information and Computation*, 176(2), 2002.

[54] Antoine Madet. A polynomial time λ-calculus with multithreading and side effects. In Danny De Schreye, Gerda Janssens, and Andy King, editors, *Principles and Practice of Declarative Programming, PPDP'12, Leuven, Belgium - September 19 - 21, 2012*, pages 55–66. ACM, 2012.

[55] D. Mazza. Multiport interaction nets and concurrency. In *CONCUR 2005–Concurrency Theory*. Springer, 2005.

[56] D. Mazza. Interaction nets: Semantics and concurrent extensions. *These de doctorat, Université Aix-Marseille II/Universita degli Studi Roma Tre*, 2006.

[57] D. Mazza. Observational equivalence and full abstraction in the symmetric interaction combinators. *Logical Methods in Computer Science*, 5(4:6), 2009.

[58] D. Mazza. Non-uniform polytime computation in the infinitary affine lambda-calculus. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part II*, volume 8573 of *Lecture Notes in Computer Science*, pages 305–317. Springer, 2014.

[59] D. Nowak and Y. Zhang. Formal security proofs with minimal fuss: Implicit computational complexity at work. *Information and Computation*, 2014.

[60] M. Perrinel. On paths-based criteria for polynomial time complexity in proof-nets. In U. Dal Lago and Peña R., editors, *FOPARA*, volume 8552 of *Lecture Notes in Computer Science*, pages 127–142. Springer, 2013.

[61] M. Perrinel. On context semantics and interaction nets. In T.A. Henzinger and D. Miller, editors, *CSL-LICS*, pages 73:1–73:10. ACM, 2014.

[62] J. Reynolds. Towards a theory of type structure. In *Programming Symposium*. Springer, 1974.

[63] L. Roversi and L. Vercelli. Some complexity and expressiveness results on multimodal and stratified proof nets. In *Types for Proofs and Programs*. Springer, 2009.

[64] L. Roversi and L. Vercelli. A local criterion for polynomial-time stratified computations. In *Foundational and Practical Aspects of Resource Analysis*. Springer, 2010.

[65] Y. Zhang. The computational SLR: A logic for reasoning about computational indistinguishability. In *Typed lambda calculi and applications: 9th international conference, TLCA 2009, Brasília, Brazil, July 1-3, 2009: proceedings*, volume 5608. Springer-Verlag New York Inc, 2009.

Appendix A

Notations

A.1 Arrows

- $\rightsquigarrow$: local relation on contexts. It is defined in page 20 and Figure 2.8.
- $\rightsquigarrow_S$ with S a subset of B_G is a restriction of $\rightsquigarrow$ which only enters boxes of S. It is defined in Definition 49 in page 61. If S is a subset of $BoxSig_G$, the notion is defined similarly in Definition 124 in page 124.
- $\hookrightarrow$: relation on contexts which makes a "jump" between an auxiliary door and a principal door of a box. It is defined in page 20 and Figure 2.8.
- $\mapsto$: relation on contexts, it is the union of $\rightsquigarrow$ and $\hookrightarrow$. It is defined in page 20.
- $\mapsto_S$ with S a subset of B_G is a restriction of $\mapsto$ which only makes $\hookrightarrow$ steps on boxes of S. It is defined in Definition 49 in page 61. If S is a subset of $BoxSig_G$, the notion is defined similarly in Definition 124 in page 124.
- $\rightarrow_\beta$ is the usual β reduction defined on λ-terms in page 41.
- $\twoheadrightarrow_\lambda$ is a stratification relation on λ-calculus defined in Definition 28 in page 41.
- $\twoheadrightarrow$ is a relation on boxes defined in Definition 30 in page 43. Intuitively $B \twoheadrightarrow C$ means that there exists a residue B' of B which is inside a residue C' of C. Formally, $B \twoheadrightarrow C$ iff there exists a path of the shape $((\sigma(B), P), [!_t]) \mapsto^* ((e, Q@R), T)$ with $e \in C$.
- $\dot{\rightsquigarrow}$ is a relation on potential boxes defined in Definition 48. We write $(B, P)\dot{\rightsquigarrow}(C, Q)$ if $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q), U)$.
- $\ddot{\rightsquigarrow}$ is a relation on $BoxSig_G$ defined in Definition 138 in page 138. We write $(B, P, t)\ddot{\rightsquigarrow}(C, Q, u)$ if $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q), U.?_u)$.
- $\rightarrow_{?N}$ is a reduction relation on proof-nets, moving $?N$ nodes around the proof-net to fix the mismatch between *cut*-elimination and β-reduction. It is defined in Figure 3.6 in page 48.
- $\succcurlyeq$ is a relation on boxes whose acyclicity is a dependence control condition. $B \succcurlyeq C$ if copies t and u of (B, P) lead to $((\sigma_i(C), \_), [!\_])$ and $((\sigma_j(C), \_), [!\_])$ with $i \neq j$. It is defined in Definition 80 in page 76.
- $\lessdot$ is a relation on boxes whose acyclicity is a nesting condition. $B \lessdot C$ if there is a strict simplification v of a copy of (B, P) such that $((\sigma(B), P), [!_v]) \mapsto^* ((\sigma(C), \_), [!_e])$. It is defined in Definition 83 in page 78.
- $\preccurlyeq$ is a relation on boxes whose acyclicity is a nesting condition. $Bs \preccurlyeq (C)$ iff there exist P, Q such that $(B, P)\dot{\preccurlyeq}(C, Q)$. It is defined in Definition 112 in page 102.
- $\dot{\preccurlyeq}$ is a relation on potential boxes. $(B, P)s \preccurlyeq (()C, Q)$ if there is a strict simplification v of two distinct copies of (B, P) such that $((\sigma(B), P), [!_v]) \mapsto^* ((\sigma(C), Q), [!_w])$ with w a standard signature. It is defined in Definition 112 in page 102.
- $\smile$ is an order on $BoxSig_G$ defined (in Definition 92 in page 92) by: $(B, P, t) \smile (C, Q, u)$ if $((\sigma(B), P), [!_t]) \mapsto^* ((\sigma(C), Q), [!_u])$.

