HAL
open science

Vers un calcul des constructions pédagogique

Vincent Demange

To cite this version:

Vincent Demange. Vers un calcul des constructions pédagogique. Logique en informatique [cs.LO]. Université de Metz; Université de Lorraine, 2012. Français. NNT : . tel-00838147

HAL Id: tel-00838147

https://theses.hal.science/tel-00838147v1

Submitted on 24 Jun 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Département de formation doctorale en informatique **École doctorale IAEM Lorraine**
UFR MIM

Vers un calcul des constructions pédagogique

THÈSE

présentée et soutenue publiquement le 7 décembre 2012

pour l'obtention du

Doctorat de l'Université de Lorraine
(spécialité informatique)

par

Vincent Demange

Composition du jury

Président :	Patrick Cégielski	
Rapporteurs :	Gilles Dowek	Directeur de recherche, Inria, Paris
	Tristan Crolard	Professeur, Cnam, Paris
Examinateurs :	Patrick Cégielski	Professeur, LACL, Université Paris Est Créteil
	Jean-Yves Marion	Professeur, Loria, École des Mines de Nancy
Encadrants :	Loïc Colson	Professeur, LITA, Université de Lorraine
	Sorin Stratulat	Maître de conférences, LITA, Université de Lorraine

Laboratoire d'Informatique Théorique et Appliquée — EA 3097

Remerciements

« Une cause très petite, qui nous échappe, détermine un effet considérable que nous ne pouvons pas ne pas voir, et alors nous disons que cet effet est dû au hasard. »

Henri Poincaré

Cette remarque d'Henri Poincaré est tout à fait transposable à la difficulté d'accorder sa reconnaissance aux personnes impliquées, directement ou indirectement, à l'aboutissement d'un travail quelle que soit sa nature. Je tiens donc à m'excuser de ne pouvoir citer personnellement tous ceux, nombreux, qui le méritent : ils se reconnaîtront dans au moins l'une des catégories suivantes.

Je tiens tout d'abord à remercier mes directeurs de thèse, Loïc Colson et Sorin Stratulat, qui par leurs réponses pertinentes, leurs attentes exigeantes et motivées, leur suivi d'excellente qualité, et des réflexions d'ordre général et philosophique, m'ont toujours encouragé à me dépasser et à voir plus, savoir plus et surtout comprendre mieux.

J'exprime toute ma gratitude à Gilles Dowek et Tristan Crolard pour avoir rédigé des rapports bienveillants à propos de cette thèse, et pour le temps consacré à la lecture et la compréhension de ce document.

Je témoigne ma reconnaissance à Patrick Cégielski et Jean-Yves Marion d'avoir accepté de participer au jury de thèse.

Je n'oublie pas de remercier les collègues et amis enseignants et chercheurs, doctorants, jeunes docteurs, personnels administratifs et techniques de l'Université de Lorraine et des laboratoires d'informatique, de physique, de mécanique, de mathématiques, de sciences humaines que j'ai eu l'occasion de côtoyer pendant ces années de doctorat, de monitorat et d'ATER.

Je remercie mes récents collègues de l'ENSIIE d'Évry qui m'ont accueilli parmi eux depuis le début de cette année universitaire et m'ont offert leur soutien lors de la finalisation de cette thèse.

Et je termine par une pensée à toute ma famille et mes amis pour leur accompagnement, nécessaire et précieux, durant toutes ces années.

Table des matières

I Présentation informelle **1**

1 Introduction **3**

1.1 Les systèmes formels et fonctionnels pédagogiques 3

1.2 Organisation du rapport 4

1.3 Présentation du travail effectué 4

2 État du domaine **7**

2.1 Bref historique 7

2.2 Les mathématiques sans négation 8

2.3 Les mathématiques *pédagogiques* 9

2.4 Morphologie et syntaxe des systèmes formels 12

3 Description de l'étude formelle **17**

3.1 Présentation liminaire du formalisme 17

3.2 *Pédagogisme* dans le calcul des constructions 21

3.3 Sous-systèmes pédagogiques du second-ordre 27

3.4 Sous-système pédagogique d'ordre supérieur 36

4 Conclusion et perspectives **43**

4.1 Synthèse du travail effectué 43

4.2 Perspectives 44

4.3 Réflexions générales 48

II Étude formelle **51**

1 Présentation liminaire du formalisme **53**

1.1 La pré-morphologie 53

1.2 Résultats sur la pré-morphologie 56

1.3 Définitions pré-syntaxiques 59
1.4 Le calcul des constructions – CC 60

2 *Pédagogisme* dans le calcul des constructions 67
2.1 Critère de Poincaré et utilité 67
2.2 Un calcul des constructions Poincaréien – $\mathrm{CC_r}$ 68
2.3 Définition d'un sous-système pédagogique de CC 78

3 Sous-systèmes pédagogiques du second-ordre 79
3.1 Lambda-calcul polymorphe – λ^2 79
3.2 Avec motivations totales explicites – λ^2_e 83
3.3 Avec motivations totales – λ^2_t 97
3.4 Avec motivations partielles – λ^2_p 102
3.5 Système F pédagogique – P-Prop^2 106
3.6 Indécidabilité de la vérification du typage 115

4 Sous-système pédagogique d'ordre supérieur 119
4.1 Lambda-calcul d'ordre supérieur – λ^ω 119
4.2 Avec motivations totales explicites – λ^ω_e 122

III Annexes 131

A Quelques résultats d'intérêt 133
A.1 Avec motivations totales explicites – λ^2_e 133
A.2 Avec motivations totales – λ^2_t 134

Bibliographie 137

Première partie

Présentation informelle

Chapitre 1

Introduction

1.1 Les systèmes formels et fonctionnels pédagogiques **3**
1.2 Organisation du rapport **4**
1.3 Présentation du travail effectué **4**

1.1 Les systèmes formels et fonctionnels pédagogiques

Les systèmes pédagogiques sont apparus récemment à propos des calculs propositionnels (jusqu'à l'ordre supérieur), et consistent à donner systématiquement des exemples des notions (hypothèses) introduites. Formellement, cela revient à contraindre la règle d'hypothèse (Hyp) en déduction naturelle en une règle (P-Hyp) de la façon suivante :

$$\frac{F \in \Delta}{\Delta \vdash F}\,(\text{Hyp}) \qquad \frac{F \in \Delta \quad \vdash \sigma \cdot \Delta}{\Delta \vdash F}\,(\text{P-Hyp})$$

où σ est une substitution qui remplace les variables propositionnelles des formules de Δ par des exemples (i.e. des formules), et $\vdash \sigma \cdot \Delta$ dénote les dérivations des formules ainsi substituées. Autrement dit pour mettre un ensemble Γ de formules en hypothèse, il est requis de donner une substitution σ telle que l'instance $\sigma(\Gamma)$ soit démontrable.

Cette nécessité d'exemplification ayant été pointée du doigt par Poincaré (1913) comme relevant du bon sens : une définition d'un objet par postulat n'ayant d'intérêt que si un tel objet existe. Cette restriction appliquée à des systèmes formels intuitionnistes rejoint l'idée des mathématiques sans négation défendues par Griss (1946) au milieu du siècle dernier, et présentées comme une version radicale de l'intuitionnisme. À travers l'isomorphisme de Curry-Howard (1980), la contrepartie calculatoire est l'utilité des programmes définis dans les systèmes fonctionnels correspondant : toute fonction peut être appliquée à un argument clos.

Les premiers résultats concernant les calculs propositionnels jusqu'au second ordre ont été publiés récemment par Colson et Michel (2007, 2008, 2009).

Nous exposons dans ce rapport une tentative d'uniformisation et d'extension au Calcul des Constructions (CC) des précédents résultats. Tout d'abord une définition formelle et précise de *sous-système pédagogique du Calcul des Constructions* est introduite, puis différents tels sous-systèmes sont déclinés en exemple.

1.2 Organisation du rapport

Ce rapport est organisé en deux parties : une présentation informelle (part. I) et une étude formelle (part. II). L'étude formelle contient les définitions, les théorèmes et les preuves mathématiques. Dans la partie informelle, en plus de la présente introduction (ch. 1), on trouve un rapide état du domaine (ch. 2), suivi d'une description de l'étude formelle (ch. 3), et enfin une conclusion accompagnée de perspectives d'extensions ou d'améliorations de ces travaux (ch. 4). En annexe (part. III) nous avons souhaité donner quelques résultats d'intérêt, bien qu'inutilisés dans l'étude formelle.

Les titres des sections de la description informelle (ch. 3) correspondent aux titres des chapitres de l'étude formelle (part. II) : ainsi, à la lecture de la description informelle, le lecteur intéressé n'aura aucune difficulté à retrouver les définitions précises qui, pour des raisons de clarté et de concision, y seraient omises.

1.3 Présentation du travail effectué

Une définition formelle Après avoir rappelé et formalisé le critère de Poincaré (i.e. la nécessité d'exemplification) pour le Calcul des Constructions, nous commençons brutalement l'étude en proposant un système (supposé) pédagogique du Calcul des Constructions, que nous nommons $\mathrm{CC_r}$. Ce système est stable par réduction et satisfait le critère de Poincaré, caractérisation initialement supposée suffisante pour qualifier un système formel de *pédagogique*. Bien que son pouvoir calculatoire soit non négligeable (au moins comparable au système T de Gödel (1958)), son pouvoir expressif logique semble faible. Ce constat présuppose un manque dans la définition entendue d'un système pédagogique, et met en lumière une attente supplémentaire pour qu'un système puisse être qualifié de « pédagogique » : non seulement le critère de Poincaré doit être satisfait —c'est-à-dire que toute définition par postulat doit être *exemplifiable*— mais l'inverse aussi est souhaité. De cette analyse découle une définition formelle de *sous-système pédagogique du Calcul des Constructions*.

Des sous-systèmes pédagogiques du second ordre Pour illustrer cette définition, nous donnons trois formalismes différents d'un sous-système pédagogique du Calcul des Constructions, correspondant exactement au λ-calcul du second-ordre de Colson et Michel (2009).

Tout d'abord nous commençons par intégrer de façon explicite les exemples (aussi nommés *motivations*) dans le formalisme (système λ_e^2) : les jugements de typage sont alors de la forme $\Gamma \vdash_\sigma u : A$ où σ est une substitution contenant les exemples des variables de l'environnement de typage Γ. Toutefois ce système ne satisfait pas exactement notre définition de sous-système pédagogique, non seulement à cause de la forme différente des jugements, mais aussi car une proposition $\top$ et son exemple o doivent être ajoutés aux λ-termes, accompagnés d'une règle d'axiome idoine afin de pouvoir débuter une dérivation.

Nous proposons alors dans un premier temps une version relâchée (λ_t^2), où les motivations deviennent implicites mais restent totales : il est nécessaire de fournir des exemples pour toutes les variables des environnements Γ. Encore une fois, ce système n'est pas exactement un sous-système pédagogique, cette fois-ci uniquement à cause des constantes $\top$ et o.

Enfin, nous proposons un système avec motivations implicites et partielles : il suffit alors de motiver certaines variables de l'environnement. Dans ce cas les constantes $\top$ et o ne sont plus nécessaires, et le système résultant (λ_p^2) satisfait pleinement la définition de sous-système pédagogique du Calcul des Constructions.

Pour terminer, nous établissons l'équivalence des divers formalismes proposés (λ_e^2, λ_t^2 et λ_p^2), ainsi que du formalisme de Colson et Michel (2009). Nos systèmes sont ainsi des sous-systèmes pédagogiques correspondant à un λ-calcul du second ordre pédagogique. La définition de sous-systèmes pédagogique du Calcul des Constructions est ainsi éprouvée, et s'avère être un bon candidat pour l'obtention de systèmes pédagogiques plus expressifs.

Vérification du typage Malgré l'introduction explicite des exemples dans le formalisme de λ_e^2, la vérification du typage y reste indécidable. La résolution de ce problème est pourtant indispensable à une *implémentation* sur machine des systèmes pédagogiques. C'est pourquoi, dans cette éventualité, il sera nécessaire d'adopter un formalisme plus précis, que nous proposons comme perspective en fin de ce rapport.

Un sous-système pédagogique d'ordre supérieur Puis, nous proposons un sous-système pédagogique du Calcul des Constructions (λ_e^ω) correspondant à l'ordre supérieur. Ce système tente de tirer parti non seulement du formalisme de CC, mais aussi des notions obtenues au second ordre : nous réutilisons le concept d'utilité des fonctions pour définir la *motivabilité d'un prédicat d'ordre supérieur* (ou fonction propositionnelle). Nous choisissons de ne présenter qu'une version avec motivations explicites : celles-ci semblent plus naturelles et le formalisme associé autorise une analyse plus fine.

Dans ce système, le *lemme de substitution* permettant d'établir la stabilité par réduction (*subject reduction*) n'est plus valide et nous en conjecturons une version affaiblie. Nous montrons que cette version affaiblie est suffisante pour établir la stabilité par réduction.

Vers un calcul des constructions pédagogique Pour conclure, nous poursuivons nos idées conceptuelles vers un Calcul des Constructions pédagogique, dont nous exposons les règles de typage dans la conclusion. Ici aussi le lemme usuel de substitution n'est plus valide, et la réciproque du critère de Poincaré semble plus délicate à établir. C'est pourquoi nous laissons son étude totalement ouverte.

Chapitre 2

État du domaine

2.1 Bref historique **7**
2.2 Les mathématiques sans négation **8**
2.3 Les mathématiques *pédagogiques* **9**
2.3.1 Le critère de Poincaré informel 9
2.3.2 Résumé des précédents travaux sur la pédagogie 10
2.4 Morphologie et syntaxe des systèmes formels **12**
2.4.1 Le cas usuel 12
2.4.2 Le cas des λ-calculs typés 13
2.4.3 Le cas des systèmes pédagogiques 14
2.4.4 Les *pure type systems* 15
2.4.5 Les *pure type systems* et la pédagogie 16

2.1 Bref historique

Traditionnellement, on fait remonter la logique à Aristote ($\simeq$ -400) et sa syllogistique : « Tous les hommes sont mortels, or Socrate est un homme, donc Socrate est mortel. » On voit déjà l'émergence d'une étude *formelle* des raisonnements : c'est aux principes de raisonnements dont on s'intéresse, à l'étude des régularités. Ainsi « Tous les hommes sont des tortues, or Socrate est un homme, donc Socrate est une tortue. » est un raisonnement valide dont on peut abstraire les composants à l'aide de symboles pour lire « Tous les A sont B, or C est A, donc C est B. ». À cette époque l'activité du logicien consiste alors à recenser les raisonnements admis : il s'agit plutôt d'une démarche *descriptive*.

Il faudra attendre Leibniz, Frege et Peano ($\simeq$ 1900) pour voir apparaître des langages purement symboliques exprimant des théories mathématiques en codifiant, par des agencements rigoureux de symboles, les énoncés mathématiques mais aussi les raisonnements admissibles à l'intérieur de ces théories : la logique peut se prévaloir d'une dimension *normative* de l'activité mathématique. Refonder les mathématiques sur des concepts logiques primitifs, le *logicisme*, peut alors être pris au sérieux, et semble inévitable pour se prémunir des défauts d'une intuition submergée par un degré d'abstraction toujours grandissant en mathématiques. Mêler au *formalisme*, le logicisme peut être perçu comme une doctrine visant à traiter tout raisonnement mathématique comme un assemblage de symboles vidés de leur sens.

C'est d'ailleurs à ce formalisme auquel Brouwer s'oppose ($\simeq$ 1920) en proposant de refonder les mathématiques sur une nouvelle conception des objets et des preuves mathématiques :

l'*intuitionnisme*. Dans la philosophie intuitionniste, les preuves sont des constructions mentales, introduisant par là une notion de subjectivité (le mathématicien) et liant sujet et objet, et par conséquent une notion de temporalité. C'est d'ailleurs sur cette notion de temporalité qu'il base nombre de ses contre-exemples faibles (*weak counter-examples*) justifiant son rejet du tiers-exclu (pour tout énoncé A, « A ou non A » est *vrai*), dont l'interprétation intuitionniste serait que toute propriété est décidable. Effectivement, la *sémantique* intuitionniste d'une preuve d'une formule disjonctive « A ou B » est une preuve de A ou bien une preuve de B associée à une information indiquant de laquelle il s'agit. Mais pour Brouwer, opposé au formalisme, le langage ne peut traduire qu'imparfaitement sa philosophie et il ne développera pas une version formelle de son intuitionnisme. C'est Heyting ($\simeq$ 1930), élève de Brouwer, qui suggéra une formalisation de la logique intuitionniste, autorisant alors une étude *méta*-mathématique de ses conséquences.

Les preuves intuitionnistes, vues comme constructions mentales, sont résumées dans ce que l'on appelle l'interprétation de Brouwer-Heyting-Kolmogorov : par exemple une preuve d'une implication mathématique « A implique B » est une construction transformant toute preuve de A en une preuve de B ; une preuve d'une formule « A et B » est composée d'une preuve de A et d'une preuve de B ; une preuve d'une formule existentielle « il existe x tel que $A(x)$ » est composée d'un objet témoin t et d'une preuve de $A(t)$; une preuve de « non A » est une construction transformant toute preuve de A en une preuve de l'absurde ; etc.

2.2 Les mathématiques sans négation

C'est d'ailleurs ce dernier point, l'interprétation intuitionniste d'une preuve de « non A », que Griss ($\simeq$ 1940) récuse en particulier : l'hypothèse A menant à l'absurde ne peut que correspondre à une construction impossible. Or, pour Griss, les mathématiques intuitionnistes défendues par Brouwer ne font références qu'à des constructions mentales donc effectives. L'une des conséquences principale est l'abandon de la négation en mathématique : c'est le programme de *mathématiques sans négation* défendu par Griss et perçu alors comme une vision radicale de l'intuitionnisme de Brouwer.

Travaux informels Les travaux de Griss (1946, 1950, 1951a,b) constituent une ébauche informelle d'une géométrie, d'une arithmétique, d'une théorie des ensembles et d'une analyse sans négation. Griss substitua au concept négatif de différence celui, positif, de *discernabilité* (*apartness*), et nota une sémantique de l'implication différente dans les mathématiques sans négation : « A implique B » impose, en plus du sens intuitionniste, à A d'être *réalisée* par au moins un objet. Heyting (1955); Franchella (1994) résument les divergences de points de vue concernant l'intuitionnisme de Brouwer et de Griss, et les origines de la conception de Griss.

Travaux formels S'ensuivent plusieurs développements formels des *desiderata* de Griss, dont on peut citer les travaux de Vredenduin (1953); Gilmore (1953); Valpola (1955); Nelson (1966, 1973); Minichiello (1969); López-Escobar (1972, 1974); Mezhlumbekova (1975) et plus récemment de Krivtsov (2000a,b), traitant de logique des prédicats (et parfois d'une arithmétique) sans négation formalisée dans des systèmes de déduction naturelle pour certains, dans des calculs des séquents pour d'autres. Les idées principales à retenir sont : l'introduction d'un symbole d'implication quantifiée $A(\vec{x}) \rightarrow_{\vec{x}} B(\vec{x})$ dont l'inteprétation en logique intuitionniste est $\forall \vec{x}.A(\vec{x}) \rightarrow B(\vec{x}) \wedge \exists \vec{x}.A(\vec{x})$; la non-nullité des implications où $A \rightarrow B$ est nulle si A est *fausse* pour toutes les instances ; la possibilité de démontrer la clôture existentielle des sous-formules d'une formule démontrable dans un système sans négation ; la traduction des systèmes usuels

dans les systèmes de base. Souvent les formalisations sont lourdes, et aucun n'étudie leur praticabilité. Mints (2006) propose un survol de ces travaux.

À l'instar des arguments de Brouwer pour défendre l'intuitionnisme face au *classicisme*, ceux de Griss sont paradoxalement emprunt de négativité : là ou Brouwer justifiait le retrait du tiers-exclu, Griss veut légitimer la suppression de la négation.

Récemment une autre conception est apparue, basée sur la nécessité de fournir des exemples des notions manipulées. Appliquée aux formalismes de logique intuitionniste, cela rejoint les arguments de Griss : les exemples ont pour rôle de réaliser toute sous-formule, et toute forme de négation disparaît nécessairement.

2.3 Les mathématiques *pédagogiques*

2.3.1 Le critère de Poincaré informel

Dans Poincaré (1913), alors que l'auteur disserte des divergences entre les mathématiciens *Pragmatistes* et les *Cantoriens*, il en vient à formuler un principe d'utilité évidente à propos des définitions par postulats :

> *« Mais nous avons encore une autre sorte de définitions, les définitions par postulats ; généralement nous saurons que l'objet à définir appartient à un genre, mais quand il s'agira d'énoncer la différence spécifique, on ne l'énoncera pas directement, mais à l'aide d'un « postulat » auquel l'objet défini devra satisfaire. C'est ainsi que les mathématiciens peuvent définir une quantité x par une équation explicite $x = f(y)$, ou par une équation implicite $F(x, y) = 0$.*
>
> **La définition par postulat n'a de valeur que quand on a démontré l'existence de l'objet défini** *; dans le langage mathématique, cela veut dire que le postulat n'implique pas contradiction ; on n'a pas le droit de négliger cette condition ; il faut ou bien admettre l'absence de contradiction comme une vérité intuitive, comme un axiome, par une sorte d'acte de foi ; mais alors il faut se rendre compte de ce qu'on fait et savoir qu'on a allongé la liste des axiomes indémontrables ; ou bien il faut construire une démonstration en règle, soit* **par l'exemple**, *soit par l'emploi du raisonnement par récurrence. Ce n'est pas que cette démonstration soit moins nécessaire quand il s'agit d'une définition directe, mais elle est généralement plus facile. »*

On en retiendra ce qu'on désignera sous le nom de **critère de Poincaré** :

> *Si une définition par postulat est utilisée, alors il doit exister un exemple d'objet la satisfaisant.*

C'est la caractérisation originelle de la *pédagogie* employée dans les précédents travaux sur les systèmes formels et fonctionnels pédagogiques [1]. Ces systèmes sont développés dans la thèse de Michel (2008) et leur étude est publiée par Colson et Michel (2007, 2008, 2009).

1. Dans les travaux précédents, le critère de Poincaré est appelé *non-nullité syntaxique des jugements* en référence à la non-nullité syntaxique des implications des travaux de Nelson (1966) sur la formalisation des mathématiques sans négation de Griss.

2.3.2 Résumé des précédents travaux sur la pédagogie

La contrainte pédagogique formelle

Dans les calculs propositionnels, un ensemble de formules Δ peut être vu comme un ensemble de définitions par postulats des variables libres de Δ. Par exemple $\Delta := \{(\alpha \to \alpha) \to \alpha, \top \to \beta\}$ définit deux propositions α et β (ici $\top$ est une constante) dont la première satisfait la propriété $(\alpha \to \alpha) \to \alpha$ tandis que la seconde satisfait la propriété $\top \to \beta$. On dira alors que Δ est *exemplifiable* si on peut trouver un exemple de formule A pour α et un exemple de formule B pour β : c'est-à-dire telles que $(A \to A) \to A$ et $\top \to B$ soient « validées ».

Les travaux précédents concernant les systèmes formels pédagogiques systématisent la nécessité du critère de Poincaré : tous les ensembles d'hypothèses (*contextes*) utilisés dans les démonstrations sont vus comme des définitions par postulats et devront alors être exemplifiables. Pour y parvenir, une **contrainte pédagogique** est introduite dans ces systèmes propositionnels de déduction naturelle : tout contexte utilisé doit être *exemplifié a priori*, nous dirons qu'il est *motivé*. Par exemple la règle usuelle d'utilisation d'une hypothèse (Hyp) est contrainte en (P-Hyp) :

$$\frac{F \in \Delta}{\Delta \vdash F}\,(\text{Hyp}) \qquad \frac{F \in \Delta \quad \vdash \sigma \cdot \Delta}{\Delta \vdash F}\,(\text{P-Hyp})$$

où σ est une substitution qui remplace les variables propositionnelles des formules de Δ par des exemples (i.e. des formules), et $\vdash \sigma \cdot \Delta$ dénote les dérivations des formules ainsi substituées. De plus, et afin de démarrer une dérivation, il convient d'ajouter une règle sans prémisse (P-Ax) ainsi qu'un exemple canonique $\top$:

$$\frac{\vdash \sigma \cdot \Delta}{\Delta \vdash \top}\,(\text{P-Ax})$$

Une telle contrainte impose à un ensemble d'hypothèses Δ d'être motivable (par la substitution σ) avant de pouvoir être utilisé. Le critère de Poincaré est alors directement satisfait : tout contexte Δ utilisable (i.e. $\Delta \vdash F$) peut être exemplifié (i.e. $\vdash \sigma \cdot \Delta$). Comme illustration, reprenons l'ensemble de formules Δ précédent : $\Delta := \{(\alpha \to \alpha) \to \alpha, \top \to \beta\}$ est motivable par la substitution $\sigma := [\alpha, \beta \leftarrow \top, \top]$ puisqu'on peut dériver $\vdash \sigma \cdot \Delta$, c'est-à-dire $\vdash (\top \to \top) \to \top$ et $\vdash \top \to \top$.

Remarquons que la motivation est effectuée à l'intérieur même du système formel, ce qui a pour conséquence d'introduire une dépendance de la morphologie envers la syntaxe (voir sec. 2.4). Insistons aussi sur l'aspect global, ou homogène, de l'exemplification : il s'agit de motiver toutes les formules de Δ par un même jeu d'exemples représenté par la substitution (ou *motivation*) σ. En effet l'énoncé informel du critère de Poincaré ne donne aucune indication concernant ces deux points : rien n'est dit sur le lieu de la démonstration d'« existence de l'objet défini » ; ni sur le traitement de multiples définitions par postulats simultanées.

D'un point de vue général, la contrainte pédagogique est la contrepartie formelle à la pratique informelle courante d'enseignement des mathématiques qui consiste à donner des exemples des objets manipulés, permettant ainsi à l'élève d'appréhender de façon concrète de nouveaux concepts. C'est ce qui explique la terminologie de *pédagogie* employée.

Le calcul propositionnel pédagogique minimal

Dans Colson et Michel (2007) le calcul propositionnel minimal sur les connecteurs $\to$, $\vee$ et $\wedge$ (MPC) est restreint par la contrainte pédagogique : la règle usuelle (Hyp) est remplacée par (P-Hyp) et la règle (P-Ax) est ajoutée (voir ci-avant). Il est prouvé que le calcul résultant (P-MPC)

est équivalent au calcul originel MPC : un jugement $\Gamma \vdash F$ est dérivable dans le calcul usuel MPC si et seulement si il est dérivable dans sa version pédagogique P-MPC. Notons qu'aucune traduction des formules de MPC vers P-MPC n'intervient : le calcul propositionnel minimal peut être qualifié d'intrinsèquement pédagogique.

L'ajout d'une constante représentant l'absurde $\bot$ avec sa règle associée, intuitionniste $(\bot_i)$ ou classique $(\bot_c)$, n'augmente pas le pouvoir expressif du calcul. La constante $\bot$ se comporte alors comme une variable propositionnelle, en particulier aucune des règles $(\bot_i)$ ou $(\bot_c)$ ne peut être déclenchée (tout contexte est cohérent). Par une analyse plus précise, on constate que la constante $\bot$ peut apparaître dans les dérivations mais uniquement dans des sous-formules de la forme $\bot \vee A$ ou $A \vee \bot$ à cause des règles d'introduction de la disjonction $\vee$ qui ne sont pas contraintes dans P-MPC et autorisent l'introduction de formules non motivables.

Le calcul propositionnel pédagogique du second ordre

Le cas du calcul propositionnel du second ordre Prop^2 est étudié dans Colson et Michel (2008). Avec (P-Ax) et (P-Hyp) pour seules règles contraintes, on aboutit à un calcul du second ordre *faiblement pédagogique* P$_s$-Prop^2, dans lequel les règles régissant la quantification du second ordre sont inchangées :

$$\frac{\Delta \vdash F \quad \alpha \notin \mathcal{V}(F)}{\Delta \vdash \forall\alpha.F}\,(\forall_i) \qquad \frac{\Delta \vdash \forall\alpha.F}{\Delta \vdash F[\alpha \leftarrow U]}\,(\forall_e)$$

Ce calcul vérifie le critère de Poincaré : si $\Delta \vdash F$ alors $\vdash \sigma \cdot \Delta$. En revanche, il n'est pas *stable par normalisation*. En effet, $\bot \to \bot$ est dérivable dans P$_s$-Prop^2 (où $\bot := \forall\alpha.\alpha$ est la définition usuelle de l'absurde dans Prop^2) :

$$\begin{array}{ll} 1.\ \beta \vdash \beta & (\beta \text{ est motivable par } \top) \\ 2.\ \vdash \beta \to \beta & (\to_i\ 1) \\ 3.\ \vdash \forall\beta.\beta \to \beta & (\forall_i\ 2) \\ 4.\ \vdash \bot \to \bot & (\forall_e\ 3) \end{array}$$

Or une forme normale de cette preuve doit terminer par une règle d'introduction $(\to_i)$ avec $\bot$ dans le contexte, ce qui est impossible par le critère de Poincaré puisque $\bot$ n'est pas motivable. La forme normale de cette preuve n'est donc pas une preuve de P$_s$-Prop^2.

La formule non motivable $\bot$ est introduite dans la dérivation à l'étape 4, lors de l'utilisation de la règle non contrainte $(\forall_e)$. Ce constat motive la définition d'un système restreint P-Prop^2 par l'addition d'une contrainte supplémentaire sur la règle $(\forall_e)$ qui devient (P-$\forall_e$) :

$$\frac{\Delta \vdash \forall\alpha.F \quad \vdash \sigma \cdot U}{\Delta \vdash F[\alpha \leftarrow U]}\,(\text{P-}\forall_e)$$

P-Prop^2 est stable par normalisation des preuves. Aussi, le codage imprédicatif usuel des connecteurs $\vee$ et $\wedge$ est possible, à ceci près que les règles dérivées d'introduction de la disjonction (p. ex. à droite) font apparaître la nécessité d'une contrainte :

$$\frac{\Delta \vdash A \quad \vdash \sigma \cdot B}{\Delta \vdash A \vee B}\,(\vee_{ir})$$

Le résultat principal concernant P-Prop^2 est la construction d'une traduction $F \mapsto F^\gamma$ inspirée de la A-traduction de Friedman (1978) — dans laquelle les formules atomiques ϕ sont remplacées par $\phi \vee A$ — telle que : $\Gamma \vdash F$ est dérivable dans Prop^2 si et seulement si $\Gamma^\gamma \vdash F^\gamma$ est dérivable dans P-Prop^2.

Le λ-calcul pédagogique du second-ordre

La correspondance de Curry-Howard (1980) permet de transférer les résultats précédents à propos du calcul propositionnel pédagogique du second-ordre P-Prop^2 à un λ-calcul pédagogique polymorphe étudié par Colson et Michel (2009) : motiver une formule revient alors à *habiter* l'instance du type correspondant. Le système obtenu est stable par réduction : si $\Delta \vdash t : A$ et $t \rightsquigarrow_\beta t'$, alors $\Delta \vdash t' : A$.

Une notion importante pour un λ-calcul est alors révélée : l'*utilité des fonctions*. C'est la contrepartie calculatoire du critère de Poincaré. Cela signifie que toute fonction typable dans ce λ-calcul pédagogique peut être appliquée à un argument : si $\vdash f : A \to B$ avec A clos, alors A est habité. Autrement dit, toute fonction décrit un algorithme pouvant être effectivement déclenché.

Le calcul propositionnel pédagogique d'ordre supérieur

Dans sa thèse Michel (2008) propose une version pédagogique du calcul propositionnel d'ordre supérieur, dont une description et une analyse approfondie se trouvent en section 3.4 de ce rapport. L'introduction d'objets d'ordre supérieur nécessite en effet une attention particulière, et les choix de contraintes pédagogiques que nous faisons ici diffèrent de ceux précédemment effectués.

La pédagogie formelle

Ces travaux éclaircissent l'idée de systèmes formels et fonctionnels pédagogiques : on constate que le critère de Poincaré, point de départ, est la caractéristique principale d'un système formel *pédagogique*. Mais ce n'est le seul : au second ordre, un système satisfaisant le critère de Poincaré mais non stable par normalisation des preuves est proposé. Il est alors qualifié de *faiblement pédagogique* et est jugé insatisfaisant, à raison puisque sa version fonctionnelle n'est pas stable par réduction. Pourtant à ce stade de l'investigation, aucune définition formelle et rigoureuse d'un système pédagogique n'est encore clairement mise au jour.

Néanmoins, d'importantes notions sont dégagées : le critère de Poincaré s'applique systématiquement à tout contexte ; l'exemplification est effectuée à l'intérieur même du système formel ; les programmes fonctionnels construits sont utiles ; et enfin il apparaît possible de définir une traduction du système usuel dans sa version pédagogique.

Les deux premières de ces notions —à savoir l'exemplification interne et systématique aux systèmes formels pédagogiques— introduisent une dépendance de la *morphologie* de ces systèmes envers leur *syntaxe*. Cette dépendance apparaît déjà dans certains formalismes, sur lesquels nous avons décidé de nous appuyer en vue d'homogénéiser et d'étendre l'étude de cette expression formelle de la pédagogie : les *pure type systems*.

2.4 Morphologie et syntaxe des systèmes formels

2.4.1 Le cas usuel

Par analogie linguistique, Andler et Martin (1984) distinguent différents constituants d'un système formel :

- la morphologie : les règles de formation des mots/formules du système ;
- la sémantique : l'interprétation des mots/formules du système (non développée ici) ;
- la syntaxe : les règles de formation des phrases autorisées/preuves du système.

Immédiatement, on anticipe une dépendance nécessaire de la syntaxe envers la morphologie puisque toute phrase/preuve est composée de mots/formules. Confirmons cette intuition et concrétisons ces différents constituants à travers l'exemple de l'*arithmétique de Peano* que l'on formalise traditionnellement de la façon suivante :

Morphologie

Termes
- 0 est un terme ;
- si t est un terme, alors $S(t)$ est un terme ;
- les variables $x, y, z, \ldots$ sont des termes ;
- si t et u sont des termes, alors $t + u$ et $t \times u$ sont des termes.

Formules
- si t et u sont des termes, alors $t = u$ est une formule (formule atomique) ;
- si A et B sont des formules, alors $A \to B$ est une formule ;
- si A est une formule, alors $\forall x A$ est une formule ;
- si A est une formule, alors $\neg A$ est une formule.