- $\overset{\circ}{\mapsto}$ is a strict order on $BoxSig_G$ defined (in Definition 92 in page 92) as a shortcut for $\subseteq \smile$. The definition is extended to contexts in Definition 94 in page 94.
- $\hookrightarrow$ is a strict order on $BoxSig_G$ defined (in Definition 92 in page 92) as a shortcut for $\subseteq \curvearrowright$.
- $\boxdot\!\!\rightarrow$ is the second component of a weak stratification (which are defined in Definition 108 in page 97). It is a relation on $BoxSig_G$ representing additive dependences.
- $\overset{\cdot\cdot}{\rightsquigarrow}$: we write $(B, P, t)\overset{\cdot\cdot}{\rightsquigarrow}(C, Q, u)$ if there exists a path of the shape $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q), [!_v]@U.?_u)$. It is defined in Definition 120 in page 110.
- $\overset{\cdot\cdot}{\rightsquigarrow}$ is a relation on $BoxSig_G$, which is a refinement of $\overset{\cdot\cdot}{\rightsquigarrow}$. It is defined (in Definition 138 in page 119) by $(B, P, t)\overset{\cdot\cdot}{\rightsquigarrow}(C, Q, u)$ iff $((\sigma(B), P), [!_t]) \rightsquigarrow^* ((\overline{\sigma(C)}, Q), [!_v]@U.?_u) \rightsquigarrow^* ((\_, \_), [!_{v'}]@\_)$.
- $\overset{\cdot}{\rightsquigarrow}$ is the projection of $\overset{\cdot\cdot}{\rightsquigarrow}$ on $Pot(B_G)$. It is defined (in Definition 138 in page 119).
- $\rightarrow_m$ is the rewriting relation on interaction nets which deletes the merging ports. It is defined in page 170.
- $\leftrightarrows_m$ is the equivalence relation obtained from $\rightarrow_m$, viewing the nets up to merging of merging ports. It is defined in page 170.
- $\rightarrow$, when dealing with interaction nets, is the relation on nets generated by a library. It is defined in Definition 226 in page 171.
- $N\Downarrow_q^p$, with p and q two free ports of N, mean that there is an observable path from p to q. It is defined in Definition 237 in page 180.

A.2 Orders

- $\sqsubseteq$: cf. the definition of "simplification".
- $\sqsubset$: $t \sqsubset t'$ iff $t \sqsubseteq t'$ and $t \neq t'$. It is defined in page 23.
- $\blacktriangleleft$: cf. the definition of "truncation".
- $\lhd$: Let t, u be signatures, we write $t \lhd u$ if the rightmost difference between t and u is a branch where t is shorter than u. It is defined in Definition 55 in page 64.
- $\trianglelefteq$: Let t, u be signatures, we write $t \trianglelefteq u$ if $t \lhd u$ or $t = u$. It is defined in Definition 55 in page 64.
- $\leq_{lex}$, if A and B are two ordered sets, $\leq_{lex}$ is the order on $A \times B$ defined by $(a, b) \leq_{lex} (a', b')$ iff either $a < a'$ or ($a = a'$ and $b \leq b'$). It is defined in page 79.
- $\subseteq$: besides being the symbol for the usual inclusion of sets, we use the symbol to represent an order on $BoxSig_G$ defined (in Definition 92 page 85) by $(B, P, t) \subseteq (C, Q, u)$ if: either $((B, P) = (C, Q)$ and $t \sqsubseteq u)$ or ($B \subset C$ and there exists $v \sqsubseteq u$ such that $P = Q.v@\_$). The definition is extended to contexts in Definition 94 in page 94.
- $<$, is defined on the weights W^S as the lexicographic order (it is defined in Definition 150 in page 124).
- $\leq$, is also defined as a subtyping order on the formulae of $SDNLL$ in page 130.
- $<_\forall$ is an order on canonical $\forall$ and $\exists$ nodes, defined in Definition 156 in page 132. $(l, P) <_\forall (m, Q)$ means that there is a dependence sequence from (l, P) to (m, Q).
- $M \geq x$, with M a multiset, is defined (in page 142) as "for every y such that $M(y) > 0$, we have $y \geq x$".
- $M > y$, with M a multiset, is defined (in page 142) as "for every y such that $M(y) > 0$, we have $y > x$".
- $M \geqq y$, with M a multiset, is defined (in page 142) as "$M \geq x$ and $M(x) \leq 1$".