Syntaxe

Axiomes

$$x = x \qquad x = y \to y = x \qquad x = y \to y = z \to x = z$$

$$x = y \to S(x) = S(y) \qquad S(x) = S(y) \to x = y \qquad \neg(S(x) = 0)$$

$$x = x' \to y = y' \to x + y = x' + y' \qquad 0 + y = y \qquad S(x) + y = S(x + y)$$

$$x = x' \to y = y' \to x \times y = x' \times y' \qquad 0 \times x = 0 \qquad S(x) \times y = y + x \times y$$

$$\neg\neg A \to A$$

Règles d'inférence

$$\frac{}{\Gamma \vdash A}\, ax \qquad \frac{}{\Gamma, A \vdash A}\, hyp \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B} \to_i \qquad \frac{\Gamma \vdash A \to B \qquad \Gamma \vdash A}{\Gamma \vdash B} \to_e$$

$$\frac{\Gamma \vdash A \qquad x \notin \mathcal{V}(\Gamma)}{\Gamma \vdash \forall x A} \forall_i \qquad \frac{\Gamma \vdash \forall x A}{\Gamma \vdash A[x \leftarrow t]} \forall_e$$

$$\frac{\Gamma \vdash P[x \leftarrow 0] \quad \Gamma \vdash \forall x (P \to P[x \leftarrow S(x)])}{\Gamma \vdash \forall x P}\, rec$$

Notons que les définitions de substitution et d'occurrences libres/liées utilisées dans les règles d'inférences (p. ex. $\forall_e$) et omises ici, font partie de la morphologie.

Les ensembles d'hypothèses Γ des règles d'inférence sont des ensembles de formules quelconques. Cela signifie que toute formule peut y apparaître. **On n'observe donc aucune dépendance de la morphologie envers la syntaxe** : la morphologie peut être définie indépendamment de la syntaxe.

2.4.2 Le cas des λ-calculs typés

Dans les λ-calculs typés, les *types* jouent le rôle de formules, les λ-*termes* celui de preuves, et les règles d'inférence sont des *règles de typage* : c'est la correspondance dite de Curry-Howard

(1980). Les notions de morphologie et de syntaxe s'y étendent alors immédiatement : ainsi, en tant que preuves, les λ-termes relèvent de la syntaxe.

Le λ-calcul simplement typé

Les présentations habituelles (à la Church) du λ-calcul simplement typé font apparaître une notion de *pré-syntaxe* : on commence par définir les types (la morphologie), puis des λ-termes typés *primitifs* (la pré-syntaxe), et enfin les règles de typage (la syntaxe). Les règles de typage ont alors pour rôle de sélectionner un sous-ensemble des termes typés primitifs : les λ-termes typés *bien formés*.

Remarquons qu'introduire une notion de pré-syntaxe n'est en fait utile qu'à la définition opératoire (β-réduction) des λ-termes en tant que dénotation d'un processus de calcul. On peut tout à fait s'en dispenser : il est possible de définir les λ-termes typés bien formés en même temps que les règles de typage de la syntaxe, puis *a posteriori* les règles de calcul associées à ces λ-termes.

Dans tous les cas, la définition des types (formules) précède la définition des λ-termes typés bien formés (preuves).

Le λ-calcul typé d'ordre supérieur

L'utilisation d'objets d'ordre supérieur nécessite l'introduction de types d'ordre supérieur. Dans ces types, on retrouve les notions d'abstraction, d'application et de calcul, et leur définition requiert alors des règles de typage. Le système formel est ainsi défini par deux ensembles distincts de règles de typage : celles intervenant dans la formation des types, et celles intervenant dans la construction des λ-termes typés d'ordre supérieur. Les règles de typage régissant la formation des types (la morphologie) ne dépendent pas de celles régissant la construction des λ-termes (la syntaxe).

Pour résumer, chez les λ-calculs typant des logiques propositionnelles (jusqu'à l'ordre supérieur), on n'observe toujours aucune dépendance de la morphologie envers la syntaxe.

Le cas des λ-calculs avec types dépendants

Les λ-calculs avec types dépendants sont plus intéressants : les types dépendent de λ-termes représentant des preuves. La définition des types (ou formules) dépend ainsi de la définition des preuves qui relève de la syntaxe. La morphologie n'est alors plus définissable en tant qu'entité autonome : **il y a dépendance de la morphologie envers la syntaxe**.

Notons que si la notion de substitution n'est pas internalisée dans la syntaxe, il est nécessaire de définir une *pré-morphologie* englobant une pré-syntaxe. En effet la définition de la substitution, qui intervient dans les règles de typage de la syntaxe, exige alors une notion de *pré-termes* : les *termes bruts*. Ces termes bruts contiennent les types et les λ-termes et constituent ainsi à la fois une pré-morphologie et une pré-syntaxe (pour un exemple voir sec. 1.1).

2.4.3 Le cas des systèmes pédagogiques

Cette interdépendance morphologie-syntaxe peut aussi être observée dans les systèmes pédagogiques. En effet, toute formule apparaissant dans une dérivation doit être préalablement motivée :

$$\frac{F \in \Delta \qquad \vdash \sigma \cdot \Delta}{\Delta \vdash F}\ \text{(P-Hyp)}$$

Or cette motivation σ des formules de Δ est contrôlée par les dérivations $\vdash \sigma \cdot \Delta$ qui se font dans le système formel lui-même et selon les règles d'inférence de la syntaxe. Les formules bien formées (la morphologie) —celles qui peuvent apparaître dans les dérivations— dépendent alors des règles d'inférence du système formel (la syntaxe).

En général tout système pédagogique, évitant les *formules vides*, doit nécessairement faire dépendre sa morphologie de sa syntaxe.

2.4.4 Les *pure type systems*

Les λ-calculs typés associés aux logiques propositionnelles jusqu'à l'ordre supérieur (système F^ω de Girard (1972)) et les λ-calculs avec types dépendants peuvent être décrits uniformément par une famille de systèmes fonctionnels que sont les *pure type systems* (PTS) (voir Barendregt (1992)).

Dans les PTS, la définition des types bien formés est *internalisée* dans la syntaxe (présentée ci-après [2]) en particulier par la règle (prod) de construction des types :

$$\frac{}{[\,]\,\mathsf{wf}}\,(\mathrm{env}_1) \qquad \frac{\Gamma \vdash A : s \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}}\,(\mathrm{env}_2)$$

$$\frac{\Gamma\,\mathsf{wf} \quad (s_1, s_2) \in \mathcal{A}}{\Gamma \vdash s_1 : s_2}\,(\mathrm{ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}}{\Gamma, x : A, \Gamma' \vdash x : A}\,(\mathrm{var})$$

$$\frac{\Gamma \vdash A : s_1 \quad \Gamma, x : A \vdash u : B \quad \Gamma, x : A \vdash B : s_2 \quad (s_1, s_2, s) \in \mathcal{R}}{\Gamma \vdash \lambda x^A.u : \forall x^A.B}\,(\mathrm{abs})$$

$$\frac{\Gamma \vdash A : s_1 \quad \Gamma, x : A \vdash B : s_2 \quad (s_1, s_2, s_3) \in \mathcal{R}}{\Gamma \vdash \forall x^A.B : s_3}\,(\mathrm{prod})$$

$$\frac{\Gamma \vdash u : \forall x^A.B \quad \Gamma \vdash v : A}{\Gamma \vdash u\ v : B[x \leftarrow v\,]}\,(\mathrm{app}) \qquad \frac{\Gamma \vdash u : A \quad A =_\beta A' \quad \Gamma \vdash A' : s}{\Gamma \vdash u : A'}\,(\mathrm{conv})$$

où s_1, s_2, s_3 et s désignent des sortes quelconques d'un ensemble donné $\mathcal{S}$, $\mathcal{A}$ (les axiomes) est une relation binaire sur $\mathcal{S}$, et $\mathcal{R}$ est une relation ternaire sur $\mathcal{S}$. Ce sont les ensembles $\mathcal{S}$, $\mathcal{A}$ et $\mathcal{R}$ qui déterminent l'instance du système formel généré par ces règles. Par exemple avec deux sortes possibles $\mathcal{S} := \{\star, \Box\}$ et une seule règle d'axiome $\mathcal{A} := \{(\star, \Box)\}$, on obtient les calculs importants du λ-cube de Barendregt (1991) en faisant varier la relation $\mathcal{R}$:

(s_1, s_2, s_3)	$\lambda \rightarrow$	λ^2	λ^ω	$\lambda\Pi$	λC
$(\star, \star, \star)$	✓	✓	✓	✓	✓
$(\Box, \star, \star)$		✓	✓		✓
$(\star, \Box, \Box)$				✓	✓
$(\Box, \Box, \Box)$			✓		✓

2. La formalisation proposée ici est similaire à celle effectuée dans Dowek (1991) et diverge légèrement de celle de Barendregt (1992) : ici il n'y a pas de règle d'affaiblissement, et deux types de jugements sont présents ($\Gamma\,\mathsf{wf}$ et $\Gamma \vdash A : B$).

En particulier le système fonctionnel λC, qui correspond au calcul des constructions (CC) de Coquand (1985), contient toutes les règles $\mathcal{R}$ des autres calculs $\lambda \to$ (λ-calcul simplement typé), λ^2 (système F), λ^ω (système F$^\omega$) et $\lambda\Pi$ (un λ-calcul simple avec types dépendants).

La définition des PTS fait apparaître une dépendance de la morphologie envers la syntaxe, et ceci de façon uniforme pour tous les calculs qu'elle permet de décrire. Ainsi à travers une syntaxe moins libérale, le système λ^2 correspond au système F de Girard (1972) (voir aussi Girard *et al.* (1990)).

2.4.5 Les *pure type systems* et la pédagogie

La *pédagogisation* du système F a déjà été étudiée par Colson et Michel (2009), mais les contraintes pédagogiques sont nombreuses :

$$\frac{\vdash \sigma \cdot \Delta}{\Delta \vdash o : \top}\ (\text{P-Ax}) \qquad \frac{x : F \in \Delta \quad \vdash \sigma \cdot \Delta}{\Delta \vdash x : F}\ (\text{P-Hyp}) \qquad \frac{\Delta \vdash t : \forall \alpha . F \quad \vdash \sigma \cdot U}{\Delta \vdash t\ U : F[\alpha \leftarrow U]}\ (\text{P-}\forall_e)$$

Pourtant ces contraintes se résument simplement : tout type introduit doit être motivé.

Dans λ^2 on peut profiter de l'interdépendance morphologie-syntaxe du symbolisme des PTS pour exprimer de façon plus concise toutes ces contraintes pédagogiques en restreignant uniquement la règle (prod) de formation des types :

$$\frac{\Gamma, x : A \vdash^{2_{\mathrm{P}}} B : \star \quad \sigma\ \widetilde{\mathrm{mot}}_\Gamma\ \forall x^A . B}{\Gamma \vdash^{2_{\mathrm{P}}} \forall x^A . B : \star}\ (\text{p-prod})$$

où $\sigma\ \widetilde{\mathrm{mot}}_\Gamma\ \forall x^A . B$ signifie que σ *motive partiellement* le type $\forall x^A . B$ sous l'environnement de typage Γ (voir les explications détaillées en ssec. 3.3.3 ci-après).

Chapitre 3

Description de l'étude formelle

3.1 Présentation liminaire du formalisme 17
3.1.1 Origines 17
3.1.2 Le calcul des constructions – CC 18
3.2 *Pédagogisme* dans le calcul des constructions 21
3.2.1 Critère de Poincaré formel 21
3.2.2 Extension naïve des résultats précédents à CC 22
3.2.3 Un calcul des constructions Poincaréien – $\mathrm{CC_r}$ 24
3.2.4 Définition d'un sous-système pédagogique de CC 26
3.3 Sous-systèmes pédagogiques du second-ordre 27
3.3.1 Avec motivations totales explicites – λ_e^2 27
3.3.2 Avec motivations totales – λ_t^2 30
3.3.3 Avec motivations partielles – λ_p^2 31
3.3.4 Système F pédagogique – P-Prop^2 33
3.3.5 Indécidabilité de la vérification du typage 34
3.3.6 Remarques d'ordre général 35
3.4 Sous-système pédagogique d'ordre supérieur 36
3.4.1 Un calcul propositionnel d'ordre supérieur pédagogique – P-CP$^\omega$. 36
3.4.2 Avec motivations totales explicites – λ_e^ω 38

3.1 Présentation liminaire du formalisme

3.1.1 Origines

Le calcul des constructions CC (où CoC pour *Calculus of Constructions*) est un système fonctionnel introduit par Coquand (1985) dans sa thèse. Il est présenté par son auteur comme une synthèse des systèmes du projet *Automath* de de Bruijn (1970) et des *calculs propositionnels d'ordre supérieurs* de Girard (1972).

Automath

Automath est l'une des premières implémentations sur machine de vérification de mathématiques formalisées, apparue à la fin des années 1960. Il s'agit d'un langage de définition d'éléments

mathématiques —axiomes, énoncés de théorèmes, preuves— ainsi que d'un programme vérifiant la bonne interaction de ceux-ci.

Dans Automath, la formalisation d'une théorie est représentée par un livre (*book*) qui est composé d'une succession de lignes. Chaque ligne définit ou introduit un nouvel élément (ou objet) d'un certain type. Toute définition est établie à partir d'un ensemble d'objets précédemment définis : un contexte. Les définitions peuvent être des *notions primitives* —axiome, hypothèse, nouveau type d'objets (p. ex. entiers, réels, etc) ou simple variable— ou l'application d'un élément à d'autres. Quand le contexte d'un objet comporte des notions primitives, elles agissent comme des objets indéfinis qui peuvent être remplacés par des objets concrets de même type. De tels objets *à trous* se comportent et sont utilisés comme des fonctions.

Automath traite de façon uniforme des conceptions jusqu'alors distinguées : l'implication comme cas particulier de quantification universelle ; les objets mathématiques (entiers, réels, etc.) et les preuves ; et l'identification des types et des formules. De fait, Automath constitue l'une des premières utilisations concrète et effective de la correspondance de Curry-Howard. Cette correspondance y est d'ailleurs étendue par la notion de *types dépendants* : des objets-preuves pouvant apparaître dans les types.

Pour plus de détails, nous suggérons de se référer à de Bruijn (1980) qui propose une vue d'ensemble (*survey*) du projet Automath, et van Daalen (1994) qui donne une définition formelle du représentant nommé AUT-QE de la famille des systèmes Automath, et établit quelques *méta*-théorèmes usuels à son sujet. Aussi Sørensen et Urzyczyn (2006) donnent des indications plus précises sur les origines de la correspondance de Curry-(De Bruijn)-Howard et la contribution apportée par les idées présentes dans Automath.

Calculs d'ordre supérieur

Les λ-calculs typés d'ordre supérieurs ont été développés par Girard (1972) afin d'établir des *interprétations fonctionnelles* des arithmétiques d'ordre supérieur. À l'origine introduite par Gödel (1958), l'interprétation fonctionnelle de l'arithmétique (intuitionniste) de Heyting HA (ou interprétation Dialectica) consiste à assigner à une formule de l'arithmétique A une formule $\exists\vec{x}\forall\vec{y}A_D(\vec{x},\vec{y})$, où A_D est une formule sans quantificateur d'un système de *fonctionnelles de type fini* nommé T : si A est prouvable dans HA, son interprétation l'est dans T. Ce système T peut être vu comme une extension des fonctions primitives récursives par l'ajout de fonctionnelles (voir sec. 2.2.2), et ainsi comme une extension du point de vue finitiste développé par Hilbert dans l'espoir de donner un fondement définitif, minimal et intuitif aux mathématiques. En particulier la cohérence de l'arithmétique est ramenée à une propriété calculatoire du système T. Avigad et Feferman (1998) dressent un éventail des conséquences de ce théorème d'aspect technique, dont les extensions de Girard menant aux calculs d'ordre supérieur.

3.1.2 Le calcul des constructions – CC

Notons que notre présentation est quelque peu anachronique : le calcul des constructions apparaît historiquement avant les *pure type systems* présentés précédemment (voir ssec. 2.4.4). Cependant le symbolisme contemporain utilisé pour décrire CC est plutôt celui des PTS, et c'est pourquoi nous avons préféré opter pour une présentation de CC en tant que membre de la famille des PTS. Toutefois, nous avons supprimé les prémisses redondantes de certains règles, p. ex. pour la règle (prod) des PTS :

$$\frac{\Gamma \vdash A : s_1 \quad \Gamma, x : A \vdash B : s_2 \quad (s_1, s_2, s_3) \in \mathcal{R}}{\Gamma \vdash \forall x^A.B : s_3}\ (\text{prod})$$

dans le Calcul des Constructions CC la prémisse $\Gamma \vdash A : s_1$ est inutile et donc supprimée.

Dans la suite nous utilisons les conventions mentionnées dans la première définition de la section 1.4, en particulier sur les jugements de la forme $\Gamma \vdash A : B : C$, qu'ils se trouvent en prémisse de règles de typage ou en conclusion.

Règles du calcul des constructions

$$\frac{}{[\,]\,\mathsf{wf}^{\mathsf{c}}}\ (\text{c-env}_1) \qquad \frac{\Gamma \vDash^{\mathsf{c}} A : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^{\mathsf{c}}}\ (\text{c-env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^{\mathsf{c}}}{\Gamma \vDash^{\mathsf{c}} \mathrm{Prop} : \mathrm{Type}}\ (\text{c-ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^{\mathsf{c}}}{\Gamma, x : A, \Gamma' \vDash^{\mathsf{c}} x : A}\ (\text{c-var})$$

$$\frac{\Gamma, x : A \vDash^{\mathsf{c}} u : B : \kappa}{\Gamma \vDash^{\mathsf{c}} \lambda x^A.u : \forall x^A.B}\ (\text{c-abs}) \qquad \frac{\Gamma \vDash^{\mathsf{c}} u : \forall x^A.B \quad \Gamma \vDash^{\mathsf{c}} v : A}{\Gamma \vDash^{\mathsf{c}} u\ v : B[x \leftarrow v\,]}\ (\text{c-app})$$

$$\frac{\Gamma, x : A \vDash^{\mathsf{c}} B : \kappa}{\Gamma \vDash^{\mathsf{c}} \forall x^A.B : \kappa}\ (\text{c-prod}) \qquad \frac{\Gamma \vDash^{\mathsf{c}} u : A \quad A =_\beta A' \quad \Gamma \vDash^{\mathsf{c}} A' : \kappa}{\Gamma \vDash^{\mathsf{c}} u : A'}\ (\text{c-conv})$$

où κ dénote l'une des constantes Prop ou Type.

Exemples d'utilisations du calcul des constructions

En fonction des types de A et B dans les règles (c-abs) et (c-prod) on parcourt le spectre allant du λ-calcul simplement typé à l'ordre supérieur en passant par les types dépendants, qu'il est alors possible de combiner.

Le λ-calcul simplement typé Quand A et B sont de type Prop dans (c-prod) et que x n'est pas libre dans B, alors $\forall x^A.B$ dénote en fait l'implication $A \to B$ du λ-calcul simplement typé. Par exemple, on peut exprimer et prouver la transitivité de l'implication :

$$A\ B\ C : \mathrm{Prop} \vDash^{\mathsf{c}} \lambda f^{A\to B}.\lambda g^{B\to C}.\lambda x^A.g\ (f\ x) : (A \to B) \to (B \to C) \to A \to C : \mathrm{Prop}$$

On voit que Prop est le type des propositions, et que tout est explicité dans CC : A, B et C sont des variables propositionnelles qui doivent être déclarées dans l'environnement avec le type Prop.

Prenons comme autre exemple la loi de Pierce dont on peut former le type correspondant dans CC :

$$A\ B : \mathrm{Prop} \vDash^{\mathsf{c}} ((A \to B) \to A) \to A : \mathrm{Prop}$$

Ce type ne possède aucun habitant avec cet environnement dans CC.

Le λ-calcul du second ordre : polymorphisme et imprédicativisme Dans le λ-calcul simplement typé, pour chaque type A il existe une fonction identité $\lambda x^A.x$ de type $A \to A$. Or le traitement effectué, l'algorithme, est indépendant du type de l'argument de la fonction identité. Reynolds (1974) propose alors un λ-calcul typé autorisant les fonctions polymorphes (quand A

et κ sont Prop), et qui permet par exemple d'exprimer une unique fonction identité *polymorphe* et son type :

$$\vdash^{\mathsf{c}} \lambda A^{\mathrm{Prop}}.\lambda x^{A}.x : \forall A^{\mathrm{Prop}}.A \to A : \mathrm{Prop}$$

Le polymorphisme correspond en logique à la quantification sur les propositions et introduit donc une notion d'imprédicativité sur les types du système. Cela autorise, entre autres choses, l'expression de types vides : par exemple la loi de Pierce quantifiée universellement peut être exprimée mais n'est pas habitée dans l'environnement vide :

$$\vdash^{\mathsf{c}} \forall A^{\mathrm{Prop}}.\forall B^{\mathrm{Prop}}.((A \to B) \to A) \to A : \mathrm{Prop}$$

Ou plus simplement, l'absurde $\bot$ peut être définie par $\forall A^{\mathrm{Prop}}.A$ dans CC :

$$\vdash^{\mathsf{c}} \bot : \mathrm{Prop} \qquad \text{avec} \qquad \vdash^{\mathsf{c}} \lambda B^{\mathrm{Prop}}.\lambda x^{\bot}.x\ B : \forall B^{\mathrm{Prop}}.\bot \to B : \mathrm{Prop}$$

D'autre part, Böhm et Berarducci (1985) constatent que le polymorphisme autorise le codage des types inductifs. Par exemple le type des entiers $\mathbb{N}$, défini par les constructeurs $0 : \mathbb{N}$ et $S : \mathbb{N} \to \mathbb{N}$, peut être représenté par les termes :

$$\begin{array}{rll} \mathbb{N} := & \forall X^{\mathrm{Prop}}.X \to (X \to X) \to X & : \mathrm{Prop} \\ 0 := & \lambda X^{\mathrm{Prop}}.\lambda x^{X}.\lambda f^{X \to X}.x & : \mathbb{N} \\ S := & \lambda n^{\mathbb{N}}.\lambda X^{\mathrm{Prop}}.\lambda x^{X}.\lambda f^{X \to X}.f\ (n\ X\ x\ f) & : \mathbb{N} \to \mathbb{N} \end{array}$$

où un entier de type $\mathbb{N}$ agit comme un *destructeur*, c'est-à-dire un itérateur (voir sec. 2.2.2).

Enfin, il est possible de définir des abréviations pour les formules produites par les connecteurs logiques usuels, par exemple :

$$\begin{array}{rl} \neg A := & A \to \bot \\ A \wedge B := & \forall X^{\mathrm{Prop}}.(A \to B \to X) \to X \\ A \vee B := & \forall X^{\mathrm{Prop}}.(A \to X) \to (B \to X) \to X \\ \exists A^{\mathrm{Prop}}.B := & \forall X^{\mathrm{Prop}}.(\forall A^{\mathrm{Prop}}.B \to X) \to X \end{array}$$

L'ordre supérieur À l'ordre supérieur (quand A est de type Type et κ est Prop), on peut définir des relations entre propositions. En particulier, les connecteurs logiques deviennent définissables en tant qu'objets du système :

$$\begin{array}{rll} \neg := & \lambda A^{\mathrm{Prop}}.A \to \bot & : \mathrm{Prop} \to \mathrm{Prop} \\ \wedge := & \lambda A^{\mathrm{Prop}}.\lambda B^{\mathrm{Prop}}.\forall X^{\mathrm{Prop}}.(A \to B \to X) \to X & : \mathrm{Prop} \to \mathrm{Prop} \to \mathrm{Prop} \\ \vee := & \lambda A^{\mathrm{Prop}}.\lambda B^{\mathrm{Prop}}.\forall X^{\mathrm{Prop}}.(A \to X) \to (B \to X) \to X & : \mathrm{Prop} \to \mathrm{Prop} \to \mathrm{Prop} \\ \exists_{\square} := & \lambda C^{\mathrm{Prop} \to \mathrm{Prop}}.\forall X^{\mathrm{Prop}}.(\forall A^{\mathrm{Prop}}.C\ A \to X) \to X & : (\mathrm{Prop} \to \mathrm{Prop}) \to \mathrm{Prop} \end{array}$$

où $\exists_{\square}$ est un quantificateur existentiel *propositionnel* avec $\exists_{\square}\ (\lambda A^{\mathrm{Prop}}.B)$ se réduit en $\exists A^{\mathrm{Prop}}.B$.

Les types dépendants À l'instar de la logique du premier ordre, il est parfois nécessaire de disposer de symboles de prédicats : par exemple si U : Prop est le domaine de quantification, alors on peut introduire un symbole P de type $U \to U \to \mathrm{Prop}$ (quand A est de type Prop et κ est Prop) afin d'écrire $P\ x\ y$ avec $x : U$ et $y : U$. Le type $P\ x\ y$ est un type qui *dépend des termes* x et y, et peut être formé dans CC. Par exemple dans l'environnement $\Gamma \equiv U : \mathrm{Prop}, P : U \to U \to \mathrm{Prop}$, on peut prouver la commutativité de la quantification universelle :

$$\Gamma \vdash^{\mathsf{c}} \lambda H^{\forall x^{U}.\forall y^{U}.P\ x\ y}.\lambda y^{U}.\lambda x^{U}.H\ y\ x : (\forall x^{U}.\forall y^{U}.P\ x\ y) \to \forall y^{U}.\forall x^{U}.P\ x\ y$$

Le calcul des constructions Toutes ces conceptions prises ensemble autorisent l'expression de nouveaux objets, comme par exemple la quantification existentielle du premier ordre sur un domaine U : Prop donné en argument :

$$\exists_\star := \lambda U^{\mathrm{Prop}}.\lambda C^{U\to\mathrm{Prop}}.\forall X^{\mathrm{Prop}}.(\forall a^U.C\ a \to X) \to X : \forall U^{\mathrm{Prop}}.(U \to \mathrm{Prop}) \to \mathrm{Prop}$$

et l'on peut par exemple prouver :

$$U : \mathrm{Prop}, u : U, P : U \to U \to \mathrm{Prop} \vdash^{\mathsf{c}} p : (\forall x^U.\forall y^U.P\ x\ y) \to \exists_\star^U z^U.P\ z\ z$$

où $p := \lambda H_1^{\forall x^U.\forall y^U.P\ x\ y}.\lambda X^{\mathrm{Prop}}.\lambda H_2^{\forall a^U.P\ a\ a\to X}.H_2\ u\ (H_1\ u\ u)$ et $\exists_\star^U z^U.P\ z\ z$ est une abréviation pour $\exists_\star\ U\ (\lambda z^U.P\ z\ z)$.

Coquand et Huet (1985) proposent des exemples plus conséquents d'utilisation du calcul des constructions pour formaliser diverses théories ou conceptions usuelles : l'idée simple est de postuler l'existence d'objets du bon type dans l'environnement (p. ex. les axiomes). L'utilisation du système est alors très proche de celle d'Automath et de ses *notions primitives* servant à postuler l'axiomatique d'une théorie formelle.

3.2 *Pédagogisme* dans le calcul des constructions

3.2.1 Critère de Poincaré formel

Rappelons le *critère de Poincaré* :

> *« Si une définition par postulat est utilisée, alors il doit exister un exemple d'objet la satisfaisant. »*

De façon similaire aux travaux précédents sur la pédagogie formelle, dans CC une définition par postulat d'un objet x peut être vue comme un environnement contenant x suivi d'hypothèses à propos de x. Par exemple, la définition par postulat informelle

> *« Soit x un entier naturel vérifiant les propriétés $P(x)$ et $Q(x)$. »*

est formellement représentée dans CC par l'environnement suivant

$$x : \mathbb{N}, H_1 : P(x), H_2 : Q(x)$$

où $\mathbb{N}$ est la définition imprédicative usuelle du type des entiers de Church dans CC (voir ci-avant).

Poincaré affirme qu'un tel ensemble d'hypothèses est une définition par postulat admissible de x **seulement si** nous sommes en mesure d'exhiber un entier naturel satisfaisant les deux propriétés P et Q. Autrement dit, les types $P(n)$ et $Q(n)$ doivent être habités pour un entier n donné. C'est-à-dire qu'il doit être possible de dériver :

$$\vdash^{\mathsf{c}} n : \mathbb{N} \qquad \vdash^{\mathsf{c}} t_1 : P(n) \qquad \vdash^{\mathsf{c}} t_2 : Q(n)$$

Si cela s'avère impossible (i.e. l'un des termes n, t_1 ou t_2 ne peut être construit) alors la définition est dénuée de sens et doit être évitée.

Inversement tout environnement d'un sous-système $\mathrm{CC}_\star$ de CC peut être vu comme un ensemble de définitions par postulats et doit alors être exemplifiable :

Exemplification d'un environnement

◊ L'environnement $x_1 : A_1, \ldots, x_n : A_n$ est *exemplifiable* dans $\mathrm{CC}_\star$ s'il existe des termes $t_1, \ldots, t_n$ tels que :

$$\vdash^\star t_1 : A_1 \qquad \vdash^\star t_2 : A_2[x_1 \leftarrow t_1] \qquad \ldots \qquad \vdash^\star t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$$

Le critère de Poincaré concerne toutes les définitions par postulats, donc tous les environnements utilisables. Dans CC un environnement Γ est utilisable dès qu'il est bien formé ($\Gamma\,\mathsf{wf}^{\mathsf{c}}$). Le critère de Poincaré pour un sous-système $\mathrm{CC}_\star$ de CC s'énonce alors formellement par :

Critère de Poincaré

◊ Un sous-système $\mathrm{CC}_\star$ de CC satisfait le *critère de Poincaré* si tous les environnements bien formés (i.e. $\Gamma\,\mathsf{wf}^\star$) sont exemplifiables dans $\mathrm{CC}_\star$.

Nous cherchons alors une version contrainte de CC satisfaisant le critère de Poincaré, et stable par réduction.

3.2.2 Extension naïve des résultats précédents à CC

Dans les travaux précédents sur la *pédagogisation* de systèmes formels usuels, chaque environnement est motivé avant d'être utilisé :

$$\frac{F \in \Delta \quad \vdash \sigma \cdot \Delta}{\Delta \vdash F}\ \text{(P-Hyp)}$$

Il est alors immédiat que tous les environnements utilisés sont exemplifiables, et donc le système satisfait trivialement le critère de Poincaré. Rappelons que la *motivation* précède l'introduction d'une définition par postulat, tandis que l'*exemplification* lui succède.

Malheureusement, un tel ajustement ne fonctionne pas dans CC : prenons le système naïf CC_n suivant dans lequel on remplace la notion d'environnement bien formé $\Gamma\,\mathsf{wf}^{\mathsf{n}}$ par le simple fait qu'il soit motivable.

Définition de CC_n

$$\frac{\vdash^{\mathsf{n}} \sigma \cdot \Gamma}{\Gamma \vdash^{\mathsf{n}} o : \top : \mathrm{Prop} : \mathrm{Type}}\ \text{(n-ax)} \qquad \frac{\vdash^{\mathsf{n}} \sigma \cdot (\Gamma, x : A, \Gamma')}{\Gamma, x : A, \Gamma' \vdash^{\mathsf{n}} x : A}\ \text{(n-var)}$$

$$\frac{\Gamma, x : A \vdash^{\mathsf{n}} u : B : \kappa}{\Gamma \vdash^{\mathsf{n}} \lambda x^A.u : \forall x^A.B}\ \text{(n-abs)} \qquad \frac{\Gamma \vdash^{\mathsf{n}} u : \forall x^A.B \quad \Gamma \vdash^{\mathsf{n}} v : A}{\Gamma \vdash^{\mathsf{n}} u\ v : B[x \leftarrow v]}\ \text{(n-app)}$$

$$\frac{\Gamma, x : A \vdash^{\mathsf{n}} B : \kappa}{\Gamma \vdash^{\mathsf{n}} \forall x^A.B : \kappa}\ \text{(n-prod)} \qquad \frac{\Gamma \vdash^{\mathsf{n}} u : A \quad A =_\beta A' \quad \Gamma \vdash^{\mathsf{n}} A' : \kappa}{\Gamma \vdash^{\mathsf{n}} u : A'}\ \text{(n-conv)}$$

où

- $\vdash^{\mathsf{n}} \sigma \cdot (x_1 : A_1, \ldots, x_n : A_n)$ dénote les dérivations $\vdash^{\mathsf{n}} \sigma(x_i) : \sigma(A_i)$;
- o et $\top$ sont deux constantes ajoutées au calcul.

L'approche naïve est trop naïve

Ces modifications sont en tous points comparables à celles que Colson et Michel (2009) ont effectuées pour le calcul du second ordre *faiblement* pédagogique $\mathrm{P_s}$-Prop^2. On y retrouve d'ailleurs le même défaut de non-stabilité de la réduction puisque

$$\vdash^{\mathsf{n}} (\lambda A^{\mathrm{Prop}}.\lambda x^A.x)\ \bot : \bot \to \bot$$

y est dérivable (avec $\bot := \forall A^{\mathrm{Prop}}.A$) mais pas

$$\vdash^{\mathsf{n}} \lambda x^{\bot}.x : \bot \to \bot$$

Cet exemple précis d'instabilité peut être évité en plaçant une contrainte sur la règle (n-app), similaire à celle introduite sur $(\forall_e)$ pour passer de $\mathrm{P_s}$-Prop^2 à P-Prop^2 et qui empêche l'instanciation de l'identité polymorphe sur $\bot$ (et donc la dérivation de $\bot \to \bot$).