A.3 Letters

- $\underline{B}_{ELL}$ is a type encoding of binary lists in *ELL*. It is defined in page 46.
- $Cont_G$ is the set of contexts of G. It is defined in page 20.
- B_x, with x an occurrence of variable of the λ-term t, is the deepest box containing e_x in t^*. It is defined in page 48.
- d_x, with x an occurrence of variable of the λ-term t, is the conclusion of the $?D$ node associated to x in t^*. It is defined in page 48.
- D_G is the maximum number of doors of boxes of G. It is defined in page 16.
- $D_G(B)$ is the set of doors of B. It is defined in page 16.
- d_A, with A a formula of the shape $!_{s',d',n'}A'$ is the label d'. It is defined in page 130.
- $\mathfrak{d}(\_)$ is the fourth component of a weak stratification (which are defined in Definition 108 in page 97). It is a mapping from B_G to $\mathbb{N}$, representing the stratum of a box for dependence control.
- $d(\Gamma)$, with Γ a context in $Con_!$, is the multiset of d indices of Γ. It is defined in page 142.
- $\mathfrak{d}_{sw}(B)$, with B a box, is the d index of $\beta(\sigma(B))$ in $SwLL_{dc}$. It is defined similarly on contexts, using $\boldsymbol{\beta}(\_)$. They are both defined in Definition 189 in page 189.
- e_u, with u a subterm of the λ-term t, is the edge corresponding to u in u^*. It is defined in page 48.
- $e^S_{i,j}$, with S a coherent set and $i, j \in \mathbb{N}$, is a part of the weight W^S. It is the number of leaves (B, P), (C, Q) at respective depth i and j such that $((\sigma(B), P), [!\_]) \mapsto^* ((\sigma(C), Q), [!_{\mathtt{e}}])$. It is defined in Definition 150 in page 124.
- $E_\circ$ is a set of edges used to parameterize $SwLL_{last}$. It is first mentioned in page 152.
- $E_{!\circ}$ is a set of edges used to parameterize. It is first mentioned in page 152.
- $E_{\boxdot\to}$ is a set of edges used to parameterize $SwLL_{dc}$ and $SwLL_{strat}$. It is first mentioned in page 152.
- $\mathcal{F}_{\mathrm{LL}}$: designs the formulae of linear logic. It is defined in page 14.
- $\mathcal{F}_s$, with $s \in \mathbb{N}$ represents the set of formulae of $SDNLL$ whose s-labels are $\geq s$. It is defined in Definition 153 in page 130.
- $\mathcal{F}_{L4}$ is the set of formulae of L^4. It is defined in page 139.
- $\mathcal{F}^\lambda_s$ is the set of formulae for the intuitionnistic presentation of L^4. It is defined in Definition 175 in page 142.
- $\mathcal{G}_{L4}$ is a set of formulae, which are formulae of L^4 without their level index. It is defined in page 139.
- I_j is the name of the edge $\overline{\psi(i_j)}$ of the net of an interaction rule. It is defined in Definition 225 in page 171.

- O_j is the name of the edge $\psi(o_j)$ of the net of an interaction rule. It is defined in Definition 225 in page 171.
- L^3 is a subsystem of linear logic characterizing *Elem*. It is defined in [8].
- L^4 is a subsystem of linear logic characterizing *Poly*. It is defined in [8] and described in Section 5.1.4.1.
- M is used, in Section 3.5.1.3, for $\max_{(B,P)\in Pot(S)} |Cop(B,P)|$ (with S the first component of a weak stratification). It is defined in page 3.5.1.3.
- $\mathfrak{M}$ is used, in Section 3.5.1.3, for $\left(|\,\mathrm{E}_G\,| \cdot M^{\partial_G}\right)^{|\,\mathrm{E}_G\,|\cdot M^{\partial_G}}$. It is defined in page 3.5.1.3.
- N_G^X refers to the nodes of G whose label is X (page 16). The set of nodes of G is written N_G.
- $\underline{N}_{ELL}$ is a type encoding of natural numbers in *ELL*. It is defined in page 46.
- n_A, with A a formula of the shape $!_{s',d',n'}A'$ is the label n'. It is defined in page 130.
- $n(\Gamma)$, with Γ a context in $Con_!$, is the multiset of n indices of Γ. It is defined in page 142.
- $\mathfrak{n}_{sw}(B)$, with B a box, is the n index of $\beta(\sigma(B))$ in $SwLL_{nest}$. It is defined similarly on contexts, using $\boldsymbol{\beta}(\_)$. They are both defined in Definition 195 in page 195.
- P^N, with N a net, is the set of ports of N. It is defined in Definition 223 in page 169.
- P_c^N, with N a net, is the set of ports of N which are attached to a cell. It is defined in Definition 223 in page 169.
- P_f^N, with N a net, is the set of free ports of N (corresponding to pending edges of proof-nets). It is defined in Definition 223 in page 169.
- P_m^N, with N a net, is the set of merging ports of N. It is defined in Definition 223 in page 169.
- $p^i(c)$, with c a cell of a net, is a port attached to c. If $i = 0$, $p^i(c)$ is the principal port of c. If $i > 0$, $p^i(c)$ is the i-th auxiliary port of c. It is defined in page 170.
- $\mathfrak{R}_{s,t}$ is the net obtained by connecting the principal port of a cell labelled by s with the principal port of a cell labelled by t. It is defined in Definition 225 in page 171.
- S_n is a set representing the boxes whose stratum is at most n. In Section 3.2, it is defined as a set of boxes (Definition 71 in page 68) by $S_n = \{B \in B_G \mid s_{\rightsquigarrow}(B) \leq n\}$. In Section 4, it is defined as a subset of boxes (Definition 139 in page 119) by $S_n = \{B \in B_G \mid s_{\rightsquigarrow}(B) \leq n\}$. It is defined in Definition 71 in page 68.
- $s_>(\_)$ cf. the definition of >-stratum.
- s_A, with A a formula of the shape $!_{s',d',n'}A'$ is the label s'. It is defined in page 130.
- s_A^{min}, with A a formula, refers to the minimum s-label in A. It is defined in page 130.
- $\mathfrak{s}(\_)$ is the third component of a weak stratification (which are defined in Definition 108 in page 97). It is a mapping from B_G to $\mathbb{N}$, representing the stratum of a box for stratification.