Mais même ainsi, $\mathrm{CC_n}$ n'est pas un sous-système de CC. En effet, les jugements suivants sont dérivables dans $\mathrm{CC_n}$ mais pas dans CC :

(a) $x_1 : \mathrm{Type} \vdash^{\mathsf{n}} \mathrm{Prop} : \mathrm{Type}$

(b) $x_1 : \mathrm{Prop}, x_2 : (\lambda H^{\top \to x_1}.\top)\ (\lambda y^{\top}.y) \vdash^{\mathsf{n}} \mathrm{Prop} : \mathrm{Type}$

(c) $x_1 : \mathbb{N}, x_2 : (\lambda H^{x_1 = 0}.\top)\ (\lambda P^{\mathbb{N} \to \mathrm{Prop}}.\lambda H^{P\ 0}.H) \vdash^{\mathsf{n}} \mathrm{Prop} : \mathrm{Type}$

(d) $x_1 : x_2, x_2 : \mathrm{Prop} \vdash^{\mathsf{n}} \mathrm{Prop} : \mathrm{Type}$

Les dérivations correspondantes utilisent la règle (n-ax) avec les motivations suivantes :

(a) $\sigma_1 := [x_1 \mapsto \mathrm{Prop}]$

(b) $\sigma_2 := [x_1 \mapsto \top; x_2 \mapsto o]$

(c) $\sigma_3 := [x_1 \mapsto 0; x_2 \mapsto o]$

(d) $\sigma_4 := [x_1 \mapsto o; x_2 \mapsto \top]$

Ces jugements ne sont pas dérivables dans CC pour les raisons suivantes :

(a) Type ne peut pas apparaître dans un environnement (lem. 1.4.2) ;

(b) $(\lambda H^{\top \to x_1}.\top)\ (\lambda y^{\top}.y)$ est mal formé puisque la fonction attend un argument de type $\top \to x_1$ mais un terme de type $\top \to \top$ lui est donné à la place ;

(c) même raison que précédemment : la fonction attend une preuve de $x_1 = 0$ alors qu'une preuve de $0 = 0$ lui est donnée ;

(d) la « définition » de x_1 dépend de celle de x_2 qui se situe après x_1 dans l'environnement, ce qui n'est pas autorisé (lem. 1.4.1).

Notons qu'en imposant des contraintes sur les substitutions et les environnements pendant les motivations, on peut éliminer les cas pathologiques (a) et (d). Les deux autres cas relèvent d'une difficulté plus fondamentale, l'interdépendance morphologie-syntaxe (nécessaire) des systèmes de types dépendants, qui se voit *court-circuitée* dans $\mathrm{CC_n}$ par les règles modifiées (n-ax) et (n-var).

Changement de stratégie

Partant de ce constat, on peut vouloir réintroduire la notion d'environnement bien formé, i.e. les règles (c-env$_1$) et (c-env$_2$), parallèlement à la motivabilité des environnements :

$$\frac{\Gamma\ \mathsf{wf}^{\mathsf{n}} \quad \vdash^{\mathsf{n}} \sigma \cdot \Gamma}{\Gamma \vdash^{\mathsf{n}} o : \top : \mathrm{Prop} : \mathrm{Type}}\ (\text{n-ax'}) \qquad \frac{\Gamma, x : A, \Gamma'\ \mathsf{wf}^{\mathsf{n}} \quad \vdash^{\mathsf{n}} \sigma \cdot (\Gamma, x : A, \Gamma')}{\Gamma, x : A, \Gamma' \vdash^{\mathsf{n}} x : A}\ (\text{n-var'})$$

Cette variante n'empêche pas la production de types non-habités comme $\vdash^{\mathsf{n}} \forall A^{\mathrm{Prop}}.A : \mathrm{Prop}$. Si de tels types *vides* étaient tolérés, il faudrait en plus veiller à restreindre ou éviter leur utilisation afin de respecter la stabilité par réduction, et contraindre toutes les règles du calcul. En général, leur simple présence en tant que sous-terme est rapidement problématique. Notons aussi que dans un tel système *mixte*, la définition formelle du critère de Poincaré devrait être adaptée à cause de la distinction entre *environnement bien formé* et *environnement motivable* qui y serait introduite.

C'est pourquoi nous préférons adopter une autre stratégie : **éviter la formation des types vides au plus tôt**. CC possède cet avantage que tous les types bien formés sont construits dans le système en passant par la règle (c-prod) : c'est sur cette dernière que nous concentrons nos prochains efforts d'investigation.

3.2.3 Un calcul des constructions Poincaréien – $\mathrm{CC_r}$

Dans CC, les règles d'inférence construisent des λ-termes avec leur type associé. La seule exception est la règle (c-prod) qui se contente de construire des types. Nous décidons de la *neutraliser* en remplaçant (c-prod) par (r-prod) :

$$\frac{\Gamma, x : A \vdash^{\mathsf{c}} B : \kappa}{\Gamma \vdash^{\mathsf{c}} \forall x^A.B : \kappa}\ \text{(c-prod)} \qquad \frac{\Gamma, x : A \vdash^{\mathsf{r}} u : B : \kappa}{\Gamma \vdash^{\mathsf{r}} \forall x^A.B : \kappa}\ \text{(r-prod)}$$

On peut alors condenser la règle de formation de l'abstraction (r-abs) et de formation du produit (r-prod) :

$$\frac{\Gamma, x : A \vdash^{\mathsf{c}} u : B : \kappa}{\Gamma \vdash^{\mathsf{c}} \lambda x^A.u : \forall x^A.B : \kappa}\ \text{(r-abs + r-prod)}$$

C'est pourquoi nous considérons qu'il s'agit d'une neutralisation de la règle (c-prod) : (r-prod) ne peut plus être déclenchée indépendamment de (r-abs).

Par cette simple contrainte additionnelle, nous obtenons un calcul $\mathrm{CC_r}$ qui satisfait le critère de Poincaré et qui est stable par réduction. Nous pouvons ainsi affirmer que (c-prod) est la règle principale responsable de la *vacuité* dans CC.

Remarquons que dans $\mathrm{CC_r}$ **la syntaxe subsume la morphologie**, autrement dit les types émergent de l'objet qui leur donne une signification : si t est de type A alors A est un type. Cela contraint évidemment tous les types à être habités. Ce phénomène a été anticipé par Michel (2008) à propos des réflexions menées par Colson (1986) qui sont à l'origine de l'étude de la pédagogie formelle.

Résultats majeurs concernant $\mathrm{CC_r}$

La stabilité par réduction du calcul résultant $\mathrm{CC_r}$ se montre de façon usuelle (voir sec. 1.4) : la formalisation dans l'assistant à la preuve Coq des méta-théorèmes usuels de CC par Barras (1996) a été adaptée pour $\mathrm{CC_r}$ [3]. En pratique il suffit de modifier la définition de l'une des règles d'inférence du système (représentées par un type inductif), et d'adapter quelques démonstrations de lemmes intermédiaires pour que la preuve de stabilité par réduction pour $\mathrm{CC_r}$ aboutisse en Coq.

3. Le code pour Coq 8.3pl2 est disponible à l'adresse `http://www.lita.sciences.univ-metz.fr/~demange/publications/sources/CoqR.tar.gz` (2012).

Le fait que $\mathrm{CC_r}$ vérifie le critère de Poincaré repose sur la normalisation forte de CC : dans CC, et donc dans $\mathrm{CC_r}$, tout type clos se réduit en un produit de la forme $\forall x^A.B$ (lem. 2.2.17). Or dans $\mathrm{CC_r}$ la règle (r-prod) impose que tout produit soit habité, ce qui implique que tout type clos est habité (cor. 2.2.19). Finalement, puisque tout environnement bien formé $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{\mathsf{r}}$ commence par un type clos A_1, ce dernier est habité par un terme t_1 que l'on peut substituer à x_1 dans l'environnement précédent pour obtenir $x_2 : A_2[x_1 \leftarrow t_1\,], \ldots, x_n : A_n[x_1 \leftarrow t_1\,]\,\mathsf{wf}^{\mathsf{r}}$, etc. On construit ainsi la suite de termes nécessaire pour montrer que l'environnement est exemplifiable. L'environnement étant quelconque, on en déduit que $\mathrm{CC_r}$ vérifie le critère de Poincaré (prop. 2.2.20).

Comme conséquence importante, on montre que toute fonction de $\mathrm{CC_r}$ est utile (thm. 2.2.21) : si $\vdash^{\mathsf{r}} f : \forall x^A.B$ alors il existe un terme u tel que $\vdash^{\mathsf{r}} u : A$.

Pouvoir logique et calculatoire de $\mathrm{CC_r}$

Ainsi $\mathrm{CC_r}$ semble être un bon candidat pour un calcul des constructions pédagogique. Malheureusement son expressivité logique paraît faible : il ne contient même pas « nativement » le λ-calcul simplement typé (thm. 2.2.30). Il n'autorise pas non plus la démonstration de la symétrie de l'égalité de Leibniz : on n'a pas de terme u tel que $\vdash^{\mathsf{r}} u : \forall A^{\mathrm{Prop}}.\forall x^A.\forall y^A.x =_A y \to y =_A x$ (lem. 2.2.31).

D'un point de vue calculatoire, on peut interpréter les termes du système T de Gödel (1958) dans $\mathrm{CC_r}$ (voir sec. 2.2.2). On s'appuie pour cela sur la méthode développée initialement par Kleene (1935) pour représenter la fonction prédécesseur sur les entiers de Church (1933). On représente alors les termes, les types, et la récursion primitive (depuis l'itération) d'une façon similaire à ce que proposent Girard *et al.* (1990) pour interpréter les termes du système T dans le système F. L'expression de la récursion primitive depuis l'itération constitue l'étape délicate de l'interprétation du système T dans $\mathrm{CC_r}$ puisque ce dernier n'autorise pas le codage usuel et général des couples.

L'itération satisfait les règles de réduction suivantes

$$\begin{aligned} \mathrm{it}_T(0, b, (y^T)step) &\overset{*}{\rightsquigarrow}_\beta b \\ \mathrm{it}_T(S(n), b, (y^T)step) &\overset{*}{\rightsquigarrow}_\beta step[y \leftarrow it_T(n, b, (y^T)step)\,] \end{aligned}$$

tandis que la récursion primitive satisfait

$$\begin{aligned} \mathrm{rec}_T(0, b, (x^{\mathbb{N}}, y^T)step') &\overset{*}{\rightsquigarrow}_\beta b \\ \mathrm{rec}_T(S^{n+1}(0), b, (x^{\mathbb{N}}, y^T)step') &\overset{*}{\rightsquigarrow}_\beta step'[x, y \leftarrow S^n(0), rec_T(S^n(0), b, (x^{\mathbb{N}}, y^T)step')\,] \end{aligned}$$

c'est-à-dire que l'étape de la récursion $step'$ « connaît » en plus la *profondeur* de la récurrence représentée par l'entier de Church $S^n(0)$.

La définition usuelle de la récursion primitive depuis l'itération consiste alors à construire en même temps deux suites g_n et d_n où g_n conserve la profondeur de la récurrence, et d_n le résultat de la récurrence, formellement :

$$\begin{aligned} g_0 &=_\beta 0 \\ g_{n+1} &=_\beta S(g_n) \end{aligned} \qquad \text{et} \qquad \begin{aligned} d_0 &=_\beta b \\ d_{n+1} &=_\beta step'[x, y \leftarrow g_n, d_n\,] \end{aligned}$$

En « encapsulant » ces deux suites dans un couple $v_n =_\beta \langle g_n, d_n \rangle$, il devient possible d'exprimer v_{n+1} en fonction de v_n uniquement, c'est-à-dire que l'on est en droit d'utiliser l'itération afin

d'exprimer v_n :

$$\begin{aligned} v_0 &=_\beta \langle g_0, d_0 \rangle =_\beta \langle 0, b \rangle^T \\ v_{n+1} &=_\beta \langle g_{n+1}, d_{n+1} \rangle \\ &=_\beta \langle S(g_n), step'[x, y \leftarrow g_n, d_n\,] \rangle \\ &=_\beta \langle S(\pi_1(v_n)), step'[x, y \leftarrow \pi_1(v_n), \pi_2(v_n)\,] \rangle \end{aligned}$$

où π_i représente la i^{e} projection. Après avoir calculé v_n par l'itération, le résultat de la récurrence est alors $\pi_2(v_n)$.

Les couples utilisés possèdent une composante entière (de type $\mathbb{N}$) et l'autre du type du résultat : ils sont *hétérogènes*. Or dans $\mathrm{CC_r}$ il n'est pas possible d'utiliser le codage usuel des couples hétérogènes car le type suivant n'est pas dérivable : $A \times B := \forall C^{\mathrm{Prop}}.(A \to B \to C) \to C$. En revanche, les produits cartésiens *homogènes* $T \times T$ sont autorisés pourvu que T soit un type dans $\mathrm{CC_r}$. En particulier, si T est un type simple sur $\mathbb{N}$ (i.e. formé à partir de $\mathbb{N}$ et $\to$), on peut coder un entier de type $\mathbb{N}$ dans T par une fonctionnelle enc_T de type $\mathbb{N} \to T$, et donc représenter des *pseudos-couples hétérogènes* : $\mathbb{N} \times T := (T \to T \to T) \to T$ où $\langle n, t \rangle^T := \lambda f^{T \to T \to T}.f\ (\mathrm{enc}_T\ n)\ t$. Ces couples ont le comportement attendu, et permettent ainsi de représenter effectivement la récursion primitive depuis l'itération dans $\mathrm{CC_r}$ par la technique exposée ci-avant.

3.2.4 Définition d'un sous-système pédagogique de CC

Les limitations logiques de $\mathrm{CC_r}$ suggèrent une définition plus précise d'un calcul des constructions pédagogique. Il semble en effet surprenant de ne pouvoir prouver la symétrie de l'égalité de Leibniz dans un calcul pédagogique. En effet, la non-vacuité du type $x =_A y$ peut être justifiée par la substitution qui à A associe $\mathbb{N}$ et à x et y associe 0. Cela signifie que non seulement nous attendons que les environnements utilisés soient exemplifiables, mais que l'inverse aussi doit être vérifié.

Cependant, on a vu que la réciproque immédiate *« tout environnement motivable est bien formé »* peut poser des difficultés (ssec. 3.2.2) : les environnements Γ motivables ne sont pas nécessairement des environnements bien formés $\Gamma\ \mathsf{wf}^{\mathsf{c}}$ de CC. La définition formelle de la réciproque du critère de Poincaré que nous proposons en tient compte :

Réciproque du critère de Poincaré

◊ Un sous-système $\mathrm{CC}_\star$ de CC vérifie la *réciproque du critère de Poincaré* si dès que

$$\vdash^\star t_1 : A_1 \quad \vdash^\star t_2 : A_2[x_1 \leftarrow t_1\,] \quad \ldots \quad \vdash^\star t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}\,]$$

et

$$x_1 : A_1, \ldots, x_n : A_n\ \mathsf{wf}^{\mathsf{c}}$$

alors

$$x_1 : A_1, \ldots, x_n : A_n\ \mathsf{wf}^\star$$

Nous en venons alors à une définition formelle d'un sous-système pédagogique de CC. Un tel système doit : (a) être un sous-système de CC ; (b) être stable par réduction ; (c) vérifier le critère de Poincaré ; (d) vérifier la réciproque du critère de Poincaré.
Ces conditions sont en effet nécessaires : le calcul propositionnel du second-ordre *faiblement pédagogique* satisfait les conditions (a), (c), (d) mais pas (b) ; $\mathrm{CC_r}$ satisfait (a), (b), (c) mais pas (d) ; (c) est la caractéristique fondamentale d'un système pédagogique ; (b) est un attribut

essentiel de tout système fonctionnel. On aboutit alors à la première définition formelle d'un sous-système pédagogique de CC :

Sous-système pédagogique de CC

◊ Un système $\mathrm{CC}_\star$ est un *sous-système pédagogique de CC* si :

(i) $\mathrm{CC}_\star$ est un sous-système de CC : si $\Gamma\,\mathsf{wf}^\star$ alors $\Gamma\,\mathsf{wf}^\mathsf{c}$; et si $\Gamma \vDash^\star w : C$ alors $\Gamma \vDash^\mathsf{c} w : C$.

(ii) $\mathrm{CC}_\star$ est stable par réduction : si $\Gamma \vDash^\star t : C$ et $t \rightsquigarrow_\beta t'$, alors $\Gamma \vDash^\star t' : C$;

(iii) $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^\star$ **si et seulement si** $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^\mathsf{c}$ et il existe des termes $t_1, \ldots, t_n$ tels que

$$\vDash^\star t_1 : A_1 \quad \vDash^\star t_2 : A_2[x_1 \leftarrow t_1] \quad \ldots \quad \vDash^\star t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$$

Il est possible de construire un sous-système de λ^2 pédagogique au sens de cette définition et équivalent au λ-calcul pédagogique du second-ordre de Colson et Michel (2009). Cette construction, que nous décrivons dans la section suivante, est un exemple de sous-système pédagogique de CC et constitue par là une justification (pédagogique) de cette définition formelle.

3.3 Sous-systèmes pédagogiques du second-ordre

Dans cette section, nous construisons un sous-système pédagogique de λ^2, et donc de CC, que nous nommons λ^2_p. Nous passons par l'intermédiaire de deux autres systèmes : λ^2_e où les motivations utilisées sont explicites et totales, et λ^2_t où les motivations sont seulement totales. Ces trois systèmes se révèlent équivalents, et P-Prop^2 de Colson et Michel (2009) peut être plongé dans l'un quelconque d'entre eux et vice-versa.

Le passage du système λ^2_e au système λ^2_t s'effectue en relâchant une contrainte sur les règles à plusieurs prémisses : dans λ^2_e une même motivation est nécessaire pour toutes les prémisses, ce qui n'est plus le cas dans λ^2_t qui rend les motivations implicites. Ce relâchement est suggéré par le fait que les motivations de λ^2_e sont interchangeables (lem. 3.2.24).

Dans les systèmes λ^2_e et λ^2_t les motivations sont totales : toutes les hypothèses de l'environnement sont motivées par des termes clos. Le système λ^2_p relâche cette contrainte : les motivations peuvent agir sur une partie (éventuellement vide) de l'environnement à motiver, et associer des termes non clos. Il n'est ainsi plus nécessaire d'introduire un exemple de formule canonique $\top$ et son habitant o que l'on peut alors identifier à l'identité polymorphe : $\top := \forall A^{\mathrm{Prop}}.A \to A$ et $o := \lambda A^{\mathrm{Prop}}.\lambda x^A.x$.

3.3.1 Avec motivations totales explicites – λ^2_e

Idées conceptuelles de base

Pour représenter un état courant d'une preuve, on utilise un *séquent* $\Gamma \vdash t : A$ qui signifie « t est une preuve de A sous les hypothèses Γ ». Dans la pratique pédagogique, on a besoin en plus d'exemples des hypothèses de Γ, ce qu'on peut formaliser en utilisant des *séquents augmentés d'une exemplification* de la forme $\Gamma \vdash_\sigma t : A$ signifiant « t est une preuve de A sous les hypothèses Γ qui sont exemplifiées par σ ». L'idée intuitive est que lorsque $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$, alors σ est une substitution $[x_1 \mapsto t_1, \ldots, x_n \mapsto t_n]$ où les t_i sont des exemples des A_i, i.e. $\vdash_{[]} t_i : A_i[x_1, \ldots, x_{i-1} \leftarrow t_1, \ldots, t_{i-1}]$. On dira alors que σ motive/exemplifie Γ [4]. De même on

passe des *jugements* de la forme $\Gamma\,\mathsf{wf}$ à des jugements augmentés $\Gamma\,\mathsf{wf}_\sigma$. Ces motivations que l'on explicite, sont totales dans le sens où toutes les hypothèses de Γ sont exemplifiées.

Rendre les motivations explicites possède deux avantages. D'une part, cela permet de rendre compte plus fidèlement de la pratique des mathématiques pédagogiques durant laquelle on conserve mentalement un même exemple tout au long de la preuve. D'autre part, cela simplifie et précise les *méta-théorèmes* du formalisme associé : on peut agir sur les motivations, et ainsi mieux apprécier les contraintes qu'elles subissent ou imposent.

Définition du système

Nous appliquons cette idée à λ^2 (voir sec. 3.1), qui est une restriction de CC correspondant au système F, pour obtenir un système que nous nommons λ^2_e et défini par les règles suivantes :

$$\frac{}{[\,]\,\mathsf{wf}^{2_e}_{[\,]}}\ (\text{e-env}_1) \qquad \frac{\Gamma \vdash^{2_e}_{\sigma} A : \kappa \quad \vdash^{2_e}_{[\,]} a : \sigma(A) \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^{2_e}_{\sigma::(x \mapsto a)}}\ (\text{e-env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^{2_e}_{\sigma}}{\Gamma \vdash^{2_e}_{\sigma} o : \top : \mathrm{Prop} : \mathrm{Type}}\ (\text{e-ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^{2_e}_{\sigma}}{\Gamma, x : A, \Gamma' \vdash^{2_e}_{\sigma} x : A}\ (\text{e-var})$$

$$\frac{\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_e}_{\sigma} \lambda x^A.u : \forall x^A.B}\ (\text{e-abs}) \qquad \frac{\Gamma \vdash^{2_e}_{\sigma} u : \forall x^A.B \quad \Gamma \vdash^{2_e}_{\sigma} v : A}{\Gamma \vdash^{2_e}_{\sigma} u\ v : B[x \leftarrow v\,]}\ (\text{e-app})$$

$$\frac{\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} B : \mathrm{Prop} \quad \vdash^{2_e}_{[\,]} t : \sigma(\forall x^A.B)}{\Gamma \vdash^{2_e}_{\sigma} \forall x^A.B : \mathrm{Prop}}\ (\text{e-prod})$$

Discussion sur les contraintes introduites

La règle (prod) de λ^2 est contrainte en (e-prod) dans λ^2_e afin d'éviter la création de types vides au plus tôt : p. ex. $\bot := \forall A^{\mathrm{Prop}}.A$ qui représente l'absurde dans λ^2 n'est pas motivable (car close et non habitée). La contrainte impose alors que le type formé soit *compatible* avec la motivation courante σ, c'est-à-dire que l'instance $\sigma(\forall x^A.B)$ du type formé soit habitée.

Afin de conserver la bonne correspondance environnement-motivation lorsque l'environnement est augmenté d'une hypothèse A, il est nécessaire d'introduire une *pseudo-contrainte* sur la règle (env$_2$) : $\vdash^{2_e}_{[\,]} a : \sigma(A)$. Cette prémisse supplémentaire ne doit pas être considérée comme une réelle contrainte puisque la dérivation $\Gamma \vdash^{2_e}_{\sigma} A : \kappa$ contient la construction de l'exemple a nécessaire (lem. 3.2.12). Ce fait est important pour les systèmes à motivations explicites : si $\Gamma \vdash_{\sigma} A : \kappa$ ne permet pas la construction d'un exemple a de $\sigma(A)$, cela signifie que la motivabilité de A n'a pas été vérifiée suffisamment tôt par le système, et que A est potentiellement *dangereuse* (pour la stabilité par réduction) voire inutilisable.

Enfin, les motivations étant totales, il est indispensable d'introduire une formule de base $\top$ qui représente la formule toujours prouvable et qui est habitée par un exemple canonique o. Ces constantes sont nécessaires puisque sans elles les deux seules dérivations possibles seraient $[\,]\,\mathsf{wf}^{2_e}_{[\,]}$ et $\vdash^{2_e}_{[\,]} \mathrm{Prop} : \mathrm{Type}$.

4. Il devient difficile de distinguer entre exemplification et motivation dans un système avec séquents augmentés : σ agit d'abord comme motivation, puis est conservée en tant qu'exemplification.

Résultats majeurs concernant λ_e^2

La correspondance environnement-motivation impose immédiatement au système de satisfaire le critère de Poincaré (prop. 3.2.11) :

$$\text{Si } x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{2_e}_{\sigma} \text{ alors } \vdash^{2_e}_{[]} \sigma(x_i) : \sigma(A_i).$$

Inversement, λ_e^2 vérifie la réciproque du critère de Poincaré (lem. 3.2.15) :

$$\text{Si}$$

$$\vdash^{2_e}_{[]} t_1 : A_1 \quad \vdash^{2_e}_{[]} t_2 : A_2[x_1 \leftarrow t_1\,] \quad \ldots \quad \vdash^{2_e}_{[]} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}\,]$$

$$\text{et}$$

$$x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n \,\mathsf{wf}^{\mathsf{c}}$$

$$\text{alors}$$

$$x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n \,\mathsf{wf}^{2_e}_{(x_1 \mapsto t_1)::(x_2 \mapsto t_2)::\ \ldots\ ::(x_n \mapsto t_n)}$$

Notons que la preuve de ce résultat n'utilise pas fondamentalement l'hypothèse $x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n \,\mathsf{wf}^{\mathsf{c}}$: il suffit que les A_i soient différents de Type. Autrement dit, sous cette hypothèse, on peut affirmer que tout environnement motivable est bien formé dans λ^2. C'est pourquoi la nécessité de la réciproque de Poincaré n'apparaît pas dans les travaux précédents sur la pédagogisation du système F ou Type n'est pas présent dans le formalisme utilisé.

Et enfin, λ_e^2 est stable par réduction (prop. 3.2.26) :

$$\text{Si } \Gamma \vdash^{2_e}_{\sigma} t : C \text{ et } t \rightsquigarrow_\beta t', \text{ alors } \Gamma \vdash^{2_e}_{\sigma} t' : C.$$

Ce résultat s'obtient de façon usuelle en établissant un lemme de substitution d'un terme dans l'environnement (lem. 3.2.25). L'énoncé de ce lemme est plus précis dans un système avec motivations explicites, et constitue l'un des résultats intermédiaires d'intérêt à propos de λ_e^2.

Résultats intermédiaires d'intérêt sur λ_e^2

Le lemme de substitution On peut vouloir formuler le lemme de substitution pour λ_e^2 par :

$$\text{Si } \Gamma, y : C, \Gamma' \,\mathsf{wf}^{2_e}_{\sigma::(y \mapsto c)::\,\sigma'} \text{ et } \Gamma \vdash^{2_e}_{\sigma} w : C, \text{ alors } \Gamma, \Gamma'[y \leftarrow w\,] \,\mathsf{wf}^{2_e}_{\sigma::\,\sigma'}.$$

où la motivation $\sigma::(y \mapsto c)::\sigma'$ est simplement « contractée » d'une position en $\sigma::\sigma'$.

Lorsque w est un terme, y étant une variable de terme, elle n'apparaît pas dans les types du reste de l'environnement Γ'. Dans ce cas σ' motive aussi $\Gamma'[y \leftarrow w\,] \equiv \Gamma'$.

Mais lorsque w est un type, y étant une variable de type, elle peut apparaître dans les types du reste de l'environnement. Et donc substituer w pour y *force* une remise en question de l'exemplification σ' qui était utilisée pour motiver Γ', et qui est éventuellement incompatible avec $\Gamma'[y \leftarrow w\,]$.

Par exemple, avec $\Gamma := [\,]$, $C := \text{Prop}$, $\Gamma' := z : y$ et $w := \top \to \top$ on a

$$y : \text{Prop}, z : y \,\mathsf{wf}^{2_e}_{(y \mapsto \top)::(z \mapsto o)}$$

mais on n'a pas

$$z : \top \to \top \,\mathsf{wf}^{2_e}_{(z \mapsto o)}$$

puisque sinon le critère de Poincaré pour λ_e^2 (prop. 3.2.11) nous donnerait

$$\vdash^{2_e}_{[]} o : \top \to \top$$

Il faut alors préciser la contrainte que subit la motivation par la formulation suivante :

$$\text{Si } \Gamma, y : C, \Gamma' \,\mathsf{wf}^{2_e}_{\sigma::(y \mapsto c)::\sigma'} \text{ et } \Gamma \vdash^{2_e}_{\sigma} w : C, \text{ alors il existe } \rho \text{ telle que } \Gamma, \Gamma'[y \leftarrow w\,]\,\mathsf{wf}^{2_e}_{\sigma::\rho}.$$

On voit ainsi comment l'explicitation des motivations précise un comportement intuitif : si l'on décide de fixer une hypothèse dans une démonstration, alors les exemples associés aux autres hypothèses qui en dépendent doivent être réévalués.

La transmission des motivations en conclusion Pour tout séquent la motivation se transmet à la conclusion (lem. 3.2.17), ce qui est déjà le cas dans (Colson et Michel, 2009, lem. 17) :

$$\text{Si } \Gamma \vdash^{2_e}_{\sigma} w : C \text{ alors } \vdash^{2_e}_{[]} \sigma(w) : \sigma(C).$$

Bien sûr cela est souhaité : un raisonnement hypothétique valide sur des objets abstraits doit le rester sur nos exemples concrets. C'est d'ailleurs le rôle de ces exemples.

Échange des motivations Pour passer de λ^2_e à λ^2_t, on relâche les contraintes sur les règles (e-abs) et (e-app) en autorisant à ce que les prémisses soient motivées par des substitutions différentes. L'équivalence de ces deux systèmes repose sur le résultat d'échange des motivations (lem. 3.2.24) suivant :

$$\text{Si } \Gamma \vdash^{2_e}_{\sigma} w : C \text{ et } \Gamma\,\mathsf{wf}^{2_e}_{\sigma'}, \text{ alors } \Gamma \vdash^{2_e}_{\sigma'} w : C.$$

En effet, dans λ^2_e les formules en exemple dans σ sont closes, de même que celles de σ'. Or les formules closes sont toutes équivalentes, et peuvent alors être substituées les unes aux autres.

3.3.2 Avec motivations totales – λ^2_t

λ^2_e n'est pas un sous-système pédagogique de CC au sens de la définition formelle précédente, en particulier à cause des motivations explicites. Ce point est résolu avec λ^2_t défini par :

$$\frac{}{[\,]\,\mathsf{wf}^{2_t}}\ (\text{t-env}_1) \qquad \frac{\Gamma \vdash^{2_t} A : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^{2_t}}\ (\text{t-env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^{2_t}}{\Gamma \vdash^{2_t} o : \top : \mathrm{Prop} : \mathrm{Type}}\ (\text{t-ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^{2_t}}{\Gamma, x : A, \Gamma' \vdash^{2_t} x : A}\ (\text{t-var})$$

$$\frac{\Gamma, x : A \vdash^{2_t} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_t} \lambda x^A.u : \forall x^A.B}\ (\text{t-abs}) \qquad \frac{\Gamma \vdash^{2_t} u : \forall x^A.B \quad \Gamma \vdash^{2_t} v : A}{\Gamma \vdash^{2_t} u\ v : B[x \leftarrow v\,]}\ (\text{t-app})$$

$$\frac{\Gamma, x : A \vdash^{2_t} B : \mathrm{Prop} \quad \sigma\,\mathrm{mot}_\Gamma\,\forall x^A.B}{\Gamma \vdash^{2_t} \forall x^A.B : \mathrm{Prop}}\ (\text{t-prod})$$

La contrainte $\sigma\,\mathrm{mot}_\Gamma\,\forall x^A.B$ signifie d'une part que σ motive Γ (noté $\sigma\,\mathrm{mot}\,\Gamma$), et d'autre part qu'il existe un terme t tel que $\vdash^{2_t} t : \sigma(\forall x^A.B)$. Rappelons que $[x_1 \mapsto t_1, \ldots, x_n \mapsto t_n]$ motive $x_1 : A_1, \ldots, x_n : A_n$ signifie simplement que $\vdash^{2_t} t_i : A_i[x_1, \ldots, x_{i-1} \leftarrow t_1, \ldots, t_{i-1}\,]$.

Équivalence des systèmes λ_e^2 et λ_t^2

L'équivalence des systèmes λ_t^2 et λ_e^2 est immédiate (thm. 3.3.6 et 3.3.7) :
- $\sigma \operatorname{mot}_\Gamma \forall x^A.B$ *si et seulement si* $\Gamma \mathsf{wf}_\sigma^{2_e}$ et il existe un terme t tel que $\vdash_{[]}^{2_e} t : \sigma(\forall x^A.B)$, c'est-à-dire que (t-prod) *simule* (e-prod) et vice-versa ;
- les règles à deux prémisses (t-abs) ou (t-app) dans λ_t^2 sont équivalentes à leur homologues (e-abs) et (e-app) dans λ_e^2 : pour simuler (t-app) par (e-app), il suffit de récupérer une motivation de l'une des prémisses et de l'utiliser pour motiver l'autre par échange de motivations dans λ_e^2.

En conséquence, λ_t^2 vérifie le critère de Poincaré et sa réciproque ainsi que la stabilité par réduction. Mais ce n'est toujours pas un sous-système pédagogique de CC à cause des constantes o et $\top$. Encore une fois, ces constantes sont nécessaires : autrement les seuls éléments de type Prop seraient des variables ou la constante $\top$.

Dans la règle (t-prod), la motivation systématique de tout l'environnement pour construire un élément de type Prop oblige la construction préalable d'un élément de type Prop, qui lui-même nécessite la construction préalable d'un élément de type Prop, etc. Pour sortir de cette impasse, dans λ_e^2 et λ_t^2, on doit fixer des éléments initiaux (arbitraires) o et $\top$ avec $o : \top : \mathrm{Prop}$.

Pourtant dans $\mathrm{CC_r}$ le problème ne se pose pas et ces constantes sont définissables par $o := \lambda A^{\mathrm{Prop}}.\lambda x^A.x$ et $\top := \forall A^{\mathrm{Prop}}.A \to A$. Cela est rendu possible par la règle (r-prod) qui nécessite d'habiter le produit construit, mais sans avoir à motiver tout l'environnement.

Dans λ_p^2 nous autorisons une *motivation partielle* de l'environnement lors de la formation des produits par la règle (p-prod). Quand la motivation est vide le comportement est celui de $\mathrm{CC_r}$, quand elle est totale le comportement est celui de λ_t^2. Ainsi, à l'instar de $\mathrm{CC_r}$, les constantes o et $\top$ deviennent définissables.

3.3.3 Avec motivations partielles – λ_p^2

Les motivations partielles

Introduisons l'idée sur un exemple : nous allons motiver l'environnement $\Gamma := x_1 : A_1, \ldots, x_5 : A_5$ par la substitution $\sigma := [x_2 \mapsto t_2, x_4 \mapsto t_4]$ dont le domaine $\{x_2, x_4\}$ est un sous-ensemble strict de $\{x_1, \ldots, x_5\}$.

Commençons par appliquer la substitution σ à l'environnement Γ :
- x_1 n'est pas motivée par σ, et son type A_1 ne contient pas d'occurrence de x_2 ni de x_4 : l'environnement résultant de l'application de la motivation partielle σ contient alors $x_1 : A_1$;
- x_2 est motivée par σ donc n'apparaît pas dans l'environnement résultant ;
- x_3 n'est pas motivée par σ, mais son type peut contenir des occurrences de x_2 qui sont motivées par t_2 : l'environnement résultant contient alors $x_3 : A_3[x_2 \leftarrow t_2]$;
- etc.