- $s(\Gamma)$, with Γ a context in $Con_!$, is the multiset of s indices of Γ. It is defined in page 142.
- $S_{\boxdot\to}$ is a subset of B_G used to parameterize $SwLL_{dc}$ and $SwLL_{strat}$. It represents the boxes C such that there exists a pair $(\_, \_, \_) \boxdot\to (C, \_, \_)$. It is first mentioned in page 152.
- $\mathfrak{s}_{sw}(B)$, with B a box, is the s index of $\beta(\sigma(B))$ in $SwLL_{strat}$. It is defined similarly on contexts, using $\boldsymbol{\beta}(\_)$. They are both defined in Definition 207 in page 207.
- $T_{s,n}$ is a set of boxes, defined in page 79 for the definition of Lemma 85. The boxes of $T_{s,n}$ are the boxes B such that $(s_{\rightsquigarrow}(B), s_{\prec\!\!\!\cdot}(B)) \leq_{lex} (s, n)$.
- V_G: for any proof-net G, we define $V_G = \sum_{n\in \mathrm{N}_G} |Can(n)|$. It is defined in Definition 17 (page 32).
- W_G: for any proof-net G, we define $W_G = 2 \cdot \sum_{e\in \mathrm{E}_G} |Can(e)|$. It is defined in Definition 17 (page 32).
- W^S, with S a coherent set, is a tuple of natural numbers. The elements are defined by $e^S_{i,j}$. The main idea is that, if $\rightsquigarrow\!\!\!\!\!\!\ddot{}\!\!\!\!\!\!\to$ is acyclic, there exists an extension T of S, such that $W_S > W_T$. It is defined in Definition 150 in page 124.

A.4 Greek letters

- $\beta_G(e)$, if e is an edge of a *LL* proof-net, then $\beta_G(e)$ refers to the formula labelling e (Defined in Definition 1, page 14).
- $\beta(A, e, P, T, T')$ is the formula A, whose variables are substituted using the context $((e, P), T@T')$ It is defined in Definition 162 in page 135.
- $\boldsymbol{\beta}((e, P), [!_t]@T)$ is the formula corresponding to the context $((e, P), [!_t]@T)$. It is defined in Definition 165 in page 136.
- λ-calculus, the terms Λ of λ-calculus are defined (in page 41) by $\Lambda = x \mid \lambda x.\Lambda \mid (\Lambda)\Lambda$.
- $\pi_{G\to H}(\_)$. If $G \to_{cut} H$ then $\pi_{G\to H}(\_)$ is a mapping, from the contexts of H to the contexts of G. Intuitively $\pi_{G\to H}(C') = C$ means that C' is the residue of C. It is defined in Definition 12 in page 26.
- $\Pi(\_)$: if there is a reduction $N \to N'$ in an interaction net library, $\Pi(\_)$ is a projection from the potentials of N to the potentials of N'. It is defined in Definition 228 in page 176.
- $\phi_G(l, P)$, when (l, P) is a potential $\forall$ node, is the tuple $(A, (m, Q))$ with (m, Q) the potential $\exists$ node which will be cut with (l, P) and A the formula associated with l. If such a potential node does not exist, $\phi_G(l, P)$ is undefined. It is defined in page 132.
- ϕ_G^{lim} is the mapping from canonical $\forall$ nodes to formulae defined by composing ϕ_G. Thus, $\phi_G^{lim}(l, P) = A$ means that the variable associated to l will be replaced by A along reduction. It is defined in Definition 158 in page 134. It is extended to $\exists$ nodes in Definition 160 in page 134.
- $\rho_G(e)$, if e is an edge, $\rho_G(e)$ is the deepest box of G containing e. It is defined in page 16.
- $\rho_{G\to H}(\_)$. If $G \to_{?N} H$, then $\rho_{G\to H}(\_)$ is a mapping from the contexts of H to the contexts of G. Intuitively $\pi_{G\to H}(C') = C$ means that C' is the residue of C. It is defined in Lemma 38 in page 48.
- $\sigma(B)$: conclusion of the principal door of B, it is defined in page 16.
- $\sigma_i(B)$: conclusion of the i-th auxiliary door of B, it is defined in page 16.
- $\theta(e, P)$, with (e, P) a canonical edge, represents the substitution on the variables of $\beta(e)$ obtained by replacing every variable X by the formula replacing X along *cut*-elimination. It is defined in Definition 159 in page 134.