À la fin de la procédure, l'environnement résultant est :

$$\sigma(\Gamma) \equiv x_1 : A_1, x_3 : A_3[x_2 \leftarrow t_2], x_5 : A_5[x_2, x_4 \leftarrow t_2, t_4]$$

Il est nécessaire de contrôler que la substitution σ associe aux variables x_i des termes correctement typés t_i (possiblement ouverts). En reprenant l'exemple précédent, il faut vérifier que t_2 est bien typé dans la partie de l'environnement résultant qui déclare la variable x_1. De même t_4 doit être bien typé dans la partie de l'environnement déclarant x_1 et x_3. Au final, on doit avoir

les dérivations suivantes :

$$x_1 : A_1 \vdash^{2_\mathrm{p}} t_2 : A_2 \qquad x_1 : A_1, x_3 : A_3[x_2 \leftarrow t_2\,] \vdash^{2_\mathrm{p}} t_4 : A_4[x_2 \leftarrow t_2\,]$$

À ce stade, on dit que la substitution σ motive partiellement l'environnement Γ, ce qu'on abrège par $\sigma\ \widetilde{\mathrm{mot}}\ \Gamma$. Il faut alors m dérivations pour établir ce fait, avec m le nombre de variables motivées par σ.

Puis nous définissons la notion de *motivation partielle σ d'un type C par rapport à un environnement* Γ, abrégée en $\sigma\ \widetilde{\mathrm{mot}}_\Gamma\ C$: σ doit motiver partiellement Γ, et il doit exister un terme t tel que $\sigma(\Gamma) \vdash^{2_\mathrm{p}} t : \sigma(C)$. Pour établir $\sigma\ \widetilde{\mathrm{mot}}_\Gamma\ C$ il faut alors $m+1$ dérivations, avec m le nombre de variables motivées par σ.

Définition du système

$$\frac{}{[\,]\ \mathsf{wf}^{2_\mathrm{p}}}\ (\text{p-env}_1) \qquad \frac{\Gamma \vdash^{2_\mathrm{p}} A : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\ \mathsf{wf}^{2_\mathrm{p}}}\ (\text{p-env}_2)$$

$$\frac{\Gamma\ \mathsf{wf}^{2_\mathrm{p}}}{\Gamma \vdash^{2_\mathrm{p}} \mathrm{Prop} : \mathrm{Type}}\ (\text{p-ax}) \qquad \frac{\Gamma, x : A, \Gamma'\ \mathsf{wf}^{2_\mathrm{p}}}{\Gamma, x : A, \Gamma' \vdash^{2_\mathrm{p}} x : A}\ (\text{p-var})$$

$$\frac{\Gamma, x : A \vdash^{2_\mathrm{p}} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_\mathrm{p}} \lambda x^A.u : \forall x^A.B}\ (\text{p-abs}) \qquad \frac{\Gamma \vdash^{2_\mathrm{p}} u : \forall x^A.B \quad \Gamma \vdash^{2_\mathrm{p}} v : A}{\Gamma \vdash^{2_\mathrm{p}} u\ v : B[x \leftarrow v\,]}\ (\text{p-app})$$

$$\frac{\Gamma, x : A \vdash^{2_\mathrm{p}} B : \mathrm{Prop} \quad \sigma\ \widetilde{\mathrm{mot}}_\Gamma\ \forall x^A.B}{\Gamma \vdash^{2_\mathrm{p}} \forall x^A.B : \mathrm{Prop}}\ (\text{p-prod})$$

Résultats majeurs concernant λ_p^2

Les constantes o et $\top$ des calculs précédents sont définissables dans λ_p^2 par $o := \lambda A^{\mathrm{Prop}}.\lambda x^A.x$ et $\top := \forall A^{\mathrm{Prop}}.A \to A$ (lem. 3.4.1) :

$$\frac{\Gamma\ \mathsf{wf}^{2_\mathrm{p}}}{\Gamma \vdash^{2_\mathrm{p}} o : \top : \mathrm{Prop} : \mathrm{Type}}$$

Ainsi, en identifiant o et $\top$ de λ_t^2 par leur définition dans λ_p^2, on démontre immédiatement que λ_t^2 est un sous-système de λ_p^2 (thm. 3.4.3) : toute motivation totale est effectivement un cas particulier de motivation partielle.

Pour montrer que λ_p^2 est un sous-système de λ_t^2 (thm. 3.4.4), il suffit que $\sigma\ \widetilde{\mathrm{mot}}_\Gamma\ C$ implique $\rho \odot \sigma\, \mathrm{mot}_\Gamma\, C$ pour une certaine substitution ρ, c'est-à-dire qu'il faut pouvoir *compléter* une motivation partielle σ en une motivation totale $\rho \odot \sigma$. Un résultat similaire est déjà établi pour le système *pédagogique* du second ordre P-Prop^2 par (Colson et Michel, 2009, prop. 9).

Finalement λ_p^2, λ_t^2 et λ_e^2 étant tous équivalents, λ_p^2 vérifie les critères nécessaires lui permettant d'être qualifié de **sous-système pédagogique de CC** :

(i) λ_p^2 est un sous-système de CC ;

(ii) Si $\Gamma \vdash^{2_\mathrm{p}} t : C$ et $t \rightsquigarrow_\beta t'$, alors $\Gamma \vdash^{2_\mathrm{p}} t' : C$;

(iii) $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{2_\mathrm{p}}$ **si et seulement si** $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{\mathsf{c}}$ et il existe des termes $t_1, \ldots, t_n$ tels que

$$\vdash^{2_\mathrm{p}} t_1 : A_1 \quad \vdash^{2_\mathrm{p}} t_2 : A_2[x_1 \leftarrow t_1] \quad \ldots \quad \vdash^{2_\mathrm{p}} t_n : A_n[x_1, .., x_{n-1} \leftarrow t_1, .., t_{n-1}]$$

3.3.4 Système F pédagogique – P-Prop²

Pour établir le lien avec les travaux précédents, nous montrons que le système fonctionnel du second ordre pédagogique P-Prop² de Colson et Michel (2009) et λ_t^2 sont équivalents : P-Prop², λ_e^2, λ_t^2 et λ_p^2 sont alors des formalismes différents du même système. On récupère alors indirectement un plongement du calcul propositionnel du second ordre Prop², et par suite de λ^2, dans l'un quelconque des systèmes λ_e^2, λ_t^2 ou λ_p^2.

Définition du système P-Prop²

$$\frac{\vdash^{\mathsf{pf}} \sigma \cdot \Delta}{\Delta \vdash^{\mathsf{pf}} \boldsymbol{o} : \top}\,(\text{P-Ax}) \qquad \frac{x : F \in \Delta \quad \vdash^{\mathsf{pf}} \sigma \cdot \Delta}{\Delta \vdash^{\mathsf{pf}} x : F}\,(\text{P-Hyp})$$

$$\frac{\Delta, x : A \vdash^{\mathsf{pf}} u : B}{\Delta \vdash^{\mathsf{pf}} \boldsymbol{\lambda} x^A.u : A \rightarrow B}\,(\rightarrow_i) \qquad \frac{\Delta \vdash^{\mathsf{pf}} u : A \rightarrow B \quad \Delta \vdash^{\mathsf{pf}} v : A}{\Delta \vdash^{\mathsf{pf}} u\ v : B}\,(\rightarrow_e)$$

$$\frac{\Delta \vdash^{\mathsf{pf}} u : B \quad \alpha \notin \mathcal{V}(\Delta)}{\Delta \vdash^{\mathsf{pf}} \boldsymbol{\Lambda} \alpha.u : \forall \alpha.B}\,(\forall_i) \qquad \frac{\Delta \vdash^{\mathsf{pf}} u : \forall \alpha.B \quad \vdash^{\mathsf{pf}} \sigma \cdot V}{\Delta \vdash^{\mathsf{pf}} u\ V : [\alpha \leftarrow V] \cdot B}\,(\text{P-}\forall_e)$$

Correspondance des formalismes de P-Prop² et de λ_t^2

La correspondance entre les termes et types de P-Prop² et ceux de λ_t^2 est immédiate : les deux abstractions $\boldsymbol{\lambda} x^A.t$ et $\boldsymbol{\Lambda} \alpha.t$ sont traduites respectivement en $\lambda x^{[\![A]\!]}.[\![t]\!]$ et $\lambda \alpha^{\mathrm{Prop}}.[\![t]\!]$ où $[\![A]\!]$ est la traduction de A et $[\![t]\!]$ celle de t ; de même pour l'implication $A \rightarrow B$ et la quantification $\forall \alpha.B$ qui sont unifiées dans le formalisme de λ_t^2 en les produits, respectivement, $\forall x^{[\![A]\!]}.[\![B]\!]$ et $\forall \alpha^{\mathrm{Prop}}.[\![B]\!]$.

Les contextes de P-Prop² sont des ensembles $\Delta = \{x_1 : A_1, \ldots, x_n : A_n\}$ où les x_i sont des variables de termes et les A_i des types, tandis que les environnements de λ_t^2 sont des listes $\Gamma \equiv y_1 : B_1, \ldots, y_n : B_n$ où les y_i sont des variables de terme quand $B_i \not\equiv \mathrm{Prop}$ ou de type quand $B_i \equiv \mathrm{Prop}$. On montre alors que les environnements de λ_t^2 peuvent se comporter comme les contextes de P-Prop² : on peut se passer de mentionner les variables de type, et l'ordre d'apparence des variables de terme dans Γ n'importe pas. On peut alors étendre simplement la traduction en interprétant un contexte $\Delta = \{x_1 : A_1, \ldots, x_n : A_n\}$ comme l'environnement $[\![\Delta]\!] \equiv x_1 : [\![A_1]\!], \ldots, x_n : [\![A_n]\!]$, dont la partie contenant les variables libres (de type Prop) des A_i est *cachée*.

Les motivations σ de P-Prop² sont des substitutions associant à des variables de type des types de P-Prop² : $\vdash^{\mathsf{pf}} \sigma \cdot A$ signifie que le type A est motivable par σ dans P-Prop², c'est-à-dire qu'il existe un terme t tel que $\vdash^{\mathsf{pf}} t : \sigma \cdot A$. Par extension pour un contexte $\Delta = \{x_1 : A_1, \ldots, x_n : A_n\}$, $\vdash^{\mathsf{pf}} \sigma \cdot \Delta$ signifie que tous les A_i sont motivables par σ, i.e. $\vdash^{\mathsf{pf}} \sigma \cdot A_i$. Il n'y a aucune restriction sur σ : $\sigma(A_i)$ peut ne pas être clos et même introduire de nouvelles variables libres absentes de A_i. Les motivations de λ_t^2 semblent plus contraintes : $\sigma \,\mathrm{mot}_\Gamma\, A$ impose à σ d'associer des termes clos

à toutes les variables de l'environnement Γ. Nous allons voir que cette souplesse de P-Prop^2 n'est en fait qu'apparente puisque tout type A de P-Prop^2 motivable par une substitution σ ($\Vdash^{\mathsf{pf}} \sigma \cdot A$) l'est par τ ($\Vdash^{\mathsf{pf}} \tau \cdot A$) qui est la substitution constante qui associe $\top$ à toute variable de type. C'est d'ailleurs aussi le cas pour λ_e^2 (lem. A.1.1), et donc pour λ_t^2 et λ_p^2. Ces motivations τ sont appelées des *motivations triviales*.

Plongements mutuels entre P-Prop^2 et λ_t^2

On montre que P-Prop^2 peut être plongé dans λ_t^2 (thm. 3.5.13), i.e. :

$$\Delta \Vdash^{\mathsf{pf}} w : C \text{ si et seulement si } [\![\Delta]\!] \vdash^{2_t} [\![w]\!] : [\![C]\!].$$

L'implication de gauche à droite s'établit de la façon suivante : on part d'une dérivation quelconque de $\Delta \Vdash^{\mathsf{pf}} w : C$ qu'on transforme en une dérivation n'utilisant que la motivation triviale τ ; ce qui nous permet de passer de $\Vdash^{\mathsf{pf}} \tau \cdot \Delta$ à $[\![\Delta]\!]\, \mathsf{wf}^{2_t}$ par la réciproque du critère de Poincaré puisque τ est totale et close. Il suffit alors de faire correspondre les règles de λ_t^2 et de P-Prop^2 : $(\rightarrow_i)$ et $(\forall_i)$ de P-Prop^2 correspondent à la règle (t-abs) de λ_t^2, et $(\rightarrow_e)$ et $(\forall_e)$ correspondent à la règle (t-app).

Dans l'autre sens : de $[\![\Delta]\!]\, \mathsf{wf}^{2_t}$, avec $\Delta = \{x_1 : A_1, \ldots, x_n : A_n\}$, le critère de Poincaré nous donne des termes t_i tels que $\vdash^{2_t} t_i : [\![A_i]\!][x_1, \ldots, x_{i-1} \leftarrow t_1, \ldots, t_{i-1}]$. Or il s'avère que ces t_i sont images de termes t'_i de P-Prop^2 par $[\![\cdot]\!]$, donc aussi $\vdash^{2_t} [\![t'_i]\!] : [\![A_i]\!][x_1, \ldots, x_{i-1} \leftarrow [\![t'_1]\!], \ldots, [\![t'_{i-1}]\!]]$, d'où on en déduit par récurrence que $\Vdash^{\mathsf{pf}} t'_i : A_i[x_1, \ldots, x_{i-1} \leftarrow t'_1, \ldots, t'_{i-1}]$, c'est-à-dire $\Vdash^{\mathsf{pf}} \sigma \cdot A_i$ avec $\sigma(x_i) := t'_i$. Finalement, on parvient à passer de $[\![\Delta]\!]\, \mathsf{wf}^{2_t}$ à $\Vdash^{\mathsf{pf}} \sigma \cdot \Delta$ et il suffit encore une fois de faire correspondre les règles de λ_t^2 et de P-Prop^2.

La construction d'antécédents des termes des dérivations de λ_t^2 (sauf Prop et Type) par $[\![\cdot]\!]$ est unique et permet de montrer immédiatement le plongement inverse de λ_t^2 vers P-Prop^2 :

$$\Gamma \vdash^{2_t} w : C \text{ avec } C \not\equiv \mathrm{Prop}, \mathrm{Type} \text{ si et seulement si } [\![\Gamma]\!]^{-1} \Vdash^{\mathsf{pf}} [\![w]\!]^{-1} : [\![C]\!]^{-1}.$$

3.3.5 Indécidabilité de la vérification du typage

Le problème de la vérification du typage

L'utilisation effective de systèmes formels pédagogiques passe nécessairement par une *implémentation* sur machine. Au minimum l'ordinateur doit être capable de vérifier qu'une démonstration donnée d'un certain énoncé mathématique en est bien une. À travers l'isomorphisme de Curry-Howard, cela revient à décider si un terme t (la démonstration) est bien de type A (l'énoncé) dans un environnement Γ (les hypothèses), c'est-à-dire si $\Gamma \vdash t : A$ ou bien si $\Gamma \nvdash t : A$: c'est le problème de la *vérification du typage* (*type-checking*). Dans CC la vérification du typage est *décidable*, ce qui rend possible l'implémentation sur machine (voir Huet (1989)).

Malheureusement, aucun des formalismes P-Prop^2, λ_t^2 ou λ_p^2 n'admet de tel algorithme : la vérification du typage y est *indécidable*. Le terme de preuve t ne contient pas suffisamment d'information : il ne contient pas les exemples de Γ, ni les preuves que ces exemples satisfont les contraintes imposées par l'environnement Γ.

Avec ses motivations explicites, le formalisme de λ_e^2 est porteur d'informations supplémentaires. La vérification du typage pour un tel formalisme consiste à construire un algorithme qui étant donnés un environnement Γ, une substitution σ, un terme t et un type A décide si $\Gamma \vdash_\sigma t : A$ ou $\Gamma \nvdash_\sigma t : A$. Malgré cette information supplémentaire, la vérification du typage reste indécidable pour λ_e^2.

Le cas des formalismes proposés pour le second ordre pédagogique

La démonstration d'indécidabilité de la vérification du typage pour ces formalismes s'établit par réduction au problème d'*habitation de type*. Ce problème consiste à construire un terme t (une preuve) habitant un type A (une formule) donné(e) sous un environnement Γ (les hypothèses) donné. Urzyczyn (1997) montre que ce problème est indécidable pour Prop^2 et donc aussi pour P-Prop^2 et par suite pour λ_t^2, λ_p^2 et finalement λ_e^2.

On constate en effet qu'une dérivation de $\vdash^{2_t} A$: Prop contient la construction préalable d'un habitant du type A (lem. 3.2.12). De façon générale, une dérivation $\Gamma \vdash^{2_t} u : A$ nécessite la création d'une exemplification de Γ ainsi que de tous les sous-types apparaissant dans u ou A. On montre ainsi que vérifier le typage pour λ_t^2 est au moins aussi difficile qu'habiter un type dans λ_t^2. L'habitation étant indécidable pour λ_t^2, il s'ensuit que la vérification du typage l'est aussi.

Enfin on établit que l'existence d'un algorithme pour décider la vérification du typage pour λ_e^2 permet la construction d'un algorithme pour la vérification du typage pour λ_t^2. L'idée principale repose sur le constat suivant : vérifier le typage de $\Gamma \vdash_{\sigma}^{2_e} u : A$ revient à vérifier le typage de $\vdash_{[]}^{2_e} \lambda\Gamma.u : \forall\Gamma.A$ où l'information contenue dans σ est perdue. De même pour λ_t^2, et il suffit alors pour vérifier $\Gamma \vdash^{2_t} u : A$ de vérifier si $\vdash_{[]}^{2_e} \lambda\Gamma.u : \forall\Gamma.A$ puisque λ_t^2 et λ_e^2 sont équivalents. On est donc forcé de conclure qu'un tel algorithme de vérification du typage pour λ_e^2 n'existe pas, c'est-à-dire que le problème est indécidable, et ce malgré l'explicitation des motivations.

3.3.6 Remarques d'ordre général

Les formalismes P-Prop^2, λ_e^2, λ_t^2 et λ_p^2 sont tous équivalents et représentent des conceptions différentes de la même pratique de la pédagogie :

- λ_e^2 explicite et nécessite la conservation d'une même exemplification en cours de démonstration ;
- λ_t^2 est plus libéral et autorise le changement d'exemplification en cours de démonstration ;
- λ_p^2 et P-Prop^2 acceptent une pratique très souple avec l'utilisation d'exemplifications partielles mais qu'il est toujours possible de compléter.

L'équivalence de P-Prop^2 avec les nouveaux formalismes λ_e^2, λ_t^2 et λ_p^2 donne un poids supplémentaire à la définition formelle de sous-système pédagogique de CC : non seulement cette définition est sensée (puisque exemplifiée), mais en plus elle caractérise précisément les précédents travaux d'investigation sur la pédagogie formelle. De plus par l'utilisation du formalisme de CC, la définition ouvre sur une étude rigoureuse de la pédagogisation des systèmes du λ-cube de Barendregt (1991).

La plupart des résultats au second ordre sont obtenus d'abord dans λ_e^2 : le formalisme est plus précis et convient mieux à une étude formelle rigoureuse. D'autre part, le système λ_e^2 transcrit plus fidèlement la pratique de mathématiques pédagogiques puisque l'exemple est conservé tout au long d'une dérivation. Aussi, il semblerait que les relâchements menant de λ_e^2 à λ_t^2 puis à λ_p^2 soient assez généraux pour pouvoir être effectués de façon similaire dans des systèmes plus expressifs du λ-cube (quitte à complexifier l'expression des contraintes). Enfin les règles de λ_e^2 ont un nombre constant de prémisses, ce qui facilite leur présentation. C'est pourquoi nous suggérons de poursuivre les investigations ultérieures sur la pédagogie dans CC en utilisant des systèmes à motivations explicites, ainsi que la définition de *pseudo sous-système pédagogique* associée (faisant apparaître les motivations et les constantes o et $\top$).

3.4 Sous-système pédagogique d'ordre supérieur

À l'ordre supérieur, l'introduction de fonctions propositionnelles ou encore *constructeurs* (ou *opérateurs* ou *prédicats d'ordre supérieur*) augmente le pouvoir expressif du calcul propositionnel du second ordre. Dans CC la conjonction est un exemple de telle fonction propositionnelle définissable par

$$\wedge := \lambda A^{\mathrm{Prop}}.\lambda B^{\mathrm{Prop}}.\forall C^{\mathrm{Prop}}.(A \to B \to C) \to C$$

Dans un calcul pédagogique où les types doivent être exemplifiables et donc non vides, ces fonctions propositionnelles peuvent mal interagir les unes avec les autres. Par exemple les fonctions P et Q suivantes

$$P := \lambda f^{\mathrm{Prop}\to\mathrm{Prop}}.\forall A^{\mathrm{Prop}}.f\ A \qquad Q := \lambda B^{\mathrm{Prop}}.B$$

génèrent par application une formule $P\ Q$ qui se réduit sur un type vide

$$P\ Q \overset{*}{\rightsquigarrow}_{\beta} \forall A^{\mathrm{Prop}}.A$$

Remarquons en plus que, en tant que fonctions, ces deux constructeurs sont utiles : P peut être appliqué à $\lambda C^{\mathrm{Prop}}.\top$ pour produire le type habité $\forall A^{\mathrm{Prop}}.\top$; et Q peut être appliqué à $\top$.

Il convient donc d'accorder une attention toute particulière à ces nouveaux objets d'ordre supérieur.

3.4.1 Un calcul propositionnel d'ordre supérieur pédagogique – P-CP$^{\omega}$

Dans sa thèse Michel (2008) traite le cas du calcul propositionnel d'ordre supérieur de Girard (1972). La syntaxe de ce système est aussi libérale que celle de Prop^2 ou P-Prop^2 : il n'y a pas de règle de formation des types et la règle de β-conversion des constructeurs est non contrainte.

Le calcul propositionnel d'ordre supérieur

Les règles du calcul propositionnel d'ordre supérieur CP$^{\omega}$ sont les suivantes :

$$\frac{}{\Delta \overset{\mathsf{f}\omega}{\vdash} \top}\,(\mathrm{Ax}) \qquad \frac{}{\Delta, F \overset{\mathsf{f}\omega}{\vdash} F}\,(\mathrm{Hyp})$$

$$\frac{\Delta, A \overset{\mathsf{f}\omega}{\vdash} B}{\Delta \overset{\mathsf{f}\omega}{\vdash} A \to B}\,(\to_i) \qquad \frac{\Delta \overset{\mathsf{f}\omega}{\vdash} A \to B \quad \Delta \overset{\mathsf{f}\omega}{\vdash} A}{\Delta \overset{\mathsf{f}\omega}{\vdash} B}\,(\to_e)$$

$$\frac{\Delta \overset{\mathsf{f}\omega}{\vdash} B \quad \alpha^{\iota} \notin \mathcal{V}(\Delta)}{\Delta \overset{\mathsf{f}\omega}{\vdash} \forall \alpha^{\iota}.B}\,(\forall_i) \qquad \frac{\Delta \overset{\mathsf{f}\omega}{\vdash} \forall \alpha^{\iota}.B}{\Delta \overset{\mathsf{f}\omega}{\vdash} [\alpha^{\iota} \leftarrow V] \cdot B}\,(\forall_e)$$

$$\frac{\Delta \overset{\mathsf{f}\omega}{\vdash} A \quad A =_{\beta} B}{\Delta \overset{\mathsf{f}\omega}{\vdash} B}\,(=_{\beta})$$

où α^{ι} est une *variable de prédicat* de *genre* ι ; et dans la règle $(\forall_e)$ V est un prédicat de genre ι.

Les genres sont les types simples sur $\star$ où $\star$ est le genre des formules. Les prédicats et leur genre associé sont définis par récurrence [5] : les formules $\alpha^{\star}$, $\forall \alpha^{\iota}.B$ et $A \to B$, où A et B sont des

formules, sont des prédicats de genre $\star$; $\boldsymbol{\lambda}\alpha^{\iota}.u$ est de genre $\iota \Rightarrow \nu$ si u est de genre ν ; et $u\ v$ est de genre ι si u est de genre $\nu \Rightarrow \iota$ et v de genre ν.

La motivabilité d'un prédicat

En vue de pédagogiser CP$^{\omega}$, l'auteur réintroduit les contraintes mises en œuvre au second ordre, pour éventuellement en ajouter d'autres par la suite. Les contraintes sur les règles (Hyp), (Ax) et ($\forall_e$) de P-Prop^2 sont réintroduites :

$$\frac{\overset{\mathsf{f}\omega}{\vdash} \sigma \cdot \Delta}{\Delta \overset{\mathsf{f}\omega}{\vdash} \top}\ (\text{P-Ax}) \qquad \frac{F \in \Delta \quad \overset{\mathsf{f}\omega}{\vdash} \sigma \cdot \Delta}{\Delta \overset{\mathsf{f}\omega}{\vdash} F}\ (\text{P-Hyp}) \qquad \frac{\Delta \overset{\mathsf{f}\omega}{\vdash} \forall\alpha^{\iota}.B \quad \overset{\mathsf{f}\omega}{\vdash} \sigma \cdot V}{\Delta \overset{\mathsf{f}\omega}{\vdash} [\alpha^{\iota} \leftarrow V] \cdot B}\ (\text{P-}\forall_e)$$

Il est alors nécessaire d'étendre la motivabilité d'une formule et de définir la motivabilité $\overset{\mathsf{f}\omega}{\vdash} \sigma \cdot V$ d'un prédicat V de genre quelconque ι.

Comme précédemment, une formule F est motivable s'il existe une substitution σ associant à toute variable α^{ι} un prédicat de genre ι telle que $\overset{\mathsf{f}\omega}{\vdash} \sigma \cdot F$. Dans sa thèse, l'auteur définit la motivabilité d'un prédicat P de genre $\nu_1 \Rightarrow \ldots \Rightarrow \nu_n \Rightarrow \star$ comme la motivation de sa fermeture universelle $\forall\alpha_1^{\nu_1}.\ldots.\forall\alpha_n^{\nu_n}.P\ \alpha_1\ \ldots\ \alpha_n$. Ainsi le prédicat $\boldsymbol{\lambda}\alpha^{\star}.\alpha$ n'est pas motivable car sa fermeture universelle $\forall\beta^{\star}.(\boldsymbol{\lambda}\alpha^{\star}.\alpha)\ \beta$ se réduit en $\forall\beta^{\star}.\beta$ qui n'est motivable.

Cette définition de la motivation d'un prédicat est encouragée par l'exemple pathologique suivant :

1.	$\forall\alpha^{\star}.\gamma\ \alpha \overset{\mathsf{f}\omega}{\vdash} \forall\alpha^{\star}.\gamma\ \alpha$	(P-Ax)
2.	$\overset{\mathsf{f}\omega}{\vdash} \forall\gamma^{\star\Rightarrow\star}(\forall\alpha^{\star}.\gamma\ \alpha)\rightarrow(\forall\alpha^{\star}.\gamma\ \alpha)$	$(\rightarrow_i)$ et $(\forall_i)$
3.	$\overset{\mathsf{f}\omega}{\vdash} (\forall\alpha^{\star}.(\boldsymbol{\lambda}\beta^{\star}.\beta)\ \alpha)\rightarrow(\forall\alpha^{\star}.(\boldsymbol{\lambda}\beta^{\star}.\beta)\ \alpha)$	$(\forall_e)$
4.	$\overset{\mathsf{f}\omega}{\vdash} (\forall\alpha^{\star}.\alpha)\rightarrow(\forall\alpha^{\star}.\alpha)$	$(=_{\beta})$

C'est-à-dire qu'avec une règle ($\forall_e$) non contrainte on peut dériver $\bot\rightarrow\bot$, or $\bot := \forall\alpha^{\star}.\alpha$ n'est pas motivable. La contrainte sur (P-$\forall_e$) empêche ce cas puisque $\boldsymbol{\lambda}\beta^{\star}.\beta$ n'est pas motivable, donc le passage de la ligne 2 à la ligne 3 n'est plus autorisé.

La β-expansion

La règle de β-conversion autorise l'introduction de types vides par β-expansion :

1.	$\overset{\mathsf{f}\omega}{\vdash} \top$	(P-Ax)
2.	$\overset{\mathsf{f}\omega}{\vdash} (\boldsymbol{\lambda}\alpha^{\star}.\top)\ \bot$	$(=_{\beta})$

L'auteur propose de contraindre la relation de β-conversion de façon à ce que deux prédicats soient β-égaux si en plus tous les sous-prédicats dont ils sont composés sont motivables. Cette nouvelle relation d'équivalence est nommée β'-conversion, et la règle $(='_{\beta})$ l'utilisant remplace l'ancienne règle $(=_{\beta})$. La β'-conversion dépend alors de la syntaxe du calcul.

5. Les relations entre morphologie et syntaxe se complexifient : la définition de prédicats, objets de la morphologie, nécessite l'introduction de règles de typages similaires à celles formant la syntaxe du calcul propositionnel minimal sur l'implication.

L'extensionnalité sur les prédicats

L'auteur exhibe une formule G non motivable et telle que $\overset{\mathsf{f}\omega}{\vdash} G \to G$. Mais plutôt que de contraindre davantage le système afin de rendre $G \to G$ non dérivable, il décide d'étendre son pouvoir expressif pour permettre à G d'être motivée en introduisant une règle d'égalité extensionnelle sur les prédicats :

$$\frac{\Delta \overset{\mathsf{f}\omega}{\vdash} [\alpha^\iota \leftarrow A] \cdot F \quad \Delta \overset{\mathsf{f}\omega}{\vdash} A =_\iota B}{\Delta \overset{\mathsf{f}\omega}{\vdash} [\alpha^\iota \leftarrow B] \cdot F} \text{(Ext)}$$

où $A =_\iota B$ est l'égalité extensionnelle sur les prédicats A et B en tant que fonctions. Par exemple quand $\iota = \nu_1 \Rightarrow \ldots \Rightarrow \nu_n \Rightarrow \star$ alors $A =_\iota B$ dénote la formule

$$\forall \alpha_1^{\nu_1} \ldots . \forall \alpha_n^{\nu_n} . (A\ \alpha_1^{\nu_1}\ \ldots\ \alpha_n^{\nu_n}) \leftrightarrow (B\ \alpha_1^{\nu_1}\ \ldots\ \alpha_n^{\nu_n})$$

Quelques résultats d'intérêt

Le système résultant de ces modification, nommé P-CP$^\omega$ par l'auteur, satisfait le critère de Poincaré puisque toute les sous-formules et sous-prédicats apparaissant dans les dérivations sont motivables de façon uniforme par une même substitution (voir (Michel, 2008, prop. 3.4.19)). De plus les règles (P-Ax) et (P-Hyp) imposent naturellement au système de vérifier la réciproque du critère de Poincaré.

D'autre part, l'auteur construit un plongement du calcul propositionnel d'ordre supérieur classique (avec règle d'extensionnalité) dans P-CP$^\omega$. La traduction est à base de double négation (*double negation translation*) où le symbole d'absurdité est remplacé par une variable propositionnelle fraîche γ.

Cependant, il n'est pas indiqué si P-CP$^\omega$ est stable par normalisation des preuves. D'autre part, P-CP$^\omega$ n'étant pas un sous-système du calcul propositionnel d'ordre supérieur usuel CP$^\omega$ à cause de la règle d'extensionnalité (Ext), il n'hérite pas simplement de sa cohérence : celle-ci est assurée par l'existence d'un plongement du calcul propositionnel d'ordre supérieur classique (cohérent) dans P-CP$^\omega$.

Il n'est pas non plus évident d'exprimer le système fonctionnel associé à P-CP$^\omega$: le statut à donner à la règle d'extensionnalité (Ext) n'est pas immédiat.

3.4.2 Avec motivations totales explicites – λ_e^ω

On peut restreindre le formalisme de CC pour obtenir le calcul λ^ω représentant la version fonctionnelle du calcul propositionnel d'ordre supérieur CP$^\omega$ (voir sec. 4.1), et qui de fait est un sous-système de CC. Dans ce formalisme, les genres de CP$^\omega$ sont appelés des *ordres*[6], et sont des types simples sur Prop. Par exemple le prédicat du début de cette section exprimant la conjonction $\wedge$ est d'ordre Prop $\to$ Prop $\to$ Prop.

Nous allons expliquer les idées menant à la construction de λ_e^ω. À l'instant de la rédaction de cette thèse, il n'est pas connu s'il s'agit effectivement d'une version pédagogique de λ^ω avec motivations explicites : la stabilité par réduction n'est pas complètement prouvée et repose sur une conjecture.

6. En référence au travail initial de Girard (1972).

L'absence de β-expansion *sauvage*

Dans le formalisme de λ^ω la β-expansion *sauvage* ($=_\beta$) du formalisme libéral de CP$^\omega$ n'est pas autorisée. Dans la règle (ω-conv) suivante, le type A' remplaçant par β-conversion le type A doit être dérivable dans le système :

$$\frac{\Gamma \overset{\omega}{\vdash} u : A \quad A =_\beta A' \quad \Gamma \overset{\omega}{\vdash} A' : \mathrm{Prop}}{\Gamma \overset{\omega}{\vdash} u : A'} \ (\omega\text{-conv})$$

Ainsi si l'on parvient à empêcher la formation de types vides suffisamment tôt, il ne devrait pas être possible d'en introduire par β-expansion en utilisant la règle (ω-conv). Il n'apparaît donc pas nécessaire de redéfinir la β-conversion pour obtenir une version pédagogique de λ^ω.

Redéfinition de la motivabilité d'un prédicat

P-CP$^\omega$ n'autorise pas l'apparition d'occurrences du prédicat $\boldsymbol{\lambda}\alpha^\star.\alpha$. Pourtant ce prédicat n'est pas *dangereux* en lui-même, c'est son utilisation qui peut l'être. Cette idée de *dangerosité* peut être exprimée formellement comme l'opposé du concept d'utilité des fonctions dégagé au second ordre : un prédicat est dangereux s'il est inutile. Les prédicats dangereux devant être évités, seuls les prédicats utiles doivent être autorisés, constat servant de base à la définition de la motivabilité d'un prédicat d'ordre supérieur.

Formellement un prédicat P d'ordre $\mathcal{O} \equiv \mathcal{O}_1 \to \ldots \to \mathcal{O}_n \to \mathrm{Prop}$ est motivable par une substitution σ, abrégé en $\sigma\,\mathrm{mot}^{\mathcal{O}}(P)$, s'il existe des termes $u_1, \ldots, u_n$, un terme t et un terme R tels que

$$\overset{\omega_e}{\vdash_{[]}} t : R \qquad \text{avec} \qquad \sigma(P)\ u_1\ \ldots\ u_n \overset{*}{\rightsquigarrow}_\beta R \qquad \text{et} \qquad \overset{\omega_e}{\vdash_{[]}} u_i : \mathcal{O}_i$$

Autrement dit, un prédicat P est motivable par σ si $\sigma(P)$ peut être totalement appliqué à des termes u_i (d'autres prédicats) pour former une proposition R habitée (par t).