A.5 Words

- auxiliary door: node labelled by $?P$, it is defined in page 16.
- auxiliary edge: conclusion of an auxiliary door, it is defined in page 16.
- bouncing: a library is said bouncing if there is an interaction rule (R, ψ) and two free ports on the same side of R which can communicate. It is defined in page 182.
- box: set of nodes of proof-nets represented by rectangles. It is defined in page 16.
- box-bounded: a subsystem S of linear logic is box-bounded if the number of boxes in the proof-net representing the binary list l in S, does not depend on l. It is defined in Definition 37 in page 47.
- $BoxSig_G$: set of tuples (B, P, t) with (B, P) a potential box and t a signature. It is defined in page 85.
- $Can(x)$: cf the definition of canonical potential.
- Can^N: if N is a net, then Can^N represents the set of cells appearing during reduction. It is defined in Definition 233 in page 178.
- $CanCop_G$: set of tuples (B, P, t) with (B, P) a canonical box and $t \in Cop(B, P)$. It is defined in page 85.
- $Can_{\rightarrow}(x)$: cf the definition of $\rightarrow$-canonical potentials for x.
- canonical box: a tuple $(B, P) \in Can(B)$ with B a box. It is defined in page 26.
- canonical context: Intuitively, a context $((e, P), T)$ is canonical if (e, P) is a canonical edge and every signature of T corresponds to a copy. It is defined in Definition 22 in page 35.
- canonical edge: a tuple $(e, P) \in Can(e)$ with e an edge. It is defined in page 26.
- canonical node: a tuple $(n, P) \in Can(n)$ with n a node. It is defined in page 26.
- canonical potential: a canonical potential for x represents a residue of x. It is a pair (x, P) with P a potential composed of copies. It is defined in Definition 11 in page 26.
- $\rightarrow$-canonical potential: a $\rightarrow$-canonical potential for x (with $\rightarrow$ a relation on contexts) is a pair (x, P) with P a potential composed of $\rightarrow$-copies. It is defined in Definition 51 in page 62.
- characterize: a subsystem S of linear logic characterizes C if it is sound and complete for C. It is defined in Definition 36 in page 47.
- n-coherent: for $n \in \mathbb{N}$, a n-coherent set is a set of canonical boxes which is somehow "well-behaved" with respect to S_n. It is defined in Definition 149 in page 124.
- complete: a subsystem S of linear logic is complete for C if every function computable in time $c \in C$ is representable in S. It is defined in Definition 36 in page 47.
- Con_λ is the set of "contexts", with the meaning of a set of variables typed by formulae. It is defined in page 142.
- $Con_!$ is the subset of Con_λ where every formula is exponential. It is defined in page 142.

- $Con_{\S}$ is the subset of $Con_{\S}$ where every formula is linear. It is defined in page 142.
- $concl_n$: cf. the definition of "conclusion".
- conclusion: the conclusions of the node n refers to the outgoing edges of n. It is defined in page 16. If n has only one conclusion, $concl_n$ refers to the conclusion of n.
- context: A context of G is a tuple $((e, P), T)$ with (e, P) a potential edge of G and T a trace. The set of contexts is written $Cont_G$. It is defined in page 20.
- $Cop(B, P)$: cf. the definition of "copy".
- $Cop_{\rightarrow}(B, P)$: cf. the definition of "$\rightarrow$-copy".
- copy: a copy of a potential box (B, P) corresponds to residues of (B, P). It is a signature t such that for every simplification u of t, $((\sigma(B), P), [!_t]) \mapsto^* (\_, \_), [!_{\mathtt{e}}]@\_)$. It is defined in Definition 10 in page 26. More intuitions can be found in Section 2.2.1.
- $\rightarrow$-copy: for any potential box (B, P), the set $Cop_{\rightarrow}(B, P)$ of $\rightarrow$-copies of (B, P) is defined similarly to the set $Cop(B, P)$ of copies of (B, P), considering $\rightarrow$-paths instead of $\mapsto$-paths. It is defined in Definition 50 in page 61.
- copy contexts: a context $C = ((e, P), [!_t]@T)$ is a copy context if t is entirely used by $\mapsto$-paths beginning by C. It is defined in Definition 10 in page 26.
- $\rightarrow$-copy context: $\rightarrow$-copy contexts are defined similarly to copy-context, considering $\rightarrow$-paths instead of $\mapsto$-paths. It is defined in Definition 50 in page 61.
- crossing: a library is said crossing if it is not bouncing. It is defined in page 182.
- A $\forall/\exists$ dependence sequence is a sequence of canonical $\forall$ and $\exists$ nodes such that the $\forall$ nodes will be cut with the following $\exists$ node, and the formulae associated with the $\exists$ nodes have a free variable corresponding to the next $\forall$ node. It is defined in Definition 156 in page 132.
- *cut*-elimination is a relation on proof-nets defined in Figures 2.4 (page 15) and 2.5 (page 17).
- definitely enter: A path $C \mapsto^* D$ is said to *definitely enter* $(B, P, t) \in BoxSig_G$ if there exist $T.?_t \in Tra$ such that $C \mapsto^+ ((\overline{\sigma(B)}, P), T.?_t) \mapsto^* D$ and $((\overline{\sigma(B)}, P), T.?_t) \mapsto^* D$ stays in (B, P, t). Those notions are defined in Definition 102 in page 92.
- dependence: An additive (resp. affine, multiplicative) dependence of Q on R corresponds to bounds of the shape $Q \leq R + a$ (resp. $Q \leq b \cdot R + a$ and $Q \leq b \cdot R^c + a$). It is defined in page 54.
- eigenvariable: the eigenvariables of a proof-net are the variables which are replaced in a $\forall$ link. It is defined in page 16.
- *Elem* is the set of exponential towers. It is defined in page 47.
- Elementary stratification property: It is a property, defined in Property 47 in page 58, satisfied by $\dot{\twoheadrightarrow}$ and $\dot{\rightsquigarrow}$. Intuitively, if R satisfies this property, the acyclicity of R enforces an elementary bound.
- $Enc_B(l)$, with l a binary list, is an encoding of l which can be typed by $\underline{B}_{ELL}$. It is defined in Figure 3.4 in page 46.

- $Enc_N(n)$, with $n \in \mathbb{N}$, is an encoding of n which can be typed by $\underline{N}_{ELL}$. It is defined in Figure 3.4 in page 46.

- exponential signature: objects used to represent sequences of choices during a path. They are defined by $Sig = \mathsf{e} \mid \mathtt{l}(Sig) \mid \mathtt{r}(Sig) \mid \mathtt{p}(Sig) \mid \mathtt{n}(Sig, Sig)$. It is defined in page 19.

- gluing: If ϕ is a mapping from the free ports of M to the free ports of N, the gluing of M and N by ϕ is the net $M \bowtie_\phi N$ obtained by merging the ports of M with their corresponding ports by ϕ. It is defined in Definition 224 in page 171.

- head: the head of the edge (l, m) refers to m. It is defined in page 16.

- something-hole: If "something" is a set defined by induction, something-holes are subterms where subterms are replaced by $\circ$. It is defined in page 41.

- an interaction rule for (s, t) is a tuple (R, ψ) with R a net with the same free ports as $\Re_{s,t}$ (the correspondence is made explicit by ψ). It is defined in Definition 225 in page 171.

- leaves: A path of the shape $((\overline{\sigma(B)}, P), T.?_t) \mapsto^k C$, is said to leave (B, P, t) if there exist $j < k$, $e_1, \cdots, e_j \in \vec{\mathrm{E}}_G$ and $T' \in Tra$ such that $((\overline{\sigma(B)}, P), T.?_t) \mapsto^j ((\sigma(B), P), T'.!_t)$. It is defined in Definition 102 in page 92.

- library: a library of interaction nets is a partial mapping from pairs (s_1, s_2) of symbols to an interaction rule for (s_1, s_2). It is defined in Definition 226 in page 171.

- lift: We say that x is a lift of x if x' is a residue of x. It is defined in page 41.

- *LL*, stands for Linear Logic. In fact, it is an abuse of language because the system considered contains neither additives nor constant [36]. The acronym is defined in page 4, the system is defined by Figure 2.1 (page 14) and Definition 1 (page 14).

- maximal canonical edge: a canonical edge (e, P) is said maximal if there is no canonical edge (e, Q) with $Q \blacktriangleright P$. It is defined in Definition 26, in page 37.

- *MS* is a framework of subsystems of *LL* defined in [64] and briefly described in Section 5.1.4.2.

- MS_{max} is a maximal system of *MS* characterizing *Poly*. It is defined in Figure 5.6 in page 141.

- $\rightarrow$-maximal context: A context $((e, P), [!_t]@T)$ is said $\rightarrow$-maximal if the $\rightarrow$-paths beginning by contexts of the shape $((e, P), [!_u]@T)$ are not longer than the $\rightarrow$-path beginning by $((e, P), [!_t]@T)$. It is defined in Definition 25 in page 37.

- maximal: if R is an order, then a is maximal for R if there is no $b \neq a$ such that $a(R)b$. We establish this convention in page 40.

- maximal copy: for any potential box (B, P), a copy t of (B, P) is said maximal if there is no $u \in Cop(B, P)$ with $u \blacktriangleright t$. It is defined in Definition 26, in page 37.

- merging port: special kind of ports in interaction ports, corresponding to *cut* and *ax* in proof-nets. It is first described in page 169 and formally defined in the definition of nets (Definition 223).

- minimal: if R is an order, then a is minimal for R if there is no $b \neq a$ such that $b(R)a$. We establish this convention in page 40.
- $\mathrm{mix}(t, u)$, with $t, u \in Sig$ is defined (in Definition 125 in page 115) as the signatures obtained by considering the longest branches of t and u. The mapping is extended to potentials and traces by applying $\mathrm{mix}(\_, \_)$ on corresponding signatures.
- mixable: a subset S of $BoxSig_G$ is said mixable if it is stable by $\mathrm{mix}(\_, \_)$: if (B, P, t) and (B, Q, u) are in S, then $(B, \mathrm{mix}(P, Q), \mathrm{mix}(t, u))$ is in S. It is defined in Definition 129 in page 115.
- a net, is the notion of program in interaction nets. It is defined as a tuple $(P^N, C^N, l^N, \sigma_w^N, \sigma_m^N, \sigma_c^N)$ with P^N a set of ports, C^N a set of cells, l^N affecting a symbol to each cell, σ_w^N an involution on ports representing the wires, σ_m^N is an involution on ports representing merging ports, and σ_c^N binding the ports to cells. It is defined in Definition 223 in page 169.
- $\rightarrow$-normal context: A context $((e, P), [!_t]@T)$ is normal if there is no path of the shape $((e, P), [!_t]) \rightarrow^* ((\overline{f}, P), [!_u])$ with f the conclusion of a $?C$ or $?N$ node. It is defined in Definition 24 in page 24.
- observable path: if N is a net, an observable path is a sequence of ports of N. Intuitively, the $\mapsto$ paths correspond to the observable paths which can not be eliminated by reduction. It is defined in Definition 236 in page 180.
- pending edge: In a proof-net, a pending edge is an edge which has no conclusion.
- *Poly*: the set of polynomials, it is defined in page 54.
- *Pot*: cf. the definition of "potential".
- $Pot(x)$: if x is an element of a proof-net, then $Pot(x)$ is a pair (x, P) with P a potential and $|P| = \partial(x)$. It is defined in page 20.
- potential: list of signatures. The set of potentials is written *Pot*. It is defined in page 20.
- potential box: an element of $Pot(B)$ with B a box. It is defined in page 20.
- potential edge: an element of $Pot(e)$ with e an edge. It is defined in page 20.
- potential node: an element of $Pot(n)$ with n a node. It is defined in page 20.
- $prem_{\_}$: cf. the definition of "premise".
- premise: the premises of the node n refers to the incoming edges of l. It is defined in page 16. If n has only one premise, $prem_n$ refers to the premise of n.
- principal door: node labelled by $!P$, it is defined in page 16.
- principal edge: conclusion of a principal door, it is defined in page 16.
- quasi-standard: We first define the notion on signatures. A signature t is said quasi-standard if for every subtree $\mathfrak{n}(t_1, t_2)$ of t, the exponential signature t_2 is standard (definition 4, page 22). Then, we extend the definition on contexts. A context is said quasi-standard if every signature of the context is standard except (possibly) the signature of the leftmost trace element (Definition 6, page 23).