Remarquons que t habite un réduit R de $\sigma(P)\ u_1\ \ldots\ u_n$ et non $\sigma(P)\ u_1\ \ldots\ u_n$ directement. En effet dans le cas inverse, la construction du terme t dont le type est une application $\sigma(P)\ u_1\ \ldots\ u_n$ nécessite de passer par une règle de conversion où $\overset{\omega_e}{\vdash_{[]}} \sigma(P)\ u_1\ \ldots\ u_n : \mathrm{Prop}$ en serait une sous-dérivation. Ce type étant clos, dans un système pédagogique il est attendu qu'un habitant ait été construit et dérivé *a priori*. Or un tel habitant est exactement ce qui est recherché, nous entraînant dans un cercle vicieux. Pour briser cette circularité, il convient d'habiter un type $Q =_\beta \sigma(P)\ u_1\ \ldots\ u_n$, c'est-à-dire tel que $\overset{\omega_e}{\vdash_{[]}} t : Q$. Mais dans ce cas tout réduit R de Q devrait convenir, à supposer que le système soit stable par réduction (ce que nous souhaitons). Il paraît donc suffisant d'habiter un réduit R de $\sigma(P)\ u_1\ \ldots\ u_n$.

Remarquons aussi que les contraintes de λ_e^2 de la forme $\overset{2_e}{\vdash_{[]}} t : \sigma(A)$ sont un cas particulier de motivabilité lorsque le prédicat A est d'ordre Prop.

Non nécessité d'un principe d'extensionnalité

Le principe d'extensionnalité (Ext) proposé pour P-CP$^\omega$ n'est pas repris ici : la formule G non motivable sans extensionnalité et telle que $G \to G$ soit dérivable n'est déjà pas dérivable dans CP$^\omega$, donc *a fortiori* ne l'est pas non plus dans λ^ω.

Ainsi nous cherchons uniquement à contraindre le formalisme λ^ω : la β-conversion est inchangée, et aucune règle n'est ajoutée. Le système résultant λ_e^ω est alors naturellement un sous-système de CC, et hérite directement de la cohérence de CC et de la normalisation forte de ses λ-termes.

Définition du système λ_e^ω

$$\frac{}{[\,]\,\mathsf{wf}_{[\,]}^{\omega_e}}\,(\omega e\text{-env}_1)$$

$$\frac{\Gamma \vdash_\sigma^{\omega_e} A : \kappa \quad \vdash_{[\,]}^{\omega_e} a : A' \quad \sigma(A) \overset{*}{\rightsquigarrow}_\beta A' \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}_{\sigma::(x \mapsto a)}^{\omega_e}}\,(\omega e\text{-env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}_\sigma^{\omega_e}}{\Gamma \vdash_\sigma^{\omega_e} o : \top : \mathrm{Prop} : \mathrm{Type}}\,(\omega e\text{-ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}_\sigma^{\omega_e}}{\Gamma, x : A, \Gamma' \vdash_\sigma^{\omega_e} x : A}\,(\omega e\text{-var})$$

$$\frac{\Gamma, x : A \vdash_{\sigma::(x \mapsto a)}^{\omega_e} u : B : \mathrm{Prop}}{\Gamma \vdash_\sigma^{\omega_e} \lambda x^A.u : \forall x^A.B}\,(\omega e\text{-abs}_\star)$$

$$\frac{\Gamma, x : A \vdash_{\sigma::(x \mapsto a)}^{\omega_e} u : B : \mathrm{Type} \quad \Gamma \vdash_\sigma^{\omega_e} A : \mathrm{Type}}{\Gamma \vdash_\sigma^{\omega_e} \lambda x^A.u : A \to B}\,(\omega e\text{-abs}_\Box)$$

$$\frac{\Gamma \vdash_\sigma^{\omega_e} u : \forall x^A.B : \mathrm{Prop} \quad \Gamma \vdash_\sigma^{\omega_e} v : A \quad \sigma\,\mathrm{mot}^{\mathrm{Prop}}(B[x \leftarrow v])}{\Gamma \vdash_\sigma^{\omega_e} u\ v : B[x \leftarrow v]}\,(\omega e\text{-app}_\star)$$

$$\frac{\Gamma \vdash_\sigma^{\omega_e} u : A \to B : \mathrm{Type} \quad \Gamma \vdash_\sigma^{\omega_e} v : A \quad \sigma\,\mathrm{mot}^{B}(u\ v)}{\Gamma \vdash_\sigma^{\omega_e} u\ v : B}\,(\omega e\text{-app}_\Box)$$

$$\frac{\Gamma, x : A \vdash_{\sigma::(x \mapsto a)}^{\omega_e} B : \mathrm{Prop} \quad \sigma\,\mathrm{mot}^{\mathrm{Prop}}(\forall x^A.B)}{\Gamma \vdash_\sigma^{\omega_e} \forall x^A.B : \mathrm{Prop}}\,(\omega e\text{-prod}_\star)$$

$$\frac{\Gamma, x : A \vdash_{\sigma::(x \mapsto a)}^{\omega_e} B : \mathrm{Type} \quad \Gamma \vdash_\sigma^{\omega_e} A : \mathrm{Type}}{\Gamma \vdash_\sigma^{\omega_e} A \to B : \mathrm{Type}}\,(\omega e\text{-prod}_\Box)$$

$$\frac{\Gamma \vdash_\sigma^{\omega_e} u : A \quad A =_\beta A' \quad \Gamma \vdash_\sigma^{\omega_e} A' : \mathrm{Prop}}{\Gamma \vdash_\sigma^{\omega_e} u : A'}\,(\omega e\text{-conv})$$

Discussion sur les contraintes introduites

Les règles (ωe-env$_2$) et (ωe-prod$_\star$) reprennent les contraintes déjà présentes dans λ_e^2. Deux nouvelles contraintes sont introduites sur les règles (ωe-app$_\star$) et (ωe-app$_\Box$), chacune justifiée par l'un des contre-exemples suivants.

En l'absence de contrainte sur la règle (ωe-app$_\star$), on a la dérivation suivante :

1. $\vdash_{[\,]}^{\omega_e} u : \forall A^{\mathrm{Prop}\to\mathrm{Prop}}.(\forall B^{\mathrm{Prop}}.A\ B) \to (\forall B^{\mathrm{Prop}}.A\ B) : \mathrm{Prop}$
2. $\vdash_{[\,]}^{\omega_e} \lambda C^{\mathrm{Prop}}.C : \mathrm{Prop} \to \mathrm{Prop}$
3. $\vdash_{[\,]}^{\omega_e} u\ (\lambda C^{\mathrm{Prop}}.C) : (\forall B^{\mathrm{Prop}}.(\lambda C^{\mathrm{Prop}}.C)\ B) \to (\forall B^{\mathrm{Prop}}.(\lambda C^{\mathrm{Prop}}.C)\ B)$

avec $u := \lambda A^{\mathrm{Prop}\to\mathrm{Prop}}.\lambda H^{\forall B^{\mathrm{Prop}}.A\ B}.H$. Or le sous-type $\forall B^{\mathrm{Prop}}.(\lambda C^{\mathrm{Prop}}.C)\ B$ ne doit pas être motivable puisqu'il se réduit en $\forall B^{\mathrm{Prop}}.B$ qui n'est pas motivable. Autoriser une telle dérivation est certainement problématique pour la propriété de stabilité par réduction.

De même, sans contrainte sur la règle (ωe-app$_\Box$) on aurait :

1. $\vdash_{[\,]}^{\omega_e} \lambda A^{\mathrm{Prop}\to\mathrm{Prop}}.\forall B^{\mathrm{Prop}}.A\ B : (\mathrm{Prop} \to \mathrm{Prop}) \to \mathrm{Prop} : \mathrm{Type}$
2. $\vdash_{[\,]}^{\omega_e} \lambda C^{\mathrm{Prop}}.C : \mathrm{Prop} \to \mathrm{Prop}$
3. $\vdash_{[\,]}^{\omega_e} (\lambda A^{\mathrm{Prop}\to\mathrm{Prop}}.\forall B^{\mathrm{Prop}}.A\ B)\ (\lambda C^{\mathrm{Prop}}.C) : \mathrm{Prop}$

Or le type produit ne doit pas être motivable car il se réduit sur $\forall B^{\mathrm{Prop}}.B$ qui n'est pas motivable.

Notons que tous les prédicats et sous-prédicats apparaissant dans les contre-exemples ci-dessus sont motivables jusqu'à l'application de la règle non contrainte fautive en ligne 3.

Remarquons enfin qu'il n'est pas nécessaire d'introduire de contrainte sur la règle (ωe-prod$_\square$). En effet, tous les types de type Type sont des ordres de la forme $\mathcal{O}_1 \to \ldots \to \mathcal{O}_n \to \mathrm{Prop}$ et sont trivialement habités par $\lambda x_1^{\mathcal{O}_1}.\ldots.\lambda x_n^{\mathcal{O}_n}.\top$. Autrement dit, les types formés par (ωe-prod$_\star$) sont tous motivables.

Résultats majeurs concernant λ_e^ω

Complexification du lemme de substitution De même que λ_e^2, λ_e^ω vérifie le critère de Poincaré et sa réciproque. Cependant il n'est pas encore établit si λ_e^ω est stable par réduction : la démonstration usuelle repose sur un lemme de substitution, qui se complexifie encore par rapport à celui de λ_e^2.

En effet, à l'inverse de λ_e^2, on n'a pas dans λ_e^ω :

$$\text{Si } \Gamma, y : C, \Gamma' \vdash^{\omega_e}_{\sigma::(y \mapsto c)::\sigma'} d : D \text{ et } \Gamma \vdash^{\omega_e}_{\sigma} w : C, \text{ alors il existe une substitution } \rho \text{ telle que } \Gamma, \Gamma'[y \leftarrow w] \vdash^{\omega_e}_{\sigma::\rho} d[y \leftarrow w] : D[y \leftarrow w].$$

Il suffit pour s'en convaincre d'adapter le contre-exemple justifiant la contrainte sur la règle (ωe-app$_\star$) :

1. $A : \mathrm{Prop} \to \mathrm{Prop} \vdash^{\omega_e}_{(A \mapsto \lambda z^{\mathrm{Prop}}.\top)} \lambda H^{\forall B^{\mathrm{Prop}}.A\ B}.H : (\forall B^{\mathrm{Prop}}.A\ B) \to (\forall B^{\mathrm{Prop}}.A\ B) : \mathrm{Prop}$
2. $\vdash^{\omega_e}_{[]} \lambda C^{\mathrm{Prop}}.C : \mathrm{Prop} \to \mathrm{Prop}$
3. $\not\vdash^{\omega_e}_{[]} \lambda H^{\forall B^{\mathrm{Prop}}.(\lambda C^{\mathrm{Prop}}.C)\ B}.H : (\forall B^{\mathrm{Prop}}.(\lambda C^{\mathrm{Prop}}.C)\ B) \to (\forall B^{\mathrm{Prop}}.(\lambda C^{\mathrm{Prop}}.C)\ B)$

L'étape 3 est interdite car sinon par génération (lem. 4.2.7) on aurait le type $\forall B^{\mathrm{Prop}}.(\lambda C^{\mathrm{Prop}}.C)\ B$ dans l'environnement, ce qui n'est pas possible par le critère de Poincaré puisque ce type est vide donc non motivable.

De la même façon le contre-exemple justifiant la contrainte sur la règle (ωe-app$_\square$) peut être adapté ici.

Un lemme de substitution pour λ_e^ω En étudiant la démonstration de stabilité par réduction (prop. 4.2.22), on peut extraire un lemme de substitution (lem. 4.2.21) adapté à λ_e^ω et suffisant. L'énoncé de ce lemme est le suivant :

(i) si $\Gamma, y : C, \Gamma' \vdash^{\omega_e}_{\sigma::(y \mapsto c)::\sigma'} D : \mathcal{O}$, $\Gamma \vdash^{\omega_e}_{\sigma} w : C$, $\Gamma, \Gamma'[y \leftarrow w]\, \mathsf{wf}^{\omega_e}_{\sigma::\rho}$ et $\sigma::\rho\, \mathrm{mot}^{\mathcal{O}}(D[y \leftarrow w])$ alors $\Gamma, \Gamma'[y \leftarrow w] \vdash^{\omega_e}_{\sigma::\rho} D[y \leftarrow w] : \mathcal{O}$;

(ii) si $\Gamma, y : C, \Gamma' \vdash^{\omega_e}_{\sigma::(y \mapsto c)::\sigma'} d : D : \mathrm{Prop}$, $\Gamma \vdash^{\omega_e}_{\sigma} w : C$, $\Gamma, \Gamma'[y \leftarrow w]\, \mathsf{wf}^{\omega_e}_{\sigma::\rho}$ et $\sigma::\rho\, \mathrm{mot}^{\mathcal{O}}(D[y \leftarrow w])$ alors $\Gamma, \Gamma'[y \leftarrow w] \vdash^{\omega_e}_{\sigma::\rho} d[y \leftarrow w] : D[y \leftarrow w]$.

La différence notable est l'hypothèse $\sigma::\rho\, \mathrm{mot}^{\mathcal{O}}(D[y \leftarrow w])$: dans le cas (i) elle correspond à la contrainte de la règle (ωe-app$_\square$), tandis que dans le cas (ii) elle correspond à la contrainte de la règle (ωe-app$_\star$).

Si ce lemme de substitution pour λ_e^ω est vérifié, alors λ_e^ω est stable par réduction et est alors un sous-système pédagogique de CC avec motivations explicites. Dans l'attente, nous n'avons pas étudié le pouvoir expressif de λ_e^ω en comparaison avec celui de son parent λ^ω.

Chapitre 4

Conclusion et perspectives

4.1 Synthèse du travail effectué **43**
4.2 Perspectives **44**
4.2.1 Vers un Calcul des Constructions pédagogique 44
4.2.2 Sur la décidabilité du typage 47
4.3 Réflexions générales **48**
4.3.1 D'ordre technique 48
4.3.2 D'ordre conceptuel 48

4.1 Synthèse du travail effectué

Ce rapport décrit le cheminement effectué dans l'exploration vers un Calcul des Constructions pédagogique. Le travail est d'ordre conceptuel et nous nous sommes efforcés de motiver chacun des choix effectués.

Nous avons apporté un éclairage nouveau sur les systèmes formels et fonctionnels pédagogiques : tout d'abord est établie une définition formelle d'un sous-système pédagogique du Calcul des Constructions ; puis des exemples de tels systèmes du second ordre sont exposés, et enfin nous avons proposé un système pédagogique d'ordre supérieur.

Au second ordre, nous avons montré que notre définition s'applique très exactement au système de Colson et Michel (2009). De plus le formalisme des *pure type systems* utilisé autorise la description de nombreux systèmes d'une façon homogène : p. ex. les contraintes pédagogiques introduites au second ordre sont nécessairement répercutées à l'ordre supérieur ; inversement, une fois un Calcul des Constructions pédagogique établi, les versions pédagogiques des systèmes du λ-cube doivent apparaître par suppression de certaines règles et simplification des contraintes associées. C'est pourquoi il nous semble que notre objectif de traitement uniforme de l'étude de la pédagogie formelle est en partie atteint.

Lors de cette exploration, nous avons mis au jour un formalisme rendant les motivations explicites, en faisant apparaître les exemples des hypothèses dans les jugements. Ce type de formalisme nous semble tout à fait naturel pour l'expression des systèmes pédagogiques. D'autre part il autorise la formulation d'énoncés *méta*-mathématiques plus précis et intuitifs sur ces systèmes. Toutefois, nous avons montré qu'il n'était pas porteur de l'information nécessaire pour d'envisager directement une *implémentation* sur machine, en particulier car la vérification du typage y est indécidable.

Nous proposons de conclure ce rapport en indiquant quelques pistes pour guider d'éventuels travaux futurs sur l'étude de la pédagogie vers des systèmes plus forts, et éventuellement pour pallier les insuffisances constatées. Puis nous terminons sur quelques réflexions informelles et personnelles.

4.2 Perspectives

4.2.1 Vers un Calcul des Constructions pédagogique

La propriété de préservation du type d'un terme lors du calcul, que nous avons appelée propriété de stabilité par réduction (*subject reduction*), est une caractéristique fondamentale d'un système fonctionnel. Nous avons vu que son obtention dans les systèmes pédagogiques nécessite d'empêcher la formation de types vides au plus tôt.

Dans le formalisme des PTS utilisé, de nouveaux types (éventuellement vides) peuvent être créés par la règle de formation des types (prod), ou celle d'application (app). Pour ces deux règles, nous proposons d'établir un résumé des contraintes nécessaires afin d'éviter la création de tels types vides, d'abord pour les systèmes λ^2 et λ^ω, puis nous étendons l'étude au cas de λC. Par suite, nous sommes alors en mesure de proposer une version pédagogique du Calcul des Constructions.

Contraintes nécessaires pour la règle (prod)

Rappelons la règle de formation des types dans les PTS pour les systèmes du λ-cube :

$$\frac{\Gamma \vdash A : s_1 \quad \Gamma, x : A \vdash B : s_2 \quad (s_1, s_2, s_2) \in \mathcal{R}}{\Gamma \vdash \forall x^A.B : s_2} \text{(prod)}$$

Ci-après un tableau résumant les contraintes nécessaires : ✓ signifie que l'instance (s_1, s_2, s_2) de cette règle ne produit pas de type vide ; ✗ signifie que l'instance (s_1, s_2, s_2) peut produire un type vide et qu'il faut donc contraindre cette règle ; l'absence de marque signifie que l'instance de la règle n'appartient pas au système ; enfin les nombres en exposant indiquent le numéro de l'une des explications qui suit.

(s_1, s_2)	λ^2	λ^ω	λC
$(\star, \star)$	✓ [1]	✓ [1]	✗ [4]
$(\Box, \star)$	✗ [2]	✗ [2]	✗ [2]
$(\star, \Box)$			✓ [3]
$(\Box, \Box)$		✓ [3]	✓ [3]

1. x n'apparaît pas dans B : dans ce cas si B est motivable par b, alors $\forall x^A.B$ est motivable par $\lambda x^A.b$, et l'éventuelle vacuité de $\forall x^A.B$ n'est pas dû à cette règle (i.e. aucun nouveau type n'est produit ici).
2. Si la règle n'est pas contrainte on peut dériver le type vide $\forall A^{\text{Prop}}.A$ de type Prop.
3. Les éléments de type Type sont de la forme $\forall \vec{z}^{\vec{C}}.\text{Prop}$ et sont habités par $\lambda \vec{z}^{\vec{C}}.\top$.
4. Sans contrainte on pourrait dériver $\forall x^{\mathbb{N}}.x =_{\mathbb{N}} 0$ de type Prop. Notons que toutes les sous-formules sont motivables.

Contraintes nécessaires pour la règle (app)

Pour la règle (app) il convient d'étudier les sortes s_1 et s_2 des types de la fonction u et de son argument v :

$$\frac{\Gamma \vdash v : A : s_1 \quad \Gamma \vdash u : \forall x^A.B : s_2}{\Gamma \vdash u\ v : B[x \leftarrow v\,]}\ (\text{app})$$

Avec les cas suivants :

(s_1, s_2)	λ^2	λ^ω	λC
$(\star, \star)$	✓ [1]	✓ [1]	✗ [5]
$(\square, \star)$	✓ [1,2]	✗ [6]	✗ [6]
$(\star, \square)$			✗ [4]
$(\square, \square)$		✗ [3]	✗ [3]

1. Ce sont des preuves qui sont produites. D'autre part les types n'étant pas dépendants, x est remplacé par un terme qui n'apparaît pas dans B (i.e. $B[x \leftarrow v\,] \equiv B$) : aucun nouveau type n'est produit ici.
2. Dans λ_e^2, nécessairement $A \equiv \text{Prop}$ et donc v est motivable, d'où $B[x \leftarrow v\,]$ puisque B est motivable. Autrement dit le fait de typer v de type Prop dans un système pédagogique évite le problème de non stabilité par réduction du calcul du second ordre faiblement pédagogique.
3. Sans contrainte on aurait :

$$\vdash (\lambda A^{\text{Prop}\to\text{Prop}}.\forall B^{\text{Prop}}.A\ B)\ (\lambda C^{\text{Prop}}.C) : \text{Prop} \quad \overset{*}{\rightsquigarrow}_\beta \quad \vdash \forall B^{\text{Prop}}.B : \text{Prop}$$

   Notons que tous les sous-prédicats sont motivables.
4. Sans contrainte on aurait :

$$\vdash (\lambda x^{\mathbb{N}}.x =_{\mathbb{N}} 0)\ 1 : \text{Prop} \quad \overset{*}{\rightsquigarrow}_\beta \quad \vdash 1 =_{\mathbb{N}} 0 : \text{Prop}$$

   Notons que tous les sous-prédicats sont motivables.
5. Sans contrainte on aurait :

$$\vdash 1 : \mathbb{N} : \text{Prop} \qquad \vdash \lambda x^{\mathbb{N}}.\lambda H^{x=_{\mathbb{N}}0}.H : \forall x^{\mathbb{N}}.x =_{\mathbb{N}} 0 \to x =_{\mathbb{N}} 0 : \text{Prop}$$

$$\vdash (\lambda x^{\mathbb{N}}.\lambda H^{x=_{\mathbb{N}}0}.H)\ 1 : 1 =_{\mathbb{N}} 0 \to 1 =_{\mathbb{N}} 0 \quad \overset{*}{\rightsquigarrow}_\beta \quad \vdash \lambda H^{1=_{\mathbb{N}}0}.H : 1 =_{\mathbb{N}} 0 \to 1 =_{\mathbb{N}} 0$$

   Notons que tous les sous-prédicats sont motivables.
6. Sans contrainte on aurait :

$$\vdash \lambda C^{\text{Prop}}.C : \text{Prop} \to \text{Prop} : \text{Type}$$

$$\vdash \lambda A^{\text{Prop}\to\text{Prop}}.\lambda H^{\forall B^{\text{Prop}}.A\ B}.H : \forall A^{\text{Prop}\to\text{Prop}}.(\forall B^{\text{Prop}}.A\ B) \to (\forall B^{\text{Prop}}.A\ B) : \text{Prop}$$

$$\vdash (\lambda A^{\text{Prop}\to\text{Prop}}.\lambda H^{\forall B^{\text{Prop}}.A\ B}.H)\ (\lambda C^{\text{Prop}}.C) : (\forall B^{\text{Prop}}.(\lambda C^{\text{Prop}}.C)\ B) \to (\forall B^{\text{Prop}}.(\lambda C^{\text{Prop}}.C)\ B)$$

$$\overset{*}{\rightsquigarrow}_\beta \quad \vdash \lambda H^{\forall B^{\text{Prop}}.B}.H : (\forall B^{\text{Prop}}.B) \to (\forall B^{\text{Prop}}.B)$$

   Notons que tous les sous-prédicats sont motivables.

Expression d'un Calcul des Constructions pédagogique

Dès lors que l'on sait où placer les contraintes dans λC, on est en mesure de proposer une version pédagogique du Calcul des Constructions. Il convient tout d'abord d'étendre la définition de motivabilité d'un prédicat au cas des types dépendants :

Définition : motivation d'un prédicat avec types dépendants

◊ Le fait que *la substitution σ motive le prédicat P de type E*, noté $\sigma\,\mathrm{mot}^E(P)$, est défini par :

$$\begin{aligned}\sigma\,\mathrm{mot}^{\mathrm{Prop}}(P) &:= \text{il existe un terme } t \text{ et un type } R \text{ tels que } \vdash^{c_e}_{[]} t : R \text{ et } \sigma(P) \overset{*}{\rightsquigarrow}_\beta R\\ \sigma\,\mathrm{mot}^{\forall z^C.D}(P). &:= \text{il existe un terme } u \text{ tel que } \vdash^{c_e}_{[]} u : \sigma(C) \text{ et } \sigma{::}(z \mapsto u)\,\mathrm{mot}^D(P\ z).\end{aligned}$$

$\sigma\,\mathrm{mot}^{\forall z_1^{C_1}.\ldots.\forall z_n^{C_n}.\,\mathrm{Prop}}(P)$ dénote l'existence de termes $(u_i)_{1\leq i\leq n}$, d'un terme t et d'un type R tels que $\vdash^{c_e}_{[]} t : R$ avec $\sigma(P)\ \vec{u} \overset{*}{\rightsquigarrow}_\beta R$ et $\vdash^{c_e}_{[]} u_i : \sigma(C_i[z_1,\ldots,z_{i-1} \leftarrow u_1,\ldots,u_{i-1}])$.

Cette définition est sensée puisque dans CC tout prédicat possède un type E syntaxiquement de la forme $\forall \vec{z}^{\vec{C}}.\,\mathrm{Prop}$. De plus il est facile de vérifier que la motivation d'un prédicat de λ_e^ω en est un cas particulier.

Nous proposons enfin comme expression d'un Calcul des Constructions pédagogique avec motivations explicites le système λC_e suivant :

$$\frac{}{[\,]\ \mathsf{wf}^{c_e}_\sigma}\ (\text{ce-env}_1)$$

$$\frac{\Gamma \vdash^{c_e}_\sigma A : \kappa \quad \vdash^{c_e}_{[]} a : A' \quad \sigma(A) \overset{*}{\rightsquigarrow}_\beta A' \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\ \mathsf{wf}^{c_e}_{\sigma::(x\,\mapsto\,a)}}\ (\text{ce-env}_2)$$

$$\frac{\Gamma\ \mathsf{wf}^{c_e}_\sigma}{\Gamma \vdash^{c_e}_\sigma o : \top : \mathrm{Prop} : \mathrm{Type}}\ (\text{ce-ax})$$

$$\frac{\Gamma, x : A, \Gamma'\ \mathsf{wf}^{c_e}_\sigma}{\Gamma, x : A, \Gamma' \vdash^{c_e}_\sigma x : A}\ (\text{ce-var})$$

$$\frac{\Gamma, x : A \vdash^{c_e}_{\sigma::(x\,\mapsto\,a)} u : B : \kappa}{\Gamma \vdash^{c_e}_\sigma \lambda x^A.u : \forall x^A.B}\ (\text{ce-abs})$$

$$\frac{\Gamma \vdash^{c_e}_\sigma u : \forall x^A.B : \mathrm{Prop} \quad \Gamma \vdash^{c_e}_\sigma v : A \quad \sigma\,\mathrm{mot}^{\mathrm{Prop}}(B[x \leftarrow v])}{\Gamma \vdash^{c_e}_\sigma u\ v : B[x \leftarrow v]}\ (\text{ce-app}_\star)$$

$$\frac{\Gamma \vdash^{c_e}_\sigma u : \forall x^A.B : \mathrm{Type} \quad \Gamma \vdash^{c_e}_\sigma v : A \quad \sigma\,\mathrm{mot}^{B[x\leftarrow v]}(u\ v)}{\Gamma \vdash^{c_e}_\sigma u\ v : B[x \leftarrow v]}\ (\text{ce-app}_\Box)$$

$$\frac{\Gamma, x : A \vdash^{c_e}_{\sigma::(x\,\mapsto\,a)} B : \mathrm{Prop} \quad \sigma\,\mathrm{mot}^{\mathrm{Prop}}(\forall x^A.B)}{\Gamma \vdash^{c_e}_\sigma \forall x^A.B : \mathrm{Prop}}\ (\text{ce-prod}_\star)$$

$$\frac{\Gamma, x : A \vdash^{c_e}_{\sigma::(x\,\mapsto\,a)} B : \mathrm{Type}}{\Gamma \vdash^{c_e}_\sigma \forall x^A.B : \mathrm{Type}}\ (\text{ce-prod}_\Box)$$

$$\frac{\Gamma \vdash^{c_e}_\sigma u : A \quad A =_\beta A' \quad \Gamma \vdash^{c_e}_\sigma A' : \kappa}{\Gamma \vdash^{c_e}_\sigma u : A'}\ (\text{ce-conv})$$

Remarquons qu'il doit être possible de contracter l'expression des règles (ce-prod$_\star$) et (ce-prod$_\Box$) en une règle (ce-prod) avec une contrainte :

$$\frac{\Gamma, x : A \vdash^{\mathrm{ce}}_{\sigma::(x \mapsto a)} B : \kappa \quad \sigma\,\mathrm{mot}^{\kappa}(\forall x^A.B)}{\Gamma \vdash^{\mathrm{ce}}_{\sigma} \forall x^A.B : \kappa}\ (\text{ce-prod})$$

en étendant la définition de $\sigma\,\mathrm{mot}^E(P)$ au cas où $E \equiv \mathrm{Type}$, produisant une contrainte trivialement réalisée.

En revanche, il semble plus délicat de condenser l'expression des contraintes sur les règles (ce-app$_\star$) et (ce-app$_\Box$) : dans un cas on agit sur le terme $B[x \leftarrow v]$ tandis que dans l'autre on agit sur $u\ v$. Une formulation pédagogique générale reprenant exactement les (méta-)règles des PTS apparaît alors compromise.

4.2.2 Sur la décidabilité du typage

Comme nous l'avons montré, la vérification du typage pour les systèmes propositionnels pédagogiques du second ordre P-Prop^2, λ^2_t, λ^2_t et λ^2_e est indécidable : le symbolisme utilisé n'intègre pas dans les termes de preuve l'information nécessaire à la reconstruction d'une dérivation.

En analysant le système avec motivations explicites de λ^2_e, on peut voir les règles responsables de cette perte d'informations : la règle (e-abs) oublie le a motivant A ; et la règle (e-prod) ne conserve pas la motivation t du type formé.

$$\frac{\Gamma, x : A \vdash^{2\mathrm{e}}_{\sigma::(x \mapsto a)} u : B : \mathrm{Prop}}{\Gamma \vdash^{2\mathrm{e}}_{\sigma} \lambda x^A.u : \forall x^A.B}\ (\text{e-abs}) \qquad \frac{\Gamma, x : A \vdash^{2\mathrm{e}}_{\sigma::(x \mapsto a)} B : \mathrm{Prop} \quad \vdash^{2\mathrm{e}}_{[]} t : \sigma(\forall x^A.B)}{\Gamma \vdash^{2\mathrm{e}}_{\sigma} \forall x^A.B : \mathrm{Prop}}\ (\text{e-prod})$$

C'est la raison pour laquelle la règle (e-env$_2$) nécessite une pseudo-contrainte pour récupérer la motivation a construite *a priori* lors de la dérivation de $\Gamma \vdash^{2\mathrm{e}}_{\sigma} A : \kappa$:

$$\frac{\Gamma \vdash^{2\mathrm{e}}_{\sigma} A : \kappa \quad \vdash^{2\mathrm{e}}_{[]} a : \sigma(A) \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\ \mathsf{wf}^{2\mathrm{e}}_{\sigma::(x \mapsto a)}}\ (\text{e-env}_2)$$

C'est pourquoi nous proposons, à l'instar du λ-calcul typé qui *décore* les λ-termes purs par des types pour contrôler leur normalisation, d'annoter les types par des termes pour contrôler leur motivabilité. Par exemple au second ordre, on peut imaginer des règles avec de tels termes :

$$\frac{\Gamma_\sigma, x : A_a \vdash u : B_b : \mathrm{Prop}}{\Gamma_\sigma \vdash \lambda x^{A_a}.u : (\forall x^{A_a}.B_b)_{\sigma(\lambda x^{A_a}.u)}}\ (\text{abs}) \qquad \frac{\Gamma_\sigma, x : A_a \vdash B_b : \mathrm{Prop} \quad \vdash t : \sigma(\forall x^{A_a}.B_b)}{\Gamma_\sigma \vdash (\forall x^{A_a}.B_b)_t : \mathrm{Prop}}\ (\text{prod})$$

et aussi

$$\frac{\Gamma_\sigma \vdash A_a : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma_\sigma, x : A_a\ \mathsf{wf}}\ (\text{env}_2)$$

où les exemples sont annotés en indice et sont clos, et où Γ_σ est une abréviation pour $x_1 : A_{1a_1}, \ldots, x_n : A_{na_n}$.

Dans un tel formalisme, les termes devraient contenir l'information nécessaire afin de permettre la reconstruction d'une dérivation et ainsi autoriser la vérification du typage. Cependant à l'ordre supérieur, l'annotation d'un constructeur reste à établir : nous rappelons qu'un constructeur est motivable s'il est utile, et qu'il faut donc fournir des termes à appliquer à ce constructeur pour vérifier ce fait, termes qu'il convient de conserver dans l'annotation.

4.3 Réflexions générales

Les réflexions présentées dans cette dernière section sont très générales et ont été peu approfondies par l'auteur de ce rapport. Cependant elles apparaissent légitimes et peuvent servir de guide à des développements ultérieurs.

4.3.1 D'ordre technique

Sur les contraintes La contrainte nécessaire sur la règle d'application (ce-app$_\star$) pour l'obtention d'un Calcul des Constructions pédagogiques parait assez forte :

$$\frac{\Gamma \vdash^{\mathrm{ce}}_{\sigma} u : \forall x^A.B : \mathrm{Prop} \quad \Gamma \vdash^{\mathrm{ce}}_{\sigma} v : A \quad \sigma\,\mathrm{mot}^{\mathrm{Prop}}(B[x \leftarrow v\,])}{\Gamma \vdash^{\mathrm{ce}}_{\sigma} u\ v : B[x \leftarrow v\,]}\ (\text{ce-app}_\star)$$

En effet, $\forall x^A.B$ représente l'énoncé d'un théorème général que l'on souhaiterait utiliser en l'appliquant à $v : A$. Et il faut pour cela vérifier la motivabilité de $B[x \leftarrow v\,]$, ce qui, si $B[x \leftarrow v\,]$ est clos, revient à l'habiter. Or il est possible que ce soit justement ce que l'on souhaite obtenir. Il semble alors qu'il soit nécessaire d'effectuer une preuve directe (sans détour), potentiellement très longue : l'application du général au particulier apparaît délicate.

En l'absence d'une implémentation, nous n'avons pas été en mesure de vérifier si cela pouvait entraver l'utilisation effective d'un système pédagogique ou non.