- $\rightarrow$-reducible context: A context $((e, P), [!_t]@T)$ is said reducible if there is a path of the shape $((e, P), [!_t]@T) \rightarrow^* ((\overline{f}, P), [!_u])$ with f the conclusion of a $?C$ or $?N$ node. It is defined in Definition 24 in page 37.

- $Repr(\_)$: for $(B, P, t) \in CanCop_G$, $Repr(B, P, t)$ is a set of simplifications u of t containing enough information to deduce t. It is defined in Definition 113 in page 103.

- residue: Let us suppose that there is a reduction from a term X to a term Y, and x is a subterm of X. The residues of x are the subterms y of Y which "come" from x. This notion is used for the proof-nets of linear logic (starting by an intuitive definition in page 18). If $G \rightarrow_{cut}^* H$, then $\pi_{G\rightarrow H}(\_)$ captures the notion of residues: B' is a residue of B iff $\pi_{G\rightarrow H}(((\sigma(B'), P'), [!_{t'}])) = ((\sigma(B), P), [!_t])$ for some $P, P' \in Pot$ and $t, t' \in Sig$. Finally, the notion of residues is used on λ-calculus in page 41.

- $Restr_{\rightarrow}((e, P), [!_t]@T)$ is a set of signatures. Precisely they are the truncations u of t such that $Restr_{\rightarrow}((e, P), [!_u]@T)$ is a $\rightarrow$-copy context.

- $\rightarrow$-copy restriction: Let $t \in Sig$, the $\rightarrow$-copy restriction of t for (B, P) is the maximum (for $\trianglelefteq$) element of $Restr_{\rightarrow}((\sigma(B), P), [!_t])$.

- $SDNLL$ is a subsystem of linear logic characterizing $Poly$ and defined in Figure 5.1 in page 131.

- $SDNLL_\lambda$ is a type system for λ-calculus characterizing $Poly$ and defined in Figure 5.7 in page 142.

- Sig: cf. the definition of "signature".

- simplification: We say that t' is a simplification of t (written $t \sqsubseteq t'$) if we can transform t into t' by transforming some of the subtrees $\mathtt{n}(t_1, t_2)$ of t into $\mathtt{p}(t_2)$. It is defined in page 23.

- smaller: if R is an order and $a(R)b$, then we say that a is smaller than b. We establish this convention in page 40.

- standard: We first define the notion on signatures. A signature t is said standard if it does not contain the constructor $\mathtt{p}(\_)$ (Definition 4, page 22). Then we extend the definition on contexts. A context is said standard if every signature of the context is standard (Definition 6, page 23).

- stays: A path of the shape $((\overline{\sigma(B)}, P), T.?_t) \mapsto^k C$, is said to stay in (B, P, t) if it does not leave (B, P, t).

- $>$-stratum, if $>$ is a relation on S and $e \in S$ then the $>$-stratum refers to the (possibly infinite) maximum length $s_>(e)$ of $>$ sequences starting from e. It is defined in page 40.

- subsystem: A subsystem S of Linear Logic is a tuple $(\mathcal{F}_S, \Pi_S, G_S, \underline{B}_S, Enc_\_(\_))$ with $\mathcal{F}_S$ representing the formulae of S, Π_S projecting those formulae on the formulae of LL, G_S the set of proof-nets of S, $\underline{B}_S$ represents the formulae encoding of binary lists, and $Enc_A(l)$ represents the encoding of the binary list l in A. It is defined in Definition 35 in page 35.

- $Succ$ refers to an encoding of the successor function $n \mapsto n{+}1$ in proof-nets. It is defined in Figure 3.4c in page 46.

- sound: a subsystem S of linear logic is sound for C if the proof-nets of S have a complexity c, with c a function of C. It is defined in Definition 36 in page 47.