Sur une éventuelle implémentation Au-delà du problème de la vérification du typage auquel nous proposons une solution, pour être utilisable une implémentation nécessiterait une étude théorique approfondie de la forme des preuves d'un système formel pédagogique. Par exemple, on peut imaginer que l'utilisateur effectue une preuve d'abord dans le système initial (non pédagogique), puis que celle-ci soit complétée, si possible, en une preuve pédagogique en demandant des exemples à l'utilisateur. Ces demandes d'interactions supplémentaires doivent être réduites au maximum : il convient alors d'étudier quel est l'ensemble minimal d'exemples nécessaires à cette reconstruction, et selon quels procédés, éventuellement automatiques.

4.3.2 D'ordre conceptuel

L'extension à des systèmes forts Dans l'éventualité où notre proposition de Calcul des Constructions pédagogique se révélerait correcte, la question de l'extension des résultats à des systèmes plus forts se pose naturellement. Si un Calcul des Constructions avec une hiérarchie d'univers prédicatifs CC^ω ne semble pas apporter *a priori* de difficultés conceptuelles nouvelles, il n'en va pas de même des types inductifs. En effet, l'ajout d'un mécanisme de définition de types inductifs nécessite de contraindre les types produits à être non vide d'une part, mais aussi à éviter les éventuels constructeurs inutiles d'autre part, ce qu'il convient de formaliser judicieusement.

En poussant la spéculation plus loin, on pourrait envisager l'étude de versions pédagogiques de systèmes inconsistants. Nous pensons en particulier au système Type : Type de Martin-Löf (1971) : il s'agit d'une instance particulière des *pure type systems* et les contraintes établies pour CC devraient pouvoir s'y transposer directement. Le résultat serait d'un intérêt philosophique particulier, que l'inconsistance persiste ou non.

Vers un développement autonome de la pédagogie formelle Jusqu'à présent, l'étude de la pédagogie formelle fait référence à un système de base. Notre définition de **sous-système** pédagogique du Calcul des Constructions n'y échappe pas. Cela comporte des avantages, comme le fait d'hériter de la cohérence du système initial, mais aussi des inconvénients, comme celui de ne pouvoir s'échapper des idées implicites liées au formalisme du système d'origine. Par exemple dans CC les environnements de typage sont implicitement universellement quantifiés, ce qui est traduit par le lemme usuel de substitution. Or dans les systèmes pédagogiques, les environnements de typages sont aussi existentiellement implicitement quantifiés, ce que le formalisme hérité ne représente pas, ce qui nous contraint à avoir recours à des énoncés *méta*-mathématiques plus techniques.

Il serait certainement utile de parvenir à dégager une notion de pédagogie indépendante de tout système actuel, autorisant une évolution complètement autonome de la *pédagogie formelle*, et débouchant potentiellement sur un formalisme tout à fait original.

Deuxième partie

Étude formelle

Chapitre 1

Présentation liminaire du formalisme

1.1 La pré-morphologie **53**
1.2 Résultats sur la pré-morphologie **56**
1.3 Définitions pré-syntaxiques **59**
1.4 Le calcul des constructions – CC **60**

Dérivation, sous-dérivation (stricte)

- ◊ Les définitions inductives induisent une structure arborescente : une *dérivation*.
- ◊ Une *sous-dérivation* d'une dérivation est alors un sous-arbre de la dérivation.
- ◊ Une *sous-dérivation stricte* d'une dérivation est une sous-dérivation qui n'est pas la dérivation elle-même.

Remarque

Dans la suite, les notations liées aux définitions inductives (p. ex. les relations de typage) pourront désigner à la fois une dérivation ou bien le fait qu'un objet satisfait ladite définition inductive.

1.1 La pré-morphologie

Nous travaillons dans le formalisme des *pure type systems* de Barendregt (1992), à quelques différences près, décelables dans la suite.

Pour plus de clarté, nous utilisons une syntaxe usuelle faisant apparaître explicitement les variables en tant qu'identificateurs (p. ex. $\lambda x^A.u$) et non pas, comme c'est le cas dans la syntaxe de De Bruijn de Bruijn (1972), comme position. On travaillera alors modulo renommage de variables liées (α-conversion) de façon implicite. Les résultats liminaires ci-après justifient et valident ce choix.

Termes bruts :

- ◊ les constantes Prop et Type sont des termes bruts ;
- ◊ les variables $x, y, \ldots$ sont des termes bruts ;
- ◊ si u et v sont des termes bruts, alors $u\ v$ est un terme brut ;
- ◊ si u et A sont des termes bruts et x une variable, alors $\lambda x^A.u$ et $\forall x^A.u$ sont des termes bruts.

Variables libres/liées/fraîches

◊ L'ensemble fini des *variables libres d'un terme brut* t, noté $\mathcal{V}(t)$ est défini par :

$$\begin{aligned}
\mathcal{V}(c) &:= \emptyset \text{ où } c \text{ est une des constantes} \\
\mathcal{V}(x) &:= \{x\} \\
\mathcal{V}(u\ v) &:= \mathcal{V}(u) \cup \mathcal{V}(v) \\
\mathcal{V}(\lambda x^A.u) &:= \mathcal{V}(A) \cup (\mathcal{V}(u) \backslash \{x\}) \\
\mathcal{V}(\forall x^A.B) &:= \mathcal{V}(A) \cup (\mathcal{V}(B) \backslash \{x\})
\end{aligned}$$

◊ Un terme sans variable libre est dit clos.

◊ L'ensemble fini des variables liées d'un terme brut t, noté $\mathcal{B}(t)$ est défini par :

$$\begin{aligned}
\mathcal{B}(c) &:= \emptyset \text{ où } c \text{ est une constante} \\
\mathcal{B}(x) &:= \emptyset \\
\mathcal{B}(u\ v) &:= \mathcal{B}(u) \cup \mathcal{B}(v) \\
\mathcal{B}(\lambda x^A.u) &:= \{x\} \cup \mathcal{B}(A) \cup \mathcal{B}(u) \\
\mathcal{B}(\forall x^A.B) &:= \{x\} \cup \mathcal{B}(A) \cup \mathcal{B}(B)
\end{aligned}$$

◊ L'ensemble (infini) des variables fraîches pour un terme brut t, noté $\mathcal{F}(t)$ est défini par :

$$x \in \mathcal{F}(t) :\leftrightarrow x \notin \mathcal{V}(t) \cup \mathcal{B}(t)$$

◊ On étend ces notations au cas d'un ensemble fini de termes bruts T :

$$\mathcal{V}(T) := \bigcup_{t \in T} \mathcal{V}(t) \quad et \quad \mathcal{B}(T) := \bigcup_{t \in T} \mathcal{B}(t) \quad et \quad \mathcal{F}(T) := \bigcap_{t \in T} \mathcal{F}(t)$$

Substitution

◊ Une *substitution* σ est une liste finie de couples $[(x_1 \mapsto u_1), \ldots, (x_n \mapsto u_n)]$ où les x_i sont des variables et les u_i des termes bruts. La substitution vide est notée $[\,]$.

◊ Le *domaine de la substitution* σ, noté $\mathrm{dom}(\sigma)$, est l'ensemble fini de variables $\{x_1, \ldots, x_n\}$.

◊ On note $\sigma\backslash_y$ la substitution σ dans laquelle tous les couples de la forme $(y \mapsto v)$ sont supprimés (v un terme brut quelconque). On étend la notation au cas d'un ensemble fini Y de variables dont on supprime les couples associés : $\sigma\backslash_Y$.

◊ :: est l'*opérateur de concaténation* entre deux substitutions. Lors des concaténations, on pourra confondre un couple $(y \mapsto v)$ avec la substitution contenant uniquement ce couple $[(y \mapsto v)]$.

◊ L'ensemble fini des *variables libres d'une substitution* est :

$$\mathcal{V}(\sigma) := \bigcup_{(x \mapsto v) \in \sigma} \mathcal{V}(v)$$

Application d'une substitution à un terme

◊ L'*application d'une substitution* σ à un terme t, notée $\sigma(t)$ est définie par :

$$\begin{aligned}
\sigma(c) &:= c \text{ où } c \text{ est une constante} \\
[\,](y) &:= y \\
(x \mapsto u) :: \sigma(x) &:= u \\
(x \mapsto u) :: \sigma(y) &:= \sigma(y) \\
\sigma(u\ v) &:= \sigma(u)\ \sigma(v) \\
\sigma(\lambda x^A.u) &:= \lambda x^{\sigma(A)}.\sigma \setminus_x (u) \\
\sigma(\forall x^A.B) &:= \forall x^{\sigma(A)}.\sigma \setminus_x (B)
\end{aligned}$$

◊ On pourra aussi noter $t[x_1, \ldots, x_n \leftarrow u_1, \ldots, u_n\,]$ pour $\sigma(t)$ si $\sigma := [(x_1 \mapsto u_1), \ldots, (x_n \mapsto t_n)]$.

α-conversion

◊ L'α-conversion $\equiv$ est une relation entre deux termes bruts définie par les règles suivantes :

$$\frac{c \text{ est une constante}}{c \equiv c}\ (\alpha\text{-co}) \qquad \frac{}{x \equiv x}\ (\alpha\text{-var}) \qquad \frac{u \equiv w \quad v \equiv t}{u\ v \equiv w\ t}\ (\alpha\text{-app})$$

$$\frac{A \equiv B \quad u[x \leftarrow z\,] \equiv v[y \leftarrow z\,] \quad z \in \mathcal{F}(u, v)}{\lambda x^A.u \equiv \lambda y^B.v}\ (\alpha\text{-abs})$$

$$\frac{A \equiv C \quad B[x \leftarrow z\,] \equiv D[y \leftarrow z\,] \quad z \in \mathcal{F}(B, D)}{\forall x^A.B \equiv \forall y^C.D}\ (\alpha\text{-prod})$$

β-réduction

◊ La β-réduction $\rightsquigarrow_\beta$ est une relation entre deux termes bruts définie par les règles suivantes :

$$\frac{}{(\lambda x^A.u)\ v \rightsquigarrow_\beta u[x \leftarrow v\,]}\ (\beta\text{-red})$$

$$\frac{u \rightsquigarrow_\beta u'}{u\ v \rightsquigarrow_\beta u'\ v}\ (\beta\text{-app}_\text{g}) \qquad \frac{v \rightsquigarrow_\beta v'}{u\ v \rightsquigarrow_\beta u\ v'}\ (\beta\text{-app}_\text{d})$$

$$\frac{u \rightsquigarrow_\beta u'}{\lambda x^A.u \rightsquigarrow_\beta \lambda x^A.u'}\ (\beta\text{-abs}) \qquad \frac{A \rightsquigarrow_\beta A'}{\lambda x^A.u \rightsquigarrow_\beta \lambda x^{A'}.u}\ (\beta\text{-abst})$$

$$\frac{B \rightsquigarrow_\beta B'}{\forall x^A.B \rightsquigarrow_\beta \forall x^A.B'}\ (\beta\text{-prod}) \qquad \frac{A \rightsquigarrow_\beta A'}{\forall x^A.B \rightsquigarrow_\beta \forall x^{A'}.B}\ (\beta\text{-prodt})$$

◊ La fermeture transitive de $\rightsquigarrow_\beta$ est notée $\overset{+}{\rightsquigarrow}_\beta$; la fermeture transitive et réflexive (selon $\equiv$) est notée $\overset{*}{\rightsquigarrow}_\beta$; et la fermeture transitive, réflexive et symétrique (selon $\equiv$) est notée $=_\beta$. Les relations symétriques correspondantes sont $\leftsquigarrow_\beta$, $\overset{+}{\leftsquigarrow}_\beta$ et $\overset{*}{\leftsquigarrow}_\beta$.

Forme normale/réductibilité/normalisation forte

◊ Un terme u est *réductible* s'il existe un terme u' avec $u \rightsquigarrow_\beta u'$.

◊ Un terme u est *irréductible* ou en *forme normale* s'il n'est pas réductible.

◊ Une *séquence de réductions* d'un terme brut t est une suite de termes bruts $(u_i)_{0 \leq i \leq n}$ telle que $u_0 \equiv t$ et $u_i \rightsquigarrow_\beta u_{i+1}$.

◊ Un terme brut t est *normalisable* s'il existe une séquence de réductions finie $(u_i)_{0 \leq i \leq n}$ de t avec u_n en forme normale.

◊ Un terme brut t est *fortement normalisable* s'il existe un entier k tel que pour toute séquence de réductions $(u_i)_{0 \leq i \leq n}$ de t, $n \leq k$.

1.2 Résultats sur la pré-morphologie

Lemme 1.2.1. *La relation $\equiv$ entre deux termes bruts est une relation d'équivalence.*

Preuve laissée au lecteur :
- la réflexivité utilise une récurrence sur le nombre de symboles utilisés pour écrire le terme brut considéré, puis par cas sur le terme brut ;
- transitivité et symétrie se montrent par récurrence sur les dérivations ($u \equiv v$ et $v \equiv w$).□

Remarques

Dans la suite, on identifiera des termes bruts α-convertibles : on quotiente l'ensemble des termes par cette relation d'équivalence.

Ainsi, et par abus de notation, on pourra considérer $\equiv$ comme une égalité syntaxique.

Lemme 1.2.2.

Si $B[x \leftarrow v] \equiv \forall y_1^{D_1}.\ldots.y_n^{D_n}.c$, où c est une constante, alors un des deux cas suivants se produit :
(i) $B \equiv \forall y_1^{E_1}.\ldots.y_i^{E_i}.x$ et $v \equiv \forall y_{i+1}^{D_{i+1}}.\ldots.y_n^{D_n}.c$ avec $E_k[x \leftarrow v] \equiv D_k$ $(0 \leq k \leq i \leq n)$;
(ii) $B \equiv \forall y_1^{E_1}.\ldots.y_n^{E_n}.c$ avec $E_k[x \leftarrow v] \equiv D_k$ $(0 \leq k \leq n)$.

Preuve par récurrence sur n. □

Substitution adaptée à un terme

◊ Une substitution σ est dite *adaptée* à un terme brut t si

$$\mathcal{V}(\sigma) \cap \mathcal{B}(t) = \emptyset$$

Lemme 1.2.3. *Pour toute substitution σ et tout terme brut t, il existe un terme brut t' tel que $t \equiv t'$ et σ adaptée à t'.*

Preuve par récurrence sur le terme brut t. □

Lemme 1.2.4. *Si $t \equiv t'$, et σ adaptée à t et t', alors $\sigma(t) \equiv \sigma(t')$.*

Preuve par récurrence sur la dérivation de $t \equiv t'$. □

Remarque (importante)

On ne mentionnera plus le fait qu'une substitution est adaptée ou non : on effectuera des α-conversions sur les termes à la volée si nécessaire avant d'appliquer des substitutions.

Composition de substitutions

◊ On note $\sigma \odot \rho$ la substitution *composition* des deux substitutions σ et ρ définie par :

$$\sigma \odot \rho := \rho^\sigma :: \sigma\backslash_{\mathrm{dom}(\rho)}$$

où

$$\begin{aligned} [\,]^\sigma &:= [\,] \\ ((y \mapsto v) :: \tau)^\sigma &:= (y \mapsto \sigma(v)) :: \tau^\sigma \end{aligned}$$

Lemme 1.2.5.

Pour tout terme brut t, pour toutes substitutions σ et ρ,

$$\sigma \odot \rho(t) \equiv \sigma(\rho(t))$$

De plus, $\mathrm{dom}(\sigma \odot \rho) = \mathrm{dom}(\sigma) \cup \mathrm{dom}(\rho)$.

Preuve immédiate par récurrence sur le terme brut t et la définition de $\sigma \odot \rho$ (similaire à (Colson et Michel, 2009, lem. 6)). □

Lemme 1.2.6. *Pour toutes substitutions σ et ρ, si* $\mathrm{dom}(\rho) \cap \mathcal{V}(\sigma) = \emptyset$ *et* $\mathrm{dom}(\sigma) \cap \mathrm{dom}(\rho) = \emptyset$*, alors*

$$\forall t \ \sigma \odot \rho(t) \equiv \rho^\sigma \odot \sigma(t)$$

avec ρ^σ défini comme précédemment.

Preuve par récurrence sur le terme brut t. □

Remarque Le lemme précédent est une généralisation du résultat usuel suivant :

$A[x \leftarrow u][y \leftarrow v] \equiv A[y \leftarrow v[x \leftarrow u]][x \leftarrow u]$ avec $y \notin \mathcal{V}(u)$ est $x \not\equiv y$.

Théorème 1.2.7 (Church-Rosser).

Si $u =_\beta v$ alors il existe un terme w tel que $u \overset{*}{\rightsquigarrow}_\beta w \overset{*}{\leftsquigarrow}_\beta v$.

Preuve similaire à celle de (Barendregt, 1992, cor. 2.3.8). □

Lemme 1.2.8. *Si $u =_\beta v$, alors pour toute substitution σ on a $\sigma(u) =_\beta \sigma(v)$.*

Preuve On montre le résultat intermédiaire suivant

Si $u \rightsquigarrow_\beta v$, alors pour toute substitution σ on a $\sigma(u) \rightsquigarrow_\beta \sigma(v)$.

par récurrence sur le terme brut u puis par cas sur $u \rightsquigarrow_\beta v$ (utilise le lem. 1.2.6).

Par Church-Rosser (thm.1.2.7) on a un terme w tel que $u \overset{*}{\rightsquigarrow}_\beta w \overset{*}{\leftsquigarrow}_\beta v$, d'où $\sigma(u) \overset{*}{\rightsquigarrow}_\beta \sigma(w) \overset{*}{\leftsquigarrow}_\beta \sigma(v)$, donc $\sigma(u) =_\beta \sigma(v)$. □

β-équivalence de substitutions

◊ On notera $\sigma =_\beta \sigma'$ si

$$\forall x \in \mathrm{dom}(\sigma) \quad \sigma(x) =_\beta \sigma'(x)$$

Lemme 1.2.9. *Si $\sigma =_\beta \sigma'$, alors pour tout terme brut t on a $\sigma(t) =_\beta \sigma'(t)$.*

Preuve par récurrence structurelle sur le terme brut t. □

Lemme 1.2.10.

(i) $\forall x^A.B =_\beta \forall x^C.D$ *si et seulement si* $A =_\beta C$ *et* $B =_\beta D$ *;*

(ii) $\lambda x^A.u =_\beta \lambda x^C.v$ *si et seulement si* $A =_\beta C$ *et* $u =_\beta v$.

Preuve immédiate par Church-Rosser (thm. 1.2.7) et la définition de $\rightsquigarrow_\beta$ et $=_\beta$. □

Plus long chemin vers la forme normale

◊ Pour tout terme fortement normalisable, on peut définir la *longueur de la plus longue séquence de réductions* vers sa forme normale $\|\cdot\|$ par :

$$
\begin{aligned}
\|c\| &:= 0 \text{ où } c \text{ est une constante}\\
\|x\| &:= 0\\
\|\lambda x^A.u\| &:= \|A\| + \|u\|\\
\|\forall x^A.B\| &:= \|A\| + \|B\|\\
\|u\ v\| &:= \begin{cases} 1 + \max(\max\limits_{u \rightsquigarrow_\beta u'} \|u'\ v\|, \max\limits_{v \rightsquigarrow_\beta v'} \|u\ v'\|, \|w[x \leftarrow v\,]\|) & \text{si } u \equiv \lambda x^A.w\\ 1 + \max(\max\limits_{u \rightsquigarrow_\beta u'} \|u'\ v\|, \max\limits_{v \rightsquigarrow_\beta v'} \|u\ v'\|) & \text{sinon}\end{cases}
\end{aligned}
$$

Remarques Pour que cette définition soit correcte, il faut que :

- $\rightsquigarrow_\beta$ soit à *branchement fini*, c'est-à-dire que pour tout terme brut u, il doit exister un nombre fini de termes u' tels que $u \rightsquigarrow_\beta u'$ (c'est le cas plus généralement pour le lambda-calcul pur) ;
- que le terme considéré soit fortement normalisable.

La définition revient à considérer toutes les réductions possibles, et calculer la plus grande des longueurs.

Lemme 1.2.11. *Si* u *est fortement normalisable alors tout* $t \in \mathcal{S}(u)$ *est fortement normalisable.*

Preuve

Tout séquence de réductions de t implique une séquence de réductions de u de même longueur. Or toutes les séquences de réductions de u sont de longueur bornée. □

Lemme 1.2.12. *Pour tout terme brut* u *fortement normalisable, si* $u \rightsquigarrow_\beta u'$ *alors* $\|u\| > \|u'\|$.

Preuve par récurrence sur $u \rightsquigarrow_\beta u'$. □

Lemme 1.2.13. *Pour tous termes bruts* u *et* v *fortement normalisables,* $\|u\ v\| \geq \|u\| + \|v\|$.

Preuve

Soit $u \rightsquigarrow_\beta u_1 \rightsquigarrow_\beta \ldots \rightsquigarrow_\beta u_n$ une plus longue séquence de réductions de u de longueur n, et $v \rightsquigarrow_\beta v_1 \rightsquigarrow_\beta \ldots \rightsquigarrow_\beta v_m$ une plus longue séquence de réductions de v de longueur m. Alors $u\ v \rightsquigarrow_\beta u_1\ v \rightsquigarrow_\beta \ldots \rightsquigarrow_\beta u_n\ v \rightsquigarrow_\beta u_n\ v_1 \rightsquigarrow_\beta \ldots \rightsquigarrow_\beta u_n\ v_m$ est une séquence de réductions de $u\ v$ de longueur $n + m$, et donc la plus longue séquence de réductions de $u\ v$ est au moins de longueur $n + m$. □

1.3 Définitions pré-syntaxiques

Environnement

◊ Un *environnement* Γ est une liste finie de couples $x_1 : A_1, \ldots, x_n : A_n$ où les x_i sont des variables et les A_i des termes bruts. L'environnement vide s'écrit $[\,]$ ou ne s'écrit pas.

◊ On note $\mathrm{dom}(\Gamma)$ l'ensemble fini des variables $\{x_1, \ldots, x_n\}$.

Sous-environnement initial

◊ Si $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$ est un environnement, on note par $\Gamma_{<i}$ le *sous-environnement* $x_1 : A_1, \ldots, x_{i-1} : A_{i-1}$. De même on note $\Gamma_{\leq i}$ pour le sous-environnement $x_1 : A_1, \ldots, x_i : A_i$.

◊ Lorsque seul le fait d'être un sous-environnement importe, on note $\Gamma' \preccurlyeq \Gamma$ si $\Gamma' \equiv \Gamma_{\leq i}$ pour un certain $i \leq n$.

Inclusion d'environnement

◊ On notera $\Gamma \subseteq \Gamma'$ si tous les couples de l'environnement Γ sont dans Γ' (et pas nécessairement dans le même ordre).

Sous-substitution initiale

◊ Si $\sigma \equiv [(x_1 \mapsto u_1), \ldots, (x_n \mapsto u_n)]$ est une substitution, on note $\sigma_{<i}$ la *sous-substitution* $[(x_1 \mapsto u_1), \ldots, (x_{i-1} \mapsto u_{i-1})]$. De même on note $\sigma_{\leq i}$ pour la *sous-substitution* $[(x_1 \mapsto u_1), \ldots, (x_i \mapsto u_i)]$.

◊ Lorsque seul le fait d'être une *sous-substitution initiale* importe, on note $\sigma' \preccurlyeq \sigma$ si $\sigma' \equiv \sigma_{\leq i}$ pour un certain $i \leq n$.

Inclusion de substitution

◊ On notera $\sigma \subseteq \sigma'$ si tous les couples de la substitution σ sont dans σ' (et pas nécessairement dans le même ordre).

Sous-termes

◊ L'ensemble des *sous-termes* $\mathcal{S}(t)$ d'un terme t est défini par :

- $t \in \mathcal{S}(t)$;
- si $t \in \mathcal{S}(w)$ alors : $t \in \mathcal{S}(w\ v)$, $t \in \mathcal{S}(v\ w)$, $t \in \mathcal{S}(\lambda x^A.w)$, $t \in \mathcal{S}(\lambda x^w.u)$, $t \in \mathcal{S}(\forall x^A.w)$, $t \in \mathcal{S}(\forall x^w.B)$.

◊ On étend la notation à un ensemble de termes T, noté $\mathcal{S}(T)$ par :

$$\mathcal{S}(T) = \bigcup_{t \in T} \mathcal{S}(t)$$

◊ On étend aussi la notation à un environnement $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$ par :

$$\mathcal{S}(\Gamma) = \bigcup_{1 \leq i \leq n} \mathcal{S}(A_i)$$

1.4 Le calcul des constructions – CC

Conventions d'écriture générales

◊ On notera $\kappa, \kappa', \ldots, \kappa_1, \kappa_2, \ldots$ pour désigner l'une des deux constantes Prop ou Type.

◊ Dans $\forall x^A.B$, lorsque $x \notin \mathcal{V}(B)$, on pourra noter $A \to B$.

◊ Les lieurs $\forall$ et λ ont la priorité la plus faible ; p. ex. $\forall x^A.\forall y^B.C$ représente $\forall x^A.(\forall y^B.(C))$, et $A \to B \to C$ représente $A \to (B \to C)$.

◊ L'application est associative à droite, c'est-à-dire que $u\ v\ w$ représente $(u\ v)\ w$.

◊ On notera $\forall \vec{x}^{\vec{A}}.B$ pour $\forall x_1^{A_1}.\forall x_2^{A_2}.\ldots.\forall x_n^{A_n}.B$ en considérant que $\vec{x}$ représente implicitement les n variables $x_1, \ldots, x_n$ et $\vec{A}$ les n termes $A_1, \ldots, A_n$ dans cet ordre. Si $n = 0$, alors $\forall \vec{x}^{\vec{A}}.B \equiv B$.

◊ On notera $A[\vec{x} \leftarrow \vec{u}\,]$ pour $A[x_1, \ldots, x_n \leftarrow u_1, \ldots, u_n\,]$ en considérant que $\vec{x}$ représente implicitement les n variables $x_1, \ldots, x_n$ et $\vec{u}$ les n termes $u_1, \ldots, u_n$ dans cet ordre. Si $n = 0$, alors $A[\vec{x} \leftarrow \vec{u}\,] \equiv A$.

◊ On notera $A\ \vec{u}$ pour $A\ u_1\ u_2\ \ldots\ u_n$ en considérant que $\vec{u}$ représente implicitement les n termes $u_1, \ldots, u_n$ dans cet ordre. Si $n = 0$, alors $A\ \vec{u} \equiv A$.

◊ On écrira $\Gamma\,\mathsf{wf}^\star$ et $\Gamma \vdash^\star w : C$ pour désigner de façon générique les relations d'un quelconque sous-système de CC étudié dans la suite (i.e. $\vdash^\star$ désigne à la fois $\vdash^{\mathsf{c}}$, $\vdash^{2_{\mathrm{e}}}$, etc.).

◊ On notera $\Gamma \vdash^\star A_1 : A_2 : \ldots : A_n$ pour désigner $\Gamma \vdash^\star A_1 : A_2$, ..., et $\Gamma \vdash^\star A_{n-1} : A_n$.

◊ Une règle de typage de la forme

$$\frac{\mathcal{P}}{\Gamma \vdash^\star A_1 : A_2 : \ldots : A_n}$$

désigne l'ensemble des règles

$$\frac{\mathcal{P}}{\Gamma \vdash^\star A_1 : A_2} \qquad \ldots \qquad \frac{\mathcal{P}}{\Gamma \vdash^\star A_{n-1} : A_n}$$

où $\mathcal{P}$ dénote la liste des prémisses de la règle.

1.4.1 Définition du système

Règles d'inférence de CC

◊ On définit la relation $\vDash^{\mathsf{c}}$ entre un environnement et deux termes bruts, ainsi que la relation wf^{c} sur les environnements par les règles suivantes :

$$\frac{}{[\,]\,\mathsf{wf}^{\mathsf{c}}}\ (\text{c-env}_1) \qquad \frac{\Gamma \vDash^{\mathsf{c}} A : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^{\mathsf{c}}}\ (\text{c-env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^{\mathsf{c}}}{\Gamma \vDash^{\mathsf{c}} \mathrm{Prop} : \mathrm{Type}}\ (\text{c-ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^{\mathsf{c}}}{\Gamma, x : A, \Gamma' \vDash^{\mathsf{c}} x : A}\ (\text{c-var})$$

$$\frac{\Gamma, x : A \vDash^{\mathsf{c}} u : B : \kappa}{\Gamma \vDash^{\mathsf{c}} \lambda x^A.u : \forall x^A.B}\ (\text{c-abs}) \qquad \frac{\Gamma \vDash^{\mathsf{c}} u : \forall x^A.B \quad \Gamma \vDash^{\mathsf{c}} v : A}{\Gamma \vDash^{\mathsf{c}} u\ v : B[x \leftarrow v\,]}\ (\text{c-app})$$

$$\frac{\Gamma, x : A \vDash^{\mathsf{c}} B : \kappa}{\Gamma \vDash^{\mathsf{c}} \forall x^A.B : \kappa}\ (\text{c-prod}) \qquad \frac{\Gamma \vDash^{\mathsf{c}} u : A \quad A =_\beta A' \quad \Gamma \vDash^{\mathsf{c}} A' : \kappa}{\Gamma \vDash^{\mathsf{c}} u : A'}\ (\text{c-conv})$$

Remarque En toute rigueur, il serait nécessaire de donner une règle pour l'α-conversion :

$$\frac{\Gamma \vDash^{\mathsf{c}} u : A \quad A \equiv A'}{\Gamma \vDash^{\mathsf{c}} u : A'}\ (\text{c-}\alpha\text{-conv})$$

D'après les remarques précédentes (ch. 1), puisqu'on identifie deux termes α-convertibles, on utilisera cette règle de façon implicite.

1.4.2 Résultats sur la syntaxe

Les résultats suivants sont bien connus et se trouvent pour la plupart dans l'ouvrage introductif du calcul des constructions de Coquand (1985), et généralisés aux *pure type systems* par Barendregt (1992).

Lemme 1.4.1.

(i) Si $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^{\mathsf{c}}$, alors pour tout i, $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$ et $x_i \not\equiv x_j$ pour tout $i \neq j$;

(ii) Si $x_1 : A_1, \ldots, x_n : A_n \vDash^{\mathsf{c}} w : C$, alors pour tout i, $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$ et $x_i \not\equiv x_j$ pour tout $i \neq j$ et $\mathcal{V}(w, C) \subseteq \{x_1, \ldots, x_n\}$.

Preuve usuelle par récurrence structurelle sur la dérivation (similaire à (Barendregt, 1992, lem. 5.2.8)). □

Lemme 1.4.2. *Si $\Gamma\,\mathsf{wf}^{\mathsf{c}}$ ou $\Gamma \vDash^{\mathsf{c}} w : C$, alors $\mathrm{Type} \notin \mathcal{S}(\Gamma)$ et $\mathrm{Type} \notin \mathcal{S}(w)$.*

Preuve par récurrence structurelle sur la dérivation. □

Lemme 1.4.3 (validité des environnements).

(i) si $\Gamma\,\mathsf{wf}^{\mathsf{c}}$, alors pour tout environnement $\Gamma' \preccurlyeq \Gamma$, $\Gamma'\,\mathsf{wf}^{\mathsf{c}}$ en est une sous-dérivation ;

(ii) si $\Gamma \vDash^{\mathsf{c}} w : C$*, alors pour tout environnement* $\Gamma' \preccurlyeq \Gamma$*,* $\Gamma' \,\mathsf{wf}^{\mathsf{c}}$ *en est une sous-dérivation stricte.*

Preuve usuelle par récurrence structurelle sur la dérivation (similaire à (Coquand, 1985, prop. 1)). □

Lemme 1.4.4 (validité des types des environnements).

Si $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{\mathsf{c}}$*, alors pour tout* i *il existe* κ *tel que* $x_1 : A_1, \ldots, x_i : A_i \vDash^{\mathsf{c}} A_{i+1} : \kappa$ *en est une sous-dérivation stricte.*

Preuve usuelle par récurrence sur la taille de l'environnement (utilise lem. 1.4.3, similaire à (Coquand, 1985, cor. de prop. 1)). □

Lemme 1.4.5 (affaiblissement). *Si* $\Gamma \vDash^{\mathsf{c}} w : C$*,* $\Gamma' \,\mathsf{wf}^{\mathsf{c}}$ *et* $\Gamma \subseteq \Gamma'$*, alors* $\Gamma' \vDash^{\mathsf{c}} w : C$*.*

Preuve usuelle par récurrence structurelle sur la dérivation de $\Gamma \vDash^{\mathsf{c}} w : C$ (similaire à celle de (Coquand, 1985, prop. 3)) :

(c-abs) $$\frac{\Gamma, x : A \vDash^{\mathsf{c}} u : B : \kappa}{\Gamma \vDash^{\mathsf{c}} \lambda x^A.u : \forall x^A.B}$$

De $\Gamma, x : A \vDash^{\mathsf{c}} u : B$, on a $\Gamma, x : A \,\mathsf{wf}^{\mathsf{c}}$ comme sous-dérivation stricte (lem. 1.4.3), puis $\Gamma \vDash^{\mathsf{c}} A : \kappa$ comme sous-dérivation (lem. 1.4.4). D'où l'on peut appliquer l'hypothèse de récurrence pour obtenir $\Gamma' \vDash^{\mathsf{c}} A : \kappa$, puis par (c-env$_2$) on obtient $\Gamma', x : A \,\mathsf{wf}^{\mathsf{c}}$.

On peut alors appliquer l'hypothèse de récurrence sur $\Gamma, x : A \vDash^{\mathsf{c}} u : B : \kappa$ pour obtenir $\Gamma', x : A \vDash^{\mathsf{c}} u : B : \kappa$, et finalement le résultat par la règle (c-abs). □

Lemme 1.4.6 (unicité du type). *Si* $\Gamma \vDash^{\mathsf{c}} w : C$ *et* $\Gamma \vDash^{\mathsf{c}} w : C'$*, alors* $C =_\beta C'$*.*

Preuve usuelle par récurrence structurelle sur les deux dérivations (similaire à celles de (Coquand, 1985, prop. 2) ou de (Barendregt, 1992, lem. 5.2.21)).

(*, c-conv) $$\frac{\Gamma \vDash^{\mathsf{c}} w : A \quad A =_\beta C' \quad \Gamma \vDash^{\mathsf{c}} C' : \kappa}{\Gamma \vDash^{\mathsf{c}} w : C'}$$ avec $\Gamma \vDash^{\mathsf{c}} w : C$ obtenu par ailleurs.

Par hypothèse de récurrence, on a $C =_\beta A$ et donc par transitivité $C =_\beta C'$.

(c-var, c-var) Ce cas utilise le lemme 1.4.1.

(c-app, c-app) $$\frac{\Gamma \vDash^{\mathsf{c}} u : \forall x^A.B \quad \Gamma \vDash^{\mathsf{c}} v : A}{\Gamma \vDash^{\mathsf{c}} u\ v : B[x \leftarrow v]} \quad , \quad \frac{\Gamma \vDash^{\mathsf{c}} u : \forall x^C.D \quad \Gamma \vDash^{\mathsf{c}} v : C}{\Gamma \vDash^{\mathsf{c}} u\ v : D[x \leftarrow v]}$$

Par hypothèse de récurrence $\forall x^A.B =_\beta \forall x^C.D$, d'où $B =_\beta D$ (lem. 1.2.10) puis $B[x \leftarrow v] =_\beta D[x \leftarrow v]$ (lem.1.2.8). □

Lemme 1.4.7 (de substitution).

(i) si $\Gamma, y : C, \Gamma' \,\mathsf{wf}^{\mathsf{c}}$ *et* $\Gamma \vDash^{\mathsf{c}} w : C$*, alors* $\Gamma, \Gamma'[y \leftarrow w] \,\mathsf{wf}^{\mathsf{c}}$*;*

(ii) si $\Gamma, y : C, \Gamma' \vDash^{\mathsf{c}} t : D$ *et* $\Gamma \vDash^{\mathsf{c}} w : C$*, alors* $\Gamma, \Gamma'[y \leftarrow w] \vDash^{\mathsf{c}} t[y \leftarrow w] : D[y \leftarrow w]$*.*

Preuve usuelle par récurrence structurelle sur la dérivation (similaire à (Coquand, 1985, prop. 4) ou (Barendregt, 1992, lem. 5.2.11)).

(c-var) Il y a trois cas à considérer selon que l'on extrait de l'environnement une variable située avant y, qui est y, ou après y. Le dernier cas se traite par simple application de l'hypothèse de récurrence sur la prémisse. Les deux autres sont :

- $\dfrac{\Gamma, x : A, \Gamma', y : C, \Gamma'' \,\mathsf{wf}^{\mathsf{c}}}{\Gamma, x : A, \Gamma', y : C, \Gamma'' \vDash^{\mathsf{c}} x : A}$

  Par hypothèse de récurrence on a $\Gamma, x : A, \Gamma', \Gamma''[y \leftarrow w\,]\,\mathsf{wf}^{\mathsf{c}}$, puis par (c-var) on obtient bien le résultat $\Gamma, x : A, \Gamma', \Gamma''[y \leftarrow w\,] \vDash^{\mathsf{c}} x : A$ (puisque $y \notin \mathcal{V}(A)$ par lem. 1.4.1).

- $\dfrac{\Gamma, y : C, \Gamma' \,\mathsf{wf}^{\mathsf{c}}}{\Gamma, y : C, \Gamma' \vDash^{\mathsf{c}} y : C}$

  L'hypothèse de récurrence donne $\Gamma, \Gamma'[y \leftarrow w\,]\,\mathsf{wf}^{\mathsf{c}}$, et par affaiblissement (lem. 1.4.5) sur $\Gamma \vDash^{\mathsf{c}} w : C$, on obtient bien $\Gamma, \Gamma'[y \leftarrow w\,] \vDash^{\mathsf{c}} w : C$ (comme $y \notin \mathcal{V}(C)$ par lem. 1.4.1 alors $C \equiv C[y \leftarrow w\,]$).

(c-app) $\dfrac{\Gamma, y : C, \Gamma' \vDash^{\mathsf{c}} u : \forall x^A.B \quad \Gamma, y : C, \Gamma' \vDash^{\mathsf{c}} v : A}{\Gamma, y : C, \Gamma' \vDash^{\mathsf{c}} u\ v : B[x \leftarrow v\,]}$

Par hypothèse de récurrence sur les deux prémisses et utilisation de la règle (c-app) on a $\Gamma, \Gamma'[y \leftarrow w\,] \vDash^{\mathsf{c}} (u\ v)[y \leftarrow w\,] : B[y \leftarrow w\,][x \leftarrow v[y \leftarrow w\,]\,]$. Or comme $x \notin \mathcal{V}(w)$ (x désigne une variable liée), alors $B[y \leftarrow w\,][x \leftarrow v[y \leftarrow w\,]\,] \equiv B[x \leftarrow v\,][y \leftarrow w\,]$ (lem. 1.2.6).

(c-conv) utilise le lemme 1.2.8. □

Lemme 1.4.8 (génération). *Soit $\Gamma \vDash^{\mathsf{c}} t : T$, un de ces cas se présente :*

(i) si $t \equiv \mathrm{Prop}$, alors $T =_\beta \mathrm{Type}$;

(ii) si $t \equiv x$, alors il existe $(x : A) \in \Gamma$ avec $T =_\beta A$;

(iii) si $t \equiv \lambda x^A.u$, alors il existe B et κ tels que $\Gamma, x : A \vDash^{\mathsf{c}} u : B : \kappa$ est une sous-dérivation stricte avec $T =_\beta \forall x^A.B$;

(iv) si $t \equiv u\ v$, alors il existe A et B tels que $\Gamma \vDash^{\mathsf{c}} u : \forall x^A.B$ et $\Gamma \vDash^{\mathsf{c}} v : A$ sont des sous-dérivations strictes avec $T =_\beta B[x \leftarrow v\,]$;

(v) si $t \equiv \forall x^A.B$, alors il existe κ tel que $\Gamma, x : A \vDash^{\mathsf{c}} B : \kappa$ est une sous-dérivation stricte avec $T =_\beta \kappa$.

Preuve usuelle par récurrence structurelle sur la dérivation (similaire à (Barendregt, 1992, lem. 5.2.13)). □

Lemme 1.4.9 (correction des types).

Si $\Gamma \vDash^{\mathsf{c}} w : C$, alors $C \equiv \mathrm{Type}$ ou bien il existe κ tel que $\Gamma \vDash^{\mathsf{c}} C : \kappa$.

Preuve usuelle par récurrence structurelle sur la dérivation (similaire à (Coquand, 1985, prop. 5) ou (Barendregt, 1992, cor. 5.2.14)).

(c-var) $\dfrac{\Gamma, x : A, \Gamma' \,\mathsf{wf}^{\mathsf{c}}}{\Gamma, x : A, \Gamma' \vDash^{\mathsf{c}} x : A}$

De $\Gamma \vDash^{\mathsf{c}} A : \kappa$ (lem. 1.4.4) par affaiblissement (lem. 1.4.5) on a bien $\Gamma, x : A, \Gamma' \vDash^{\mathsf{c}} A : \kappa$.

(c-app) $\dfrac{\Gamma \vDash^{\mathsf{c}} u : \forall x^A.B \quad \Gamma \vDash^{\mathsf{c}} v : A}{\Gamma \vDash^{\mathsf{c}} u\ v : B[x \leftarrow v\,]}$

Par hypothèse de récurrence, $\Gamma \vDash^{\mathsf{c}} \forall x^A.B : \kappa'$, et par génération (lem. 1.4.8) on a $\Gamma, x : A \vDash^{\mathsf{c}} B : \kappa$, et enfin par substitution (lem. 1.4.7) on obtient bien $\Gamma \vDash^{\mathsf{c}} B[x \leftarrow v\,] : \kappa$.

(c-prod) $\dfrac{\Gamma, x : A \vDash^{\mathsf{c}} B : \kappa}{\Gamma \vDash^{\mathsf{c}} \forall x^A.B : \kappa}$

Comme $\Gamma\,\mathsf{wf}^{\mathsf{c}}$ (lem. 1.4.3), le résultat s'obtient par la règle (c-ax) quand $\kappa \equiv \mathrm{Prop}$. □

Réduction dans l'environnement

◊ Soient $\Gamma \equiv x_1 : A_1, \ldots, x_i : A_i, \ldots, x_n : A_n$ et $\Gamma' \equiv x_1 : A_1, \ldots, x_i : A'_i, \ldots, x_n : A_n$ deux environnements. On étend la relation de β-réduction aux environnements : $\Gamma \rightsquigarrow_\beta \Gamma'$ si $A_i \rightsquigarrow_\beta A'_i$ pour un certain i.

◊ De même, on étend les définitions de $=_\beta$, $\overset{+}{\rightsquigarrow}_\beta$, $\overset{*}{\rightsquigarrow}_\beta$, $\leftsquigarrow_\beta$, $\overset{+}{\leftsquigarrow}_\beta$ et $\overset{*}{\leftsquigarrow}_\beta$.

Lemme 1.4.10. *Si $\Gamma \vDash^{\mathsf{c}} w : C$ avec $\Gamma \rightsquigarrow_\beta \Gamma'$ et $\Gamma'\,\mathsf{wf}^{\mathsf{c}}$, alors $\Gamma' \vDash^{\mathsf{c}} w : C$.*

Preuve usuelle par récurrence structurelle sur la dérivation de $\Gamma \vDash^{\mathsf{c}} w : C$ (similaire à à (Coquand, 1985, prop. 7) ou (Barendregt, 1992, thm. 5.2.15)).

(c-var) $\dfrac{\Gamma_1, x : A, \Gamma_2\,\mathsf{wf}^{\mathsf{c}}}{\Gamma_1, x : A, \Gamma_2 \vDash^{\mathsf{c}} x : A}$ Seul le cas où $A \rightsquigarrow_\beta A'$ est non immédiat.

Par hypothèse $\Gamma_1, x : A', \Gamma_2\,\mathsf{wf}^{\mathsf{c}}$ et par (c-var) $\Gamma_1, x : A', \Gamma_2 \vDash^{\mathsf{c}} x : A'$.

Or $\Gamma_1 \vDash^{\mathsf{c}} A : \kappa$ (lem. 1.4.4), d'où par affaiblissement (lem. 1.4.5) $\Gamma_1, x : A', \Gamma_2 \vDash^{\mathsf{c}} A : \kappa$. Comme de plus $A' =_\beta A$, alors par (c-conv) on a bien $\Gamma_1, x : A', \Gamma_2 \vDash^{\mathsf{c}} x : A$. □

Théorème 1.4.11 (stabilité par réduction). *Si $\Gamma \vDash^{\mathsf{c}} t : C$ et $t \rightsquigarrow_\beta t'$, alors $\Gamma \vDash^{\mathsf{c}} t' : C$.*

Preuve usuelle par récurrence structurelle sur la dérivation puis par cas sur la définition de $\rightsquigarrow_\beta$ (similaire à (Coquand, 1985, prop. 7) ou (Barendregt, 1992, thm. 5.2.15)).

(c-abs) $\dfrac{\Gamma, x : A \vDash^{\mathsf{c}} u : B : \kappa}{\Gamma \vDash^{\mathsf{c}} \lambda x^A.u : \forall x^A.B}$

On a deux cas :

- $u \rightsquigarrow_\beta u'$: il suffit d'appliquer l'hypothèse de récurrence sur les prémisses.
- $A \rightsquigarrow_\beta A'$: comme $\Gamma \vDash^{\mathsf{c}} A : \kappa$ est une sous-dérivation stricte (lem. 1.4.4) alors par hypothèse de récurrence $\Gamma \vDash^{\mathsf{c}} A' : \kappa$, et donc $\Gamma, x : A'\,\mathsf{wf}^{\mathsf{c}}$ (c-env$_2$) d'où $\Gamma, x : A' \vDash^{\mathsf{c}} u : B : \kappa$ (lem. 1.4.10), et finalement (c-abs) permet de conclure.

(c-app) $\dfrac{\Gamma \vDash^{\mathsf{c}} u : \forall x^A.B \quad \Gamma \vDash^{\mathsf{c}} v : A}{\Gamma \vDash^{\mathsf{c}} u\ v : B[x \leftarrow v\,]}$

On a trois cas :

- $u \rightsquigarrow_\beta u'$: il suffit d'appliquer l'hypothèse de récurrence sur la première prémisse puis (c-app).
- $v \rightsquigarrow_\beta v'$: par hypothèse de récurrence sur la seconde prémisse et (c-app) on a $\Gamma \vDash^{\mathsf{c}} u\ v' : B[x \leftarrow v'\,]$.

  Or comme $\Gamma \vDash^{\mathsf{c}} \forall x^A.B : \kappa'$ (lem. 1.4.9), par génération (lem. 1.4.8) $\Gamma, x : A \vDash^{\mathsf{c}} B : \kappa$ puis par substitution (lem. 1.4.7) $\Gamma \vDash^{\mathsf{c}} B[x \leftarrow v\,] : \kappa$. Et puisque $B[x \leftarrow v'\,] =_\beta B[x \leftarrow v\,]$ (lem. 1.2.9) alors par (c-conv) on a bien $\Gamma \vDash^{\mathsf{c}} u\ v' : B[x \leftarrow v\,]$.

- $u \equiv \lambda x^C.w$ et $u\ v \rightsquigarrow_\beta w[x \leftarrow v\,]$:
  Par génération (lem. 1.4.8) sur la première prémisse on a

$$\Gamma, x : C \vdash^{\mathsf{c}} w : D$$

  avec $\forall x^A.B =_\beta \forall x^C.D$, d'où $A =_\beta C$ et $B =_\beta D$ (lem. 1.2.10).
  On a ainsi $\Gamma \vdash^{\mathsf{c}} C : \kappa$ (lem. 1.4.3 puis lem. 1.4.4) et donc par (c-conv) sur $\Gamma \vdash^{\mathsf{c}} v : A$ aussi

$$\Gamma \vdash^{\mathsf{c}} v : C$$

  Par substitution (lem. 1.4.7) on obtient alors

$$\Gamma \vdash^{\mathsf{c}} w[x \leftarrow v\,] : D[x \leftarrow v\,]$$

  Or comme $B =_\beta D$ alors $B[x \leftarrow v\,] =_\beta D[x \leftarrow v\,]$ (lem. 1.2.9). De plus $\Gamma \vdash^{\mathsf{c}} \forall x^A.B : \kappa'$ (lem. 1.4.9), d'où par génération $\Gamma, x : A \vdash^{\mathsf{c}} B : \kappa''$ (lem. 1.4.8) et par substitution (lem. 1.4.7)

$$\Gamma \vdash^{\mathsf{c}} B[x \leftarrow v\,] : \kappa''$$

  Finalement par (c-conv) on a bien $\Gamma \vdash^{\mathsf{c}} w[x \leftarrow v\,] : B[x \leftarrow v\,]$. □

Théorème 1.4.12 (CC est fortement normalisant).
Si $\Gamma \vdash^{\mathsf{c}} t : C$ alors t et C sont fortement normalisables.

Preuve dans (Barendregt, 1992, sec. 5.3). □

Lemme 1.4.13. *Si $\Gamma \vdash^{\mathsf{c}} C : \mathrm{Type}$ alors $C \equiv \forall \vec{y}^{\vec{D}}.\mathrm{Prop}$.*

Preuve par récurrence structurelle sur la dérivation.

(c-var) Cas impossible (lem. 1.4.2).

(c-app) $$\frac{\Gamma \vdash^{\mathsf{c}} u : \forall x^A.B \quad \Gamma \vdash^{\mathsf{c}} v : A}{\Gamma \vdash^{\mathsf{c}} u\ v : B[x \leftarrow v\,]}$$

On a deux cas (lem. 1.2.2) :

- $B \equiv x$ et $v \equiv \mathrm{Type}$ ce qui est impossible puisque $\Gamma \vdash^{\mathsf{c}} v : A$ (lem. 1.4.2).
- $B \equiv \mathrm{Type}$, et alors $\Gamma \vdash^{\mathsf{c}} \forall x^A.\mathrm{Type} : \kappa$ (lem. 1.4.9) ce qui est impossible (lem. 1.4.2).

(c-prod) Il suffit d'appliquer l'hypothèse de récurrence sur la prémisse.

(c-conv) Cas impossible à cause de la troisième prémisse (lem. 1.4.2). □

Lemme 1.4.14. *Si $\Gamma \vdash^{\mathsf{c}} \forall y_1^{C_1}.\ldots.y_n^{C_n}.\mathrm{Prop} : T$ alors $T \equiv \mathrm{Type}$.*

Preuve par récurrence sur n.

Par génération (lem. 1.4.8) $\Gamma, y_1 : C_1 \vdash^{\mathsf{c}} \forall y_2^{C_2}.\ldots.y_n^{C_n}.\mathrm{Prop} : \kappa$ avec $T =_\beta \kappa$, et alors par hypothèse de récurrence $\kappa \equiv \mathrm{Type}$.

Par Church-Rosser (thm. 1.2.7) et la définition de $\rightsquigarrow_\beta$ on a forcément $T \overset{*}{\rightsquigarrow}_\beta \mathrm{Type}$. On a alors deux cas (lem. 1.4.9) :

- $T \equiv \mathrm{Type}$ et tout va bien ;
- $\Gamma \vdash^{\mathsf{c}} T : \kappa'$ et par réduction (thm. 1.4.11) $\Gamma \vdash^{\mathsf{c}} \mathrm{Type} : \kappa'$ ce qui n'est pas possible. □

Chapitre 2

Pédagogisme dans le calcul des constructions

2.1 Critère de Poincaré et utilité **67**
2.2 Un calcul des constructions Poincaréien – $\mathbf{CC_r}$ **68**
2.3 Définition d'un sous-système pédagogique de CC **78**

2.1 Critère de Poincaré et utilité

Critère de Poincaré

◊ L'environnement $x_1 : A_1, \ldots, x_n : A_n$ satisfait le *critère de Poincaré* dans $\mathrm{CC}_\star$ s'il existe des termes $t_1, \ldots, t_n$ tels que :

$$\vdash^\star t_1 : A_1 \qquad \vdash^\star t_2 : A_2[x_1 \leftarrow t_1] \qquad \ldots \qquad \vdash^\star t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$$

◊ Un sous-système de CC satisfait le critère de Poincaré si tout environnement bien formé de ce système (i.e. $\Gamma\,\mathsf{wf}^\star$) satisfait le critère de Poincaré.

Utilité

◊ Si $x_1 : A_1, \ldots, x_n : A_n \vdash^\star f : \forall x^B.C$, on dit que f est *utile* s'il existe des termes t_i tels que

$$\vdash^\star t_1 : A_1 \qquad \vdash^\star t_2 : A_2[x_1 \leftarrow t_1] \qquad \ldots \qquad \vdash^\star t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$$

et un terme u tel que

$$\vdash^\star \sigma(f) : \forall x^{\sigma(B)}.\sigma(C) \qquad \text{et} \qquad \vdash^\star u : \sigma(B)$$

avec $\sigma := [x_1 \mapsto t_1, \ldots, x_n \mapsto t_n]$.

Remarque

Autrement dit, une fonction f est utile si l'une de ses instances $\sigma(f)$ peut être appliquée à un argument.

2.2 Un calcul des constructions Poincaréien – $\mathrm{CC_r}$

Règles d'inférence de $\mathrm{CC_r}$

◊ On définit la relation $\vdash^{\mathsf{r}}$ entre un environnement et deux termes bruts, ainsi que la relation wf^{r} sur les environnements par les règles suivantes :

$$\frac{}{[\,]\,\mathsf{wf}^{\mathsf{r}}}\ (\text{r-env}_1) \qquad \frac{\Gamma \vdash^{\mathsf{r}} A : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^{\mathsf{r}}}\ (\text{r-env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^{\mathsf{r}}}{\Gamma \vdash^{\mathsf{r}} \mathrm{Prop} : \mathrm{Type}}\ (\text{r-ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^{\mathsf{r}}}{\Gamma, x : A, \Gamma' \vdash^{\mathsf{r}} x : A}\ (\text{r-var})$$

$$\frac{\Gamma, x : A \vdash^{\mathsf{r}} u : B : \kappa}{\Gamma \vdash^{\mathsf{r}} \lambda x^A.u : \forall x^A.B}\ (\text{r-abs}) \qquad \frac{\Gamma \vdash^{\mathsf{r}} u : \forall x^A.B \quad \Gamma \vdash^{\mathsf{r}} v : A}{\Gamma \vdash^{\mathsf{r}} u\ v : B[x \leftarrow v\,]}\ (\text{r-app})$$

$$\frac{\Gamma, x : A \vdash^{\mathsf{r}} u : B : \kappa}{\Gamma \vdash^{\mathsf{r}} \forall x^A.B : \kappa}\ (\text{r-prod}) \qquad \frac{\Gamma \vdash^{\mathsf{r}} u : A \quad A =_\beta A' \quad \Gamma \vdash^{\mathsf{r}} A' : \kappa}{\Gamma \vdash^{\mathsf{r}} u : A'}\ (\text{r-conv})$$

2.2.1 Résultats sur la syntaxe de $\mathrm{CC_r}$

Théorème 2.2.1 ($\mathrm{CC_r}$ est un sous-système de CC).

(i) si $\Gamma\,\mathsf{wf}^{\mathsf{r}}$*, alors* $\Gamma\,\mathsf{wf}^{\mathsf{c}}$ *;*

(ii) si $\Gamma \vdash^{\mathsf{r}} w : C$*, alors* $\Gamma \vdash^{\mathsf{c}} w : C$*.*

Preuve immédiate : $\mathrm{CC_r}$ est plus contraint que CC. □

Les résultats précédents pour CC (sec. 1.4.2) restent valides : ils sont mentionnés ici pour $\mathrm{CC_r}$ pour des raisons de complétude et leur preuve est omise.

Lemme 2.2.2 (voir lem. 1.4.1).

(i) Si $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^{\mathsf{r}}$*, alors pour tout* i*,* $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$ *et* $x_i \not\equiv x_j$ *pour tout* $i \neq j$ *;*

(ii) Si $x_1 : A_1, \ldots, x_n : A_n \vdash^{\mathsf{r}} w : C$*, alors pour tout* i*,* $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$ *et* $x_i \not\equiv x_j$ *pour tout* $i \neq j$ *et* $\mathcal{V}(w, C) \subseteq \{x_1, \ldots, x_n\}$*.*

Lemme 2.2.3 (voir lem. 1.4.2). *Si* $\Gamma\,\mathsf{wf}^{\mathsf{r}}$ *ou* $\Gamma \vdash^{\mathsf{r}} w : C$*, alors* $\mathrm{Type} \notin \mathcal{S}(\Gamma)$ *et* $\mathrm{Type} \notin \mathcal{S}(w)$*.*

Lemme 2.2.4 (validité des environnements, voir lem. 1.4.3).

(i) si $\Gamma\,\mathsf{wf}^{\mathsf{r}}$*, alors pour tout environnement* $\Gamma' \preccurlyeq \Gamma$*,* $\Gamma'\,\mathsf{wf}^{\mathsf{r}}$ *en est une sous-dérivation ;*

(ii) si $\Gamma \vdash^{\mathsf{r}} w : C$*, alors pour tout environnement* $\Gamma' \preccurlyeq \Gamma$*,* $\Gamma'\,\mathsf{wf}^{\mathsf{r}}$ *en est une sous-dérivation stricte.*

Lemme 2.2.5 (validité des types des environnements, voir lem. 1.4.4).

Si $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^{\mathsf{r}}$*, alors pour tout* i *il existe* κ *tel que* $x_1 : A_1, \ldots, x_i : A_i \vdash^{\mathsf{r}} A_{i+1} : \kappa$ *en est une sous-dérivation stricte.*

Lemme 2.2.6 (affaiblissement, voir lem. 1.4.5). *Si* $\Gamma \vdash^{\mathsf{r}} w : C$*,* $\Gamma'\,\mathsf{wf}^{\mathsf{r}}$ *et* $\Gamma \subseteq \Gamma'$*, alors* $\Gamma' \vdash^{\mathsf{r}} w : C$*.*

Lemme 2.2.7 (unicité du type, voir lem. 1.4.6). *Si* $\Gamma \vdash^{\mathsf{r}} w : C$ *et* $\Gamma \vdash^{\mathsf{r}} w : C'$*, alors* $C =_\beta C'$*.*

Lemme 2.2.8 (de substitution, voir lem. 1.4.7).

(i) si $\Gamma, y : C, \Gamma' \,\mathsf{wf}^{\mathsf{r}}$ *et* $\Gamma \vdash^{\mathsf{r}} w : C$*, alors* $\Gamma, \Gamma'[y \leftarrow w]\,\mathsf{wf}^{\mathsf{r}}$ *;*

(ii) si $\Gamma, y : C, \Gamma' \vdash^{\mathsf{r}} t : D$ *et* $\Gamma \vdash^{\mathsf{r}} w : C$*, alors* $\Gamma, \Gamma'[y \leftarrow w] \vdash^{\mathsf{r}} t[y \leftarrow w] : D[y \leftarrow w]$*.*

Lemme 2.2.9 (génération, voir lem. 1.4.8).

Soit $\Gamma \vdash^{\mathsf{r}} t : T$*, un de ces cas se présente :*

(i) si $t \equiv \mathrm{Prop}$*, alors* $T =_\beta \mathrm{Type}$ *;*

(ii) si $t \equiv x$*, alors il existe* $(x : A) \in \Gamma$ *avec* $T =_\beta A$ *;*

(iii) si $t \equiv \lambda x^A.u$*, alors il existe* B *et* κ *tels que* $\Gamma, x : A \vdash^{\mathsf{r}} u : B : \kappa$ *est une sous-dérivation stricte avec* $T =_\beta \forall x^A.B$ *;*

(iv) si $t \equiv u\ v$*, alors il existe* A *et* B *tels que* $\Gamma \vdash^{\mathsf{r}} u : \forall x^A.B$ *et* $\Gamma \vdash^{\mathsf{r}} v : A$ *sont des sous-dérivations strictes avec* $T =_\beta B[x \leftarrow v]$ *;*

(v) si $t \equiv \forall x^A.B$*, alors il existe* u *et* κ *tels que* $\Gamma, x : A \vdash^{\mathsf{r}} u : B : \kappa$ *sont des sous-dérivations strictes avec* $T =_\beta \kappa$*.*

Lemme 2.2.10 (correction des types, voir lem. 1.4.9).

Si $\Gamma \vdash^{\mathsf{r}} w : C$*, alors* $C \equiv \mathrm{Type}$ *ou bien il existe* κ *tel que* $\Gamma \vdash^{\mathsf{r}} C : \kappa$*.*

Théorème 2.2.11 (stabilité par réduction, voir thm. 1.4.11).

Si $\Gamma \vdash^{\mathsf{r}} t : C$ *et* $t \rightsquigarrow_\beta t'$*, alors* $\Gamma \vdash^{\mathsf{r}} t' : C$*.*

Théorème 2.2.12 ($\mathbf{CC_r}$ est fortement normalisant, voir thm. 1.4.12).

Si $\Gamma \vdash^{\mathsf{r}} t : C$ *alors* t *est fortement normalisable.*

Preuve immédiate car $\mathrm{CC_r}$ est un sous-système de CC (thm. 2.2.1). □

Lemme 2.2.13 (voir lem. 1.4.13). *Si* $\Gamma \vdash^{\mathsf{r}} C : \mathrm{Type}$ *alors* $C \equiv \forall \vec{y}^{\vec{D}}.\mathrm{Prop}$*.*

Preuve immédiate car $\mathrm{CC_r}$ est un sous-système de CC (thm. 2.2.1). □

Lemme 2.2.14 (voir lem. 1.4.14). *Si* $\Gamma \vdash^{\mathsf{r}} \forall y_1^{C_1}.\ldots.y_n^{C_n}.\mathrm{Prop} : T$ *alors* $T \equiv \mathrm{Type}$*.*

Preuve immédiate car $\mathrm{CC_r}$ est un sous-système de CC (thm. 2.2.1). □

Lemme 2.2.15. *Dans* CC_r*, on a les règles dérivées suivantes :*

$$\frac{\Gamma\,\mathsf{wf}^{\mathsf{r}}}{\Gamma \vdash^{\mathsf{r}} o : \top : \mathrm{Prop} : \mathrm{Type}}$$

avec $o := \lambda A^{\mathrm{Prop}}.\lambda x^A.x$ *et* $\top := \forall A^{\mathrm{Prop}}.A \to A$*.*

Preuve

1.	$\Gamma\,\mathsf{wf}^{\mathsf{r}}$	(hypothèse)
2.	$\Gamma \vdash^{\mathsf{r}} \mathrm{Prop} : \mathrm{Type}$	(r-ax 1)
3.	$\Gamma, A : \mathrm{Prop}\ \mathsf{wf}^{\mathsf{r}}$	(r-env$_2$ 2)
4.	$\Gamma, A : \mathrm{Prop} \vdash^{\mathsf{r}} A : \mathrm{Prop}$	(r-var 3)
5.	$\Gamma, A : \mathrm{Prop}, x : A\,\mathsf{wf}^{\mathsf{r}}$	(r-env$_2$ 4)
6.	$\Gamma, A : \mathrm{Prop}, x : A \vdash^{\mathsf{r}} x : A : \mathrm{Prop}$	(r-var 5)
7.	$\Gamma, A : \mathrm{Prop} \vdash^{\mathsf{r}} \lambda x^A.x : A \to A$	(r-abs 6)
8.	$\Gamma, A : \mathrm{Prop} \vdash^{\mathsf{r}} A \to A : \mathrm{Prop}$	(r-prod 6 et 7)
9.	$\Gamma \vdash^{\mathsf{r}} \lambda A^{\mathrm{Prop}}.\lambda x^A.x : \forall A^{\mathrm{Prop}}.A \to A$	(r-abs 7 8)
10.	$\Gamma \vdash^{\mathsf{r}} \forall A^{\mathrm{Prop}}.A \to A : \mathrm{Prop}$	(r-prod 7 et 8)

□

Lemme 2.2.16. *Si* $\Gamma \vdash^{\mathsf{r}} C : \mathrm{Type}$ *alors il existe un terme* t *tel que* $\Gamma \vdash^{\mathsf{r}} t : C$.

Preuve

Puisque $C \equiv \forall y_1^{D_1}.\ldots.y_n^{D_n}.\mathrm{Prop}$ (lem. 2.2.13), on a deux cas :

- $n = 0$: comme $\Gamma\ \mathsf{wf}^{\mathsf{c}}$ (lem. 2.2.4), alors $t := \top$ convient (lem. 2.2.15) ;
- $n = m + 1$: par génération (lem. 2.2.9) on a $\Gamma, y_1 : D_1 \vdash^{\mathsf{c}} u : \forall y_2^{D_2}.\ldots.y_n^{D_n} : \kappa$ et donc par (r-abs) $t := \lambda y_1^{D_1}.u$ convient. □

Lemme 2.2.17.

Si $\Gamma \vdash^{\mathsf{c}} C : \forall y_1^{D_1}.\forall y_2^{D_2}.\ldots.\forall y_n^{D_n}.\mathrm{Prop}$ *avec* C *clos, alors pour tous termes clos* $w_1, \ldots, w_n$ *vérifiant*

$$\Gamma \vdash^{\mathsf{c}} w_1 : D_1 \qquad \Gamma \vdash^{\mathsf{c}} w_2 : D_2[y_1 \leftarrow w_1] \qquad \ldots \qquad \Gamma \vdash^{\mathsf{c}} w_n : D_n[y_1, \ldots, y_{n-1} \leftarrow w_1, \ldots, w_{n-1}]$$

il existe des termes bruts E *et* F *tels que*

$$C\ \vec{w} \overset{*}{\rightsquigarrow}_\beta \forall z^E.F$$

Preuve par récurrence sur l'ordre lexicographique de $\|C\ \vec{w}\|$ et de la hauteur de la dérivation de $\Gamma \vdash^{\mathsf{c}} C : \forall \vec{y}^{\vec{D}}.\mathrm{Prop}$. En effet $\Gamma \vdash^{\mathsf{c}} C\ \vec{w} : \mathrm{Prop}$ donc $C\ \vec{w}$ est fortement normalisable (thm. 1.4.12) et $\|C\ \vec{w}\|$ est bien définie.

$$\textbf{(c-abs)}\ \frac{\Gamma, y_1 : D_1 \vdash^{\mathsf{c}} u : \forall y_2^{D_2}.\ldots.\forall y_n^{D_n}.\mathrm{Prop} : \kappa}{\Gamma \vdash^{\mathsf{c}} \lambda y_1^{D_1}.u : \forall y_1^{D_1}.\forall y_2^{D_2}.\ldots.\forall y_n^{D_n}.\mathrm{Prop}}$$

De $\Gamma, y_1 : D_1 \vdash^{\mathsf{c}} u : \forall y_2^{D_2}.\ldots.\forall y_n^{D_n}.\mathrm{Prop} : \kappa$ et $\Gamma \vdash^{\mathsf{c}} w_1 : D_1$, par substitution on a (lem. 1.4.7)

$$\Gamma \vdash^{\mathsf{c}} u[y_1 \leftarrow w_1] : \forall y_2^{D_2[y_1 \leftarrow w_1]}.\ldots.\forall y_n^{D_n[y_1 \leftarrow w_1]}.\mathrm{Prop}$$

avec $u[y_1 \leftarrow w_1]$ clos puisque $\mathcal{V}(u) \subset \{y_1\}$ et w_1 est clos. De plus (lem. 1.2.13 et lem. 1.2.12)

$$\|u[y_1 \leftarrow w_1]\ w_2\ \ldots\ w_n\| < \|(\lambda y_1^{D_1}.u)\ w_1\ w_2\ \ldots\ w_n\|$$

on peut alors appliquer l'hypothèse de récurrence avec les termes

$$\Gamma \vdash^{\mathsf{c}} w_2 : D_2[y_1 \leftarrow w_1] \qquad \ldots \qquad \Gamma \vdash^{\mathsf{c}} w_n : D_n[y_1 \leftarrow w_1][y_2, \ldots, y_{n-1} \leftarrow w_2, \ldots, w_{n-1}]$$

car $D_i[y_1 \leftarrow w_1][y_2, \ldots, y_{i-1} \leftarrow w_2, \ldots, w_{i-1}] \equiv D_i[y_1, \ldots, y_{i-1} \leftarrow w_1, \ldots, w_{i-1}]$ puisque les w_i sont clos, pour en déduire finalement que

$$(\lambda y_1^{D_1}.u)\ w_1\ w_2\ \ldots\ w_n \rightsquigarrow_\beta u[y_1 \leftarrow w_1]\ w_2\ \ldots\ w_n \overset{*}{\rightsquigarrow}_\beta \forall z^E.F$$

$$\textbf{(c-app)}\ \frac{\Gamma \vdash^{\mathsf{c}} u : \forall x^A.B \quad \Gamma \vdash^{\mathsf{c}} v : A}{\Gamma \vdash^{\mathsf{c}} u\ v : B[x \leftarrow v]} \quad \text{avec } B[x \leftarrow v] \equiv \forall y_1^{D_1}.\ldots.\forall y_n^{D_n}.\mathrm{Prop}.$$

On a deux cas (lem. 1.2.2) :

- $B \equiv \forall y_1^{E_1}.\ldots.\forall y_i^{E_i}.x$ et $v \equiv \forall y_{i+1}^{D_{i+1}}.\ldots.\forall y_n^{D_n}.\mathrm{Prop}$ avec $E_k[x \leftarrow v] \equiv D_k$.
  De $\Gamma \vdash^{\mathsf{c}} v : A$ on a alors que $A \equiv \mathrm{Type}$ (lem. 1.4.14). Or par génération (lem. 1.4.8) $\Gamma, x : A \vdash^{\mathsf{c}} B : \kappa$ ce qui n'est pas possible (lem. 1.4.2).

- $B \equiv \forall y_1^{E_1}.\ldots.\forall y_n^{E_n}.\,\mathrm{Prop}$ avec $E_k[x \leftarrow v\,] \equiv D_k$.
  La règle se récrit :
$$\frac{\Gamma \vdash^{\mathsf{c}} u : \forall x^A.\forall y_1^{E_1}.\ldots.\forall y_n^{E_n}.\,\mathrm{Prop} \quad \Gamma \vdash^{\mathsf{c}} v : A}{\Gamma \vdash^{\mathsf{c}} u\ v : \forall y_1^{D_1}.\ldots.\forall y_n^{D_n}.\,\mathrm{Prop}}$$
  On peut appliquer l'hypothèse de récurrence sur la prémisse et les termes
$$\Gamma \vdash^{\mathsf{c}} v : A \qquad \Gamma \vdash^{\mathsf{c}} w_1 : E_1[x \leftarrow v\,] \quad \ldots \quad \Gamma \vdash^{\mathsf{c}} w_n : E_n[x, y_1, \ldots, y_{n-1} \leftarrow v, w_1, \ldots, w_{n-1}\,]$$
  puisque comme v est clos
$$\begin{aligned} E_i[x, y_1, \ldots, y_{i-1} \leftarrow w_1, \ldots, w_{i-1}\,] &\equiv E_i[x \leftarrow v\,][y_1, \ldots, y_{i-1} \leftarrow w_1, \ldots, w_{i-1}\,] \\ &\equiv D_i[y_1, \ldots, y_{i-1} \leftarrow w_1, \ldots, w_{i-1}\,] \end{aligned}$$
  pour obtenir finalement
$$(u\ v)\ w_1\ \ldots\ w_n \overset{*}{\rightsquigarrow}_\beta \forall z^E.F$$

(c-prod) Cas trivial.

(c-conv) $$\frac{\Gamma \vdash^{\mathsf{c}} u : A \quad A =_\beta \forall y_1^{D_1}.\ldots.\forall y_n^{D_n}.\,\mathrm{Prop} \quad \Gamma \vdash^{\mathsf{c}} \forall y_1^{D_1}.\ldots.\forall y_n^{D_n}.\,\mathrm{Prop} : \kappa}{\Gamma \vdash^{\mathsf{c}} u : \forall y_1^{D_1}.\ldots.\forall y_n^{D_n}.\,\mathrm{Prop}}$$

Par Church-Rosser (thm. 1.2.7) et la définition de $\rightsquigarrow_\beta$, on a $A \overset{*}{\rightsquigarrow}_\beta \forall \vec{y}^{\vec{E}}.\,\mathrm{Prop} \overset{*}{\leftsquigarrow}_\beta \forall \vec{y}^{\vec{D}}.\,\mathrm{Prop}$. On a alors trois cas possibles (lem. 1.4.9) :

- $A \equiv$ Type n'arrive jamais par définition de $\rightsquigarrow_\beta$ car $A \overset{*}{\rightsquigarrow}_\beta \forall \vec{y}^{\vec{E}}.\,\mathrm{Prop}$;
- $\Gamma \vdash^{\mathsf{c}} A : \mathrm{Prop}$ et donc par réduction (lem. 1.4.11) $\Gamma \vdash^{\mathsf{c}} \forall \vec{y}^{\vec{E}}.\,\mathrm{Prop} : \mathrm{Prop}$ ce qui est impossible (lem. 1.4.14) ;
- $\Gamma \vdash^{\mathsf{c}} A :$ Type d'où $A \equiv \forall y_1^{F_1}.\ldots.\forall y_m^{F_m}.\,\mathrm{Prop}$ (lem. 1.4.13) et par définition de $\rightsquigarrow_\beta$, nécessairement $m = n$ et $F_i =_\beta D_i$.

Pour pouvoir appliquer l'hypothèse de récurrence sur la première prémisse, il faut que
$$\Gamma \vdash^{\mathsf{c}} w_1 : F_1 \qquad \Gamma \vdash^{\mathsf{c}} w_2 : F_2[y_1 \leftarrow w_1\,] \qquad \ldots \qquad \Gamma \vdash^{\mathsf{c}} w_n : F_n[y_1, \ldots, y_{n-1} \leftarrow w_1, \ldots, w_{n-1}\,]$$
ce qu'on prouve par récurrence forte sur n :

Pour tout i (lem. 1.2.8), comme $D_i =_\beta F_i$ alors
$$D_i[y_1, \ldots, y_{i-1} \leftarrow w_1, \ldots, w_{i-1}\,] =_\beta F_i[y_1, \ldots, y_{i-1} \leftarrow w_1, \ldots, w_{i-1}\,]$$
D'autre part, par génération (lem. 1.4.8) sur $\Gamma \vdash^{\mathsf{c}} \forall y_1^{F_1}.\ldots.\forall y_n^{F_n}.\,\mathrm{Prop}$ on a $\Gamma, y_1 : F_1, \ldots, y_n : F_n \vdash^{\mathsf{c}} \mathrm{Prop} : \kappa$ et donc $\Gamma, y_1 : F_1, \ldots, y_{i-1} : F_{i-1} \vdash^{\mathsf{c}} F_i : \kappa_i$ (lem. 1.4.3 et lem. 1.4.4).

Par hypothèse de récurrence sur les $(w_k)_{k<i}$ et après substitutions (lem. 1.4.7) on a
$$\Gamma \vdash^{\mathsf{c}} F_i[y_1, \ldots, y_{i-1} \leftarrow w_1, \ldots, w_{i-1}\,] : \kappa_i$$
puis par (c-conv) on obtient bien
$$\Gamma \vdash^{\mathsf{c}} w_i : F_i[y_1, \ldots, y_{i-1} \leftarrow w_1, \ldots, w_{i-1}\,]$$

Les w_i étant bien typés, on peut appliquer l'hypothèse de récurrence sur la première prémisse pour finalement conclure que $u\ w_1\ \ldots\ w_n$ se réduit bien en un produit. □

Lemme 2.2.18. *Si* $\Gamma \vdash^{\mathsf{r}} C : \mathrm{Prop}$ *avec* C *clos, alors il existe un terme* t *tel que* $\Gamma \vdash^{\mathsf{r}} t : C$.

Preuve

Comme $\Gamma \vdash^{\mathsf{r}} C : \mathrm{Prop}$ alors $\Gamma \vdash^{\mathsf{c}} C : \mathrm{Prop}$ (thm. 2.2.1), et puisque C est clos alors il se réduit en un produit $\forall z^E.F$ (lem. 2.2.17) d'où $\Gamma \vdash^{\mathsf{r}} \forall z^E.F : \mathrm{Prop}$ (thm. 2.2.11).

Par génération on a un terme u tel que $\Gamma, z : E \vdash^{\mathsf{c}} u : F : \kappa$ puis par (r-abs) $\Gamma \vdash^{\mathsf{c}} \lambda z^E.u : \forall z^E.F$. Finalement par (r-conv) on a bien $\Gamma \vdash^{\mathsf{r}} t : C$ avec $t := \lambda z^E.u$. □

Corollaire 2.2.19. *Si* $\Gamma \vdash^{\mathsf{r}} C : \kappa$ *avec* C *clos, alors il existe un terme* t *tel que* $\Gamma \vdash^{\mathsf{r}} t : C$.

Preuve immédiate par cas sur κ (lem. 2.2.16 et lem. 2.2.18). □

Proposition 2.2.20 ($\mathbf{CC_r}$ vérifie le critère de Poincaré).
Si $x_1 : A_1, \ldots, x_n : A_n \, \mathsf{wf}^{\mathsf{r}}$ *alors il existe des termes* $t_1, \ldots, t_n$ *tels que*

$$\vdash^{\mathsf{r}} t_1 : A_1 \qquad \vdash^{\mathsf{r}} t_2 : A_2[x_1 \leftarrow t_1] \qquad \ldots \qquad \vdash^{\mathsf{r}} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$$

Preuve par récurrence sur la taille de l'environnement n.

De $x_1 : A_1, \ldots, x_{i+1} : A_{i+1} \, \mathsf{wf}^{\mathsf{r}}$, on a $x_1 : A_1, \ldots, x_i : A_i \, \mathsf{wf}^{\mathsf{r}}$ (lem. 2.2.4) et par hypothèse de récurrence il existe des termes

$$\vdash^{\mathsf{r}} t_1 : A_1 \qquad \vdash^{\mathsf{r}} t_2 : A_2[x_1 \leftarrow t_1] \qquad \ldots \qquad \vdash^{\mathsf{r}} t_i : A_i[x_1, \ldots, x_{i-1} \leftarrow t_1, \ldots, t_{i-1}]$$

Or $x_1 : A_1, \ldots, x_i : A_i \vdash^{\mathsf{r}} A_{i+1} : \kappa$ (lem. 2.2.5) et donc par substitutions (lem. 2.2.8)

$$\vdash^{\mathsf{r}} A_{i+1}[x_1, \ldots, x_i \leftarrow w_1, \ldots, w_i] : \kappa$$

avec $A_{i+1}[x_1, \ldots, x_i \leftarrow w_1, \ldots, w_i]$ clos (lem. 2.2.2), d'où on a un terme t_{i+1} tel que (cor. 2.2.19)

$$\vdash^{\mathsf{r}} t_{i+1} : A_{i+1}[x_1, \ldots, x_i \leftarrow w_1, \ldots, w_i]$$

Théorème 2.2.21 (utilité des fonctions de $\mathbf{CC_r}$). *Si* $\Gamma \vdash^{\mathsf{r}} f : \forall x^A.B$ *alors* f *est utile.*

Preuve

De $\Gamma \vdash^{\mathsf{r}} f : \forall x^A.B$ on a $\Gamma \vdash^{\mathsf{r}} \forall x^A.B : \kappa$ (lem. 2.2.10) puis par génération (lem. 2.2.9) $\Gamma, x : A \vdash^{\mathsf{r}} B : \kappa'$ et donc $\Gamma, x : A \, \mathsf{wf}^{\mathsf{2}}$ (lem. 2.2.4).

En posant $\Gamma \equiv y_1 : C_1, \ldots, y_n : C_n$, le critère de Poincaré (prop. 2.2.20) nous donne alors des termes $t_1, \ldots, t_n$ et u tels que

$$\vdash^{\mathsf{c}} t_{i+1} : C_{i+1}[y_1, \ldots, y_i \leftarrow t_1, \ldots, t_i] \qquad \text{et} \qquad \vdash^{\mathsf{c}} u : A[y_1, \ldots, y_n \leftarrow t_1, \ldots, t_n]$$

c'est-à-dire avec $\sigma := [y_1 \mapsto t_1, \ldots, y_n \mapsto t_n]$ et n substitutions (lem. 2.2.8)

$$\vdash^{\mathsf{r}} \sigma(f) : \forall x^{\sigma(A)}.\sigma(B) \qquad \text{et} \qquad \vdash^{\mathsf{r}} u : \sigma(A)$$

2.2.2 Fonctions typables de $\mathbf{CC_r}$

Termes du système T

◊ On définit les notations suivantes pour les *termes du système T* :

$$\begin{aligned}
\mathbb{N} &:= \forall A^{\mathrm{Prop}}.A \to (A \to A) \to A \\
0 &:= \lambda A^{\mathrm{Prop}}.\lambda x^A.\lambda f^{A\to A}.x \\
S(n) &:= \lambda A^{\mathrm{Prop}}.\lambda x^A.\lambda f^{A\to A}.f\ (n\ A\ x\ f) \\
\mathrm{it}_T(n, b, (y^T)step) &:= n\ T\ b\ (\lambda y^T.step)
\end{aligned}$$

Lemme 2.2.22. *Dans CC_r on a les règles dérivées/admissibles suivantes :*

$$\frac{\Gamma\,\mathsf{wf}^{\mathsf{r}}}{\Gamma \vdash^{\mathsf{r}} 0 : \mathbb{N} : \mathrm{Prop}} \qquad \frac{\Gamma \vdash^{\mathsf{r}} n : \mathbb{N}}{\Gamma \vdash^{\mathsf{r}} S(n) : \mathbb{N}}$$

$$\frac{\Gamma \vdash^{\mathsf{r}} T : \mathrm{Prop} \quad \Gamma \vdash^{\mathsf{r}} n : \mathbb{N} \quad \Gamma \vdash^{\mathsf{r}} b : T \quad \Gamma, y : T \vdash^{\mathsf{r}} step : T}{\Gamma \vdash^{\mathsf{r}} \mathrm{it}_T(n, b, (y^T)step) : T}$$

Preuve immédiate. □

Lemme 2.2.23. *On a les réductions suivantes :*

$$\begin{aligned}
\mathrm{it}_T(0, b, (y^T)step) &\overset{*}{\rightsquigarrow}_\beta b \\
\mathrm{it}_T(S(n), b, (y^T)step) &\overset{*}{\rightsquigarrow}_\beta step[y \leftarrow it_T(n, b, (y^T)step)]
\end{aligned}$$

Preuve immédiate. □

Types simples sur un type

◊ Les *types simples sur un terme brut T* sont définis par :
- T est un type simple sur T ;
- si A et B sont des types simples sur T alors $A \to B$ est un type simple sur T.

Lemme 2.2.24.

Si $\Gamma\,\mathsf{wf}^{\mathsf{r}}$ et T est un type simple sur $\mathbb{N}$, alors il existe un terme t tel que $\Gamma \vdash^{\mathsf{r}} t : T : \mathrm{Prop}$.

Preuve par récurrence structurelle sur T en tant que type simple sur $\mathbb{N}$.

- $T \equiv \mathbb{N}$: alors $t := 0$ convient ;
- $T \equiv A \to B$ avec A et B des types simples sur $\mathbb{N}$:
  Par hypothèse de récurrence sur A, on a $\Gamma \vdash^{\mathsf{r}} A : \mathrm{Prop}$ et par (r-env$_2$) $\Gamma, x : A\,\mathsf{wf}^{\mathsf{r}}$. Par hypothèse de récurrence sur B, on a $\Gamma \vdash^{\mathsf{r}} b : B : \mathrm{Prop}$, et par affaiblissement (lem. 1.4.5) $\Gamma, x : A \vdash^{\mathsf{r}} b : B : \mathrm{Prop}$.
  Finalement par (r-abs) et (r-prod) on a bien $\Gamma \vdash^{\mathsf{r}} \lambda x^A.b : A \to B : \mathrm{Prop}$. □

Lemme 2.2.25.

Si $\Gamma\,\mathsf{wf}^{\mathsf{r}}$ *et* T *est un type simple sur* $\mathbb{N}$ *alors il existe deux termes* enc_T *et* dec_T *tels que*

$$\Gamma \vdash^{\mathsf{r}} \mathrm{enc}_T : \mathbb{N} \to T \qquad \textit{et} \qquad \Gamma \vdash^{\mathsf{r}} \mathrm{dec}_T : T \to \mathbb{N}$$

et pour tout terme n *on a*

$$\mathrm{dec}_T(\mathrm{enc}_T\ n) \overset{*}{\rightsquigarrow}_\beta n$$

Preuve par récurrence structurelle sur T en tant que type simple sur $\mathbb{N}$.

- $T \equiv \mathbb{N}$: il suffit de prendre l'identité $\lambda x^{\mathbb{N}}.x$ pour enc_T et dec_T.
- $T \equiv A \to B$ avec A et B des types simples sur $\mathbb{N}$:

$$\begin{array}{l} \mathrm{enc}_{A\to B} := \lambda x^{\mathbb{N}}.\lambda z^A.\,\mathrm{enc}_B\ x \\ \mathrm{dec}_{A\to B} := \lambda f^{A\to B}.\,\mathrm{dec}_B\ (f\ a) \end{array}$$

où a est un terme de type A obtenu par le lemme 2.2.24.
En effet :

1.	$\Gamma\,\mathsf{wf}^{\mathsf{r}}$	(hypothèse)
2.	$\Gamma \vdash^{\mathsf{r}} \mathbb{N} : \mathrm{Prop}$	(lem. 2.2.22 1)
3.	$\Gamma, x : \mathbb{N}\,\mathsf{wf}^{\mathsf{r}}$	(r-env$_2$ 2)
4.	$\Gamma, x : \mathbb{N} \vdash^{\mathsf{r}} A : \mathrm{Prop}$	(lem. 2.2.24 3)
5.	$\Gamma, x : \mathbb{N}, z : A\,\mathsf{wf}^{\mathsf{r}}$	(r-env$_2$ 4)
6.	$\Gamma, x : \mathbb{N}, z : A \vdash^{\mathsf{r}} \mathrm{enc}_B : \mathbb{N} \to B$	(H.R.)
7.	$\Gamma, x : \mathbb{N}, z : A \vdash^{\mathsf{r}} x : \mathbb{N}$	(r-var 5)
8.	$\Gamma, x : \mathbb{N}, z : A \vdash^{\mathsf{r}} \mathrm{enc}_B\ x : B$	(r-app 6 7)
9.	$\Gamma, x : \mathbb{N}, z : A \vdash^{\mathsf{r}} B : \mathrm{Prop}$	(lem. 2.2.24 5)
10.	$\Gamma, x : \mathbb{N} \vdash^{\mathsf{r}} \lambda z^A.\,\mathrm{enc}_B\ x : A \to B : \mathrm{Prop}$	(r-abs+r-prod 8 9)
11.	$\Gamma \vdash^{\mathsf{r}} \lambda x^{\mathbb{N}}.\lambda z^A.\,\mathrm{enc}_B\ x : \mathbb{N} \to A \to B$	(r-abs 10)

1.	$\Gamma\,\mathsf{wf}^{\mathsf{r}}$	(hypothèse)
2.	$\Gamma \vdash^{\mathsf{r}} A \to B : \mathrm{Prop}$	(lem. 2.2.24 1)
3.	$\Gamma, f : A \to B\,\mathsf{wf}^{\mathsf{r}}$	(r-env$_2$ 2)
4.	$\Gamma, f : A \to B \vdash^{\mathsf{r}} \mathrm{dec}_B : B \to \mathbb{N}$	(H.R. 3)
5.	$\Gamma, f : A \to B \vdash^{\mathsf{r}} f : A \to B$	(r-var 3)
6.	$\Gamma, f : A \to B \vdash^{\mathsf{r}} a : A$	(lem. 2.2.24 3)
7.	$\Gamma, f : A \to B \vdash^{\mathsf{r}} f\ a : B$	(r-app 5 6)
8.	$\Gamma, f : A \to B \vdash^{\mathsf{r}} \mathrm{dec}_B\ (f\ a) : \mathbb{N}$	(r-app 4 7)
9.	$\Gamma, f : A \to B \vdash^{\mathsf{r}} \mathbb{N} : \mathrm{Prop}$	(lem. 2.2.22 3)
10.	$\Gamma \vdash^{\mathsf{r}} \lambda f^{A\to B}.\,\mathrm{dec}_B\ (f\ a) : (A \to B) \to \mathbb{N}$	(r-abs 8 9)

$$\begin{array}{lll} \mathrm{dec}_{A\to B}(\mathrm{enc}_{A\to B}\ n) & \rightsquigarrow_\beta \mathrm{dec}_B(\mathrm{enc}_{A\to B}\ n\ a) & \\ & \rightsquigarrow_\beta \mathrm{dec}_B((\lambda z^A.\,\mathrm{enc}_B\ n)\ a) & \\ & \rightsquigarrow_\beta \mathrm{dec}_B(\mathrm{enc}_B\ n) & \\ & \overset{*}{\rightsquigarrow}_\beta n & \text{(H.R.)} \end{array}$$

□

Définition

◊ On définit les notations suivantes pour les *couples* :

$$\begin{aligned}
\mathbb{N} \times T &:= (T \to T \to T) \to T \\
\langle n, t\rangle^T &:= \lambda f^{T\to T\to T}.f\ (\mathrm{enc}_T\ \ n)\ t \\
\pi_1(c) &:= \mathrm{dec}_T\ (c\ (\lambda x^T.\lambda y^T.x)) \\
\pi_2(c) &:= c\ (\lambda x^T.\lambda y^T.y)
\end{aligned}$$

Lemme 2.2.26. *Dans CC_r on a les règles dérivées/admissibles suivantes :*

$$\frac{\Gamma\ \mathsf{wf}^{\mathsf{r}}}{\Gamma \vdash^{\mathsf{r}} \mathbb{N} \times T : \mathrm{Prop}} \qquad \frac{\Gamma \vdash^{\mathsf{r}} n : \mathbb{N} \quad \Gamma \vdash^{\mathsf{r}} t : T}{\Gamma \vdash^{\mathsf{r}} \langle n, t\rangle^T : \mathbb{N} \times T}$$

$$\frac{\Gamma \vdash^{\mathsf{r}} c : \mathbb{N} \times T}{\Gamma \vdash^{\mathsf{r}} \pi_1(c) : \mathbb{N}} \qquad \frac{\Gamma \vdash^{\mathsf{r}} c : \mathbb{N} \times T}{\Gamma \vdash^{\mathsf{r}} \pi_2(c) : T}$$

avec T un type simple sur $\mathbb{N}$.

Preuve immédiate (la première règle utilise le lem. 2.2.24). □

Lemme 2.2.27. *On a les réductions suivantes :*

$$\begin{aligned}
\pi_1(\langle n, t\rangle^T) &\overset{*}{\rightsquigarrow}_\beta n \\
\pi_2(\langle n, t\rangle^T) &\overset{*}{\rightsquigarrow}_\beta t
\end{aligned}$$

Preuve

$$\begin{aligned}
\pi_1(\langle n, t\rangle^T) &\rightsquigarrow_\beta \mathrm{dec}_T((\lambda x^T.\lambda y^T.x)\ (\mathrm{enc}_T\ \ n)\ t) \\
&\rightsquigarrow_\beta \mathrm{dec}_T((\lambda y^T.\,\mathrm{enc}_T\ \ n)\ t) \\
&\rightsquigarrow_\beta \mathrm{dec}_T(\mathrm{enc}_T\ \ n) \\
&\overset{*}{\rightsquigarrow}_\beta n && \text{(lem. 2.2.25)}
\end{aligned}$$

□

Récursion primitive

◊ On définit la notation suivante pour l'*opérateur de récursion primitive* rec :

$$\mathrm{rec}_T(n, b, (x^{\mathbb{N}}, y^T)step) := \pi_2\left[\mathrm{it}_{\mathbb{N}\times T}(n, \langle 0, b\rangle^T, (z^{\mathbb{N}\times T})step')\right]$$

où

$$step' := \langle S(\pi_1(z)), (\lambda x^{\mathbb{N}}.\lambda y^T.step)\ \pi_1(z)\ \pi_2(z)\rangle^T$$

Lemme 2.2.28. *Dans CC_r on a la règle admissible suivante :*

$$\frac{\Gamma \vdash^{\mathsf{r}} T : \mathrm{Prop} \quad \Gamma \vdash^{\mathsf{r}} n : \mathbb{N} \quad \Gamma \vdash^{\mathsf{r}} b : T \quad \Gamma, x : \mathbb{N}, y : T \vdash^{\mathsf{r}} step : T}{\Gamma \vdash^{\mathsf{r}} \mathrm{rec}_T(n, b, (x^{\mathbb{N}}, y^T)step) : T}$$

avec T un type simple sur $\mathbb{N}$.

Preuve immédiate (utilise les lem. 2.2.22, 2.2.24, 2.2.26, ainsi que lem. 2.2.4, 2.2.6). □

Lemme 2.2.29. *On a les réductions suivantes :*

$$\begin{aligned} \mathrm{rec}_T(0, b, (x^{\mathbb{N}}, y^T)step) &\overset{*}{\rightsquigarrow}_\beta b \\ \mathrm{rec}_T(S^{n+1}(0), b, (x^{\mathbb{N}}, y^T)step) &\overset{*}{\rightsquigarrow}_\beta step[x, y \leftarrow S^n(0), rec_T(S^n(0), b, (x^{\mathbb{N}}, y^T)step)\,] \end{aligned}$$

où $S^{n+1}(0) := \underbrace{S(S(\ldots(S}_{n+1\ fois}\ 0)))$.

Preuve En posant $v_n := \mathrm{it}_{\mathbb{N}\times T}(S^n(0), \langle 0, b\rangle^T, (z^{\mathbb{N}\times T})step')$, avec $step'$ comme définit précédemment, on montre par récurrence sur n que :

$$\begin{aligned} v_0 &\overset{*}{\rightsquigarrow}_\beta \langle 0, b\rangle^T \\ v_{n+1} &\overset{*}{\rightsquigarrow}_\beta \langle S^{n+1}(0), step[x, y \leftarrow S^n(0), \pi_2(v_n)\,]\rangle \end{aligned}$$

- $n = 0$ et $n = 1$: cas immédiats (lem. 2.2.23).
- $n = m + 2$:

$$\begin{aligned} v_{m+2} &\overset{*}{\rightsquigarrow}_\beta step'[z \leftarrow v_{m+1}\,] && \text{(lem. 2.2.23)} \\ &:= \langle S(\pi_1(v_{m+1})), (\lambda x^{\mathbb{N}}.\lambda y^T.step)\ \pi_1(v_{m+1})\ \pi_2(v_{m+1})\rangle^T \\ &\overset{*}{\rightsquigarrow}_\beta \langle S(S^{m+1}(0)), step[x, y \leftarrow S^{m+1}(0), \pi_2(v_{m+1})\,]\rangle^T && \text{(H.R. et lem. 2.2.27)} \end{aligned}$$

Or $\mathrm{rec}_T(S^n(0), b, (x^{\mathbb{N}}, y^T)step) := \pi_2(v_n)$, d'où

$$\begin{aligned} \pi_2(v_0) &\overset{*}{\rightsquigarrow}_\beta \pi_2(\langle 0, b\rangle^T) \\ &\overset{*}{\rightsquigarrow}_\beta b && \text{(lem. 2.2.27)} \\ \pi_2(v_{n+1}) &\overset{*}{\rightsquigarrow}_\beta \pi_2(\langle S^{n+1}(0), step[x, y \leftarrow S^n(0), \pi_2(v_n)\,]\rangle^T) \\ &\overset{*}{\rightsquigarrow}_\beta step[x, y \leftarrow S^n(0), \pi_2(v_n)\,] && \text{(lem. 2.2.27)} \end{aligned}$$

□

Remarque

On n'a pas en général $\mathrm{rec}_T(S(n), b, (x^{\mathbb{N}}, y^T)step) \overset{*}{\rightsquigarrow}_\beta step[x, y \leftarrow n, rec_T(n, b, (x^{\mathbb{N}}, y^T)step)\,]$ car le premier argument $S(n)$ doit être complètement décomposé par rec jusqu'à l'obtention d'un 0. Ce fait est bien connu et survient aussi dans le système F (voir (Girard *et al.*, 1990, ch. 11)).

2.2.3 Expressivité logique de $\mathrm{CC_r}$

Théorème 2.2.30. *Il n'existe aucun terme u tel que*

$$A : \mathrm{Prop}, B : \mathrm{Prop}, C : \mathrm{Prop} \vdash^{\mathsf{r}} u : (A \to B) \to (B \to C) \to (A \to C)$$

Preuve par contradiction : supposons qu'un tel terme u existe.

Comme $A : \mathrm{Prop}, B : \mathrm{Prop}, C : \mathrm{Prop} \vdash^{\mathsf{r}} (A \to B) \to (B \to C) \to (A \to C) : \kappa$ (lem. 2.2.10), alors par génération (lem. 2.2.9)

$$A : \mathrm{Prop}, B : \mathrm{Prop}, C : \mathrm{Prop}, f : (A \to B) \vdash^{\mathsf{r}} (B \to C) \to (A \to C) : \kappa'$$

d'où (lem. 2.2.4 et lem. 2.2.5)

$$A : \mathrm{Prop}, B : \mathrm{Prop}, C : \mathrm{Prop} \vdash^{\mathsf{r}} (A \to B) : \kappa''$$

et puisque $A \to B$ est un produit, par génération (lem. 2.2.9) et (r-abs) il est habité par un terme t, donc

$$A : \mathrm{Prop}, B : \mathrm{Prop}, C : \mathrm{Prop} \vdash^{\mathsf{r}} t : A \to B$$

Or $\mathrm{CC_r}$ étant un sous-système de CC (thm. 2.2.1), on a aussi

$$A : \mathrm{Prop}, B : \mathrm{Prop}, C : \mathrm{Prop} \vdash^{\mathsf{c}} t : A \to B$$

et donc pas substitution (lem. 1.4.7) sur $\top := \forall A^{\mathrm{Prop}}.A \to A$ et $\bot := \forall A^{\mathrm{Prop}}.A$ et (r-app) sur $o := \lambda A^{\mathrm{Prop}}.\lambda x^A.x$ on a

$$\vdash^{\mathsf{c}} t\ o : \bot$$

mais il n'existe pas de preuve en forme normale de $\bot$ dans CC. □

Remarque

$\mathsf{CC_r}$ ne contient pas nativement le λ-calcul simplement typé.

Théorème 2.2.31. *Il n'existe pas de terme u tel que $\vdash^{\mathsf{r}} u : \forall A^{\mathrm{Prop}}.\forall x^A.\forall y^A.x =_A y \to y =_A x$ avec*

$$x =_A y := \forall Q^{A \to \mathrm{Prop}}.Q\ x \to Q\ y$$

Preuve par contradiction : supposons qu'un tel terme u existe.

De même que précédemment (lem. 2.2.10, 2.2.9, 2.2.4, 2.2.5) on doit avoir

$$A : \mathrm{Prop}, x : A, y : A \vdash^{\mathsf{r}} x =_A y : \kappa$$

et donc un terme t (lem. 2.2.9) tel que

$$A : \mathrm{Prop}, x : A, y : A \vdash^{\mathsf{r}} t : x =_A y$$

d'où aussi (thm. 2.2.1)

$$A : \mathrm{Prop}, x : A, y : A \vdash^{\mathsf{c}} t : x =_A y$$

et en instanciant sur $\mathbb{N}$, 0 et $S(0)$ (lem. 1.4.7)

$$\vdash^{\mathsf{c}} t' : 0 =_{\mathbb{N}} S(0)$$

ce qui est impossible dans CC. □

2.3 Définition d'un sous-système pédagogique de CC

Sous-système pédagogique de CC

◊ Un système (ici $\star$) est un *sous-système pédagogique* de CC si :

(i) C'est un sous-système de CC : si $\Gamma\,\mathsf{wf}^\star$ alors $\Gamma\,\mathsf{wf}^\mathsf{c}$; et si $\Gamma \vdash^\star w : C$ alors $\Gamma \vdash^\mathsf{c} w : C$.

(ii) Si $\Gamma \vdash^\star t : C$ et $t \rightsquigarrow_\beta t'$, alors $\Gamma \vdash^\star t' : C$.

(iii) $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^\star$ **si et seulement si** $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^\mathsf{c}$ et il existe des termes $t_1, \ldots, t_n$ tels que

$$\vdash^\star t_1 : A_1 \quad \vdash^\star t_2 : A_2[x_1 \leftarrow t_1] \quad \ldots \quad \vdash^\star t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$$

Chapitre 3

Sous-systèmes pédagogiques du second-ordre

3.1	**Lambda-calcul polymorphe – λ^2**	**79**
3.2	**Avec motivations totales explicites – λ^2_e**	**83**
3.3	**Avec motivations totales – λ^2_t**	**97**
3.4	**Avec motivations partielles – λ^2_p**	**102**
3.5	**Système F pédagogique – P-Prop^2**	**106**
3.6	**Indécidabilité de la vérification du typage**	**115**

3.1 Lambda-calcul polymorphe – λ^2

3.1.1 Définition du système

Règles d'inférence de λ^2

◊ On définit la relation $\vdash^2$ entre un environnement et deux termes bruts, ainsi que la relation wf^2 sur les environnements par les règles suivantes :

$$\frac{}{[\,]\,\mathsf{wf}^2}\,(\mathrm{env}_1) \qquad \frac{\Gamma \vdash^2 A : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^2}\,(\mathrm{env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^2}{\Gamma \vdash^2 \mathrm{Prop} : \mathrm{Type}}\,(\mathrm{ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^2}{\Gamma, x : A, \Gamma' \vdash^2 x : A}\,(\mathrm{var})$$

$$\frac{\Gamma, x : A \vdash^2 u : B : \mathrm{Prop}}{\Gamma \vdash^2 \lambda x^A.u : \forall x^A.B}\,(\mathrm{abs}) \qquad \frac{\Gamma \vdash^2 u : \forall x^A.B \quad \Gamma \vdash^2 v : A}{\Gamma \vdash^2 u\ v : B[x \leftarrow v\,]}\,(\mathrm{app})$$

$$\frac{\Gamma, x : A \vdash^2 B : \mathrm{Prop}}{\Gamma \vdash^2 \forall x^A.B : \mathrm{Prop}}\,(\mathrm{prod})$$

Remarques

λ^2 est constitué d'un sous-ensemble des règles de CC, ce qui en fait un sous-système (lem. 3.1.1).

La règle de conversion des types (c-conv) n'est pas nécessaire car les types sont tous en forme normale (cor. 3.1.14).

3.1.2 Résultats

Théorème 3.1.1 (λ^2 est un sous-système de CC).

(i) si $\Gamma\,\mathsf{wf}^2$*, alors* $\Gamma\,\mathsf{wf}^{\mathsf{c}}$ *;*

(ii) si $\Gamma \vdash