- $SwLL_{dc}$ is a subsystem of linear logic, defined in Definition 188 in page 154 and used to define the part of $SwLL$ controlling dependence.
- $SwLL_{nest}$ is a subsystem of linear logic, defined in Definition 194 in page 156 and used to define the part of $SwLL$ enforcing nesting.
- $SwLL_{last}$ is a subsystem of linear logic, defined in Definition 200 in page 158 and used to ensure that some contexts are not used in $SwLL$.
- $SwLL_{strat}$ is a subsystem of linear logic, defined in Definition 205 in page 162 and used to ensure the existence of a weak stratification in $SwLL$.
- $SwLL$ is a subsystem of linear logic characterizing *Poly*. It is defined in Definition 211 in page 163.
- A symbol set is a tuple $S = (\mathcal{S}, \alpha)$ with $\mathcal{S}$ a countable set of symbols and $\alpha : \mathcal{S} \mapsto \mathbb{N}$ an arity function. It is defined in Definition 222 in page 169.
- tail: the tail of the edge (l, m) refers to l. It is defined in page 16.
- Tra: cf. the definition of "trace".
- trace: A trace is a non-empty list of trace elements. The set of traces is written Tra. It is defined in page 20.
- trace element: A trace element is one of the following: $\parr_l$, $\parr_r$, $\otimes_l$, $\otimes_r$, $\forall$, $\exists$, $!_t$ and $?_t$ (with t a signature). It is defined in page 20.
- truncation: Let $t, u \in Sig$, we say that t is a truncation of u (written $t \blacktriangleleft u$), if t can be obtained by u by replacing some subterms by Sig. It is defined in Definition 54 in page 63. The order is extended to potential in Definition 62 in page 65.
- used context: A context C is said to be used if there exists a path $D \mapsto^* C$ crossing a $?N$ or $?C$ node, using the leftmost trace element. It is defined in Definition 105, in page 93.
- weak stratification: A weak stratification is a tuple $(S, \boxdot\!\!\rightarrow, \mathfrak{s}(\_), \mathfrak{d}(\_))$ with S a subset of B_G, $\boxdot\!\!\rightarrow$ an acyclic relation on $BoxSig_G$, and $\mathfrak{s}(\_)$ and $\mathfrak{d}(\_)$ mappings from B_G to $\mathbb{N}$, satisfying the conditions of Definition 108 (page 97).

A.6 Exponents

- $(\_)^{\perp}$ is used to denote the "dual" of an object. It is first defined on variables in the definition of $\mathcal{F}_{\mathrm{LL}}$ (page 14). We extend it on formulae (page 14), trace element and traces (page 20).
- t^*, with t a λ-term is a proof-net corresponding to t (specifically the girard Encoding of t). It is defined in Figure 3.5 in page 47.
- $((e, P), [!_t]@T)^{/\rightarrow}$ is the maximum element (for $\trianglelefteq$) of $Restr_{\rightarrow}((e, P), [!_t]@T)$.
- $(e, P)^{\rightarrow}$ is the maximum $\rightarrow$-canonical potential (e, P') such that P' is a truncation of P. It is defined in Definition 61 in page 65.
- $((e, P), T)^{\rightarrow}$ is the maximum $\rightarrow$-canonical context $((e, P'), T')$ such that P' is a truncation of P and T' is a truncation of T. It is defined in Definition 66 in page 66.
- In several definition of exponents, a natural number n can be used instead of the relation $\mapsto_{S_n}$. For instance, if C is a context, C^n stands for $C^{\mapsto_{S_n}}$. Let us recall that the definition of $\mapsto_{S_n}$ is not the same in Section 4 and Section 3.2. Thus this shortcut is defined both in Definition 71 in page 68, and in page 119.
- $A_{|T}$, with A a formula and T a list of trace elements, is the formula obtained by pruning the syntactic tree of A using T. It is defined in Definition 155 in page 130.
- $\Gamma^{\varnothing}$, with Γ a context in $Con_!$, is the context in $Con_{\S}$ obtained by "linearizing" every formula. It is defined in page 142.
- $\Gamma^{s,d,n}$, with Γ a context in $Con_!$, is the context in $Con_!$ obtained by translating the labels by, respectively, s, d and n. It is defined in page 142.

A.7 Others

- $\overline{e}$, if e is a directed edge (l, m) then $\overline{e}$ refers to the inverted edge: (m, l). It is defined in page 16.
- $\partial(x)$, if x is an element of a proof-net, $\partial(x)$ refers to the number of boxes containing x. It is defined in page 16.
- $l_1 @ l_2$ is equal to the concatenation of the lists l_1 and l_2. It is defined in page 19.
- $l.x$, with l a list, is the list obtained by adding the element x on the right of l. It is defined in page 19.
- $|[a_1; \cdots ; a_k]|$ is equal to k, the number of elements of the list. It is defined in page 19.
- $|[a_1; \cdots ; a_k]|_X$ is the number of indices i such that a_i is in X. It is defined in page 19.
- $|T|_{!,?}$, with T a trace, refers to the number of $!_{\_}$ and $?_{\_}$ trace elements in T. It is defined in Definition 33 in page 45.
- $| \rightarrow |$ with $\rightarrow$ a relation on a set E refers to the maximum length of a $\rightarrow$ sequence. It is defined in page 40.
- $|t|$, with t a λ-term refers to the size of t. It is defined in page 41.
- $\|H\|$, with H a hole-formula, is the number of exponential connectives above $\circ$ in H. It is defined in page 139.
- $\approx$ is an (observational) equivalence relation on nets, defined (in Definition 238 in page 180) by $N_1 \approx N_2$ iff: in every context N, the ports connected by observable paths are the same in $N_1 \bowtie_\phi N$ and $N_2 \bowtie_\phi N$.
- $\simeq$ is an (observational) equivalence relation on nets, defined (in Definition 239 in page 181) by $N_1 \simeq N_2$ iff: in every context N, there exists an observable paths in $N_1 \bowtie_\phi N$ iff there exists one in $N_2 \bowtie_\phi N$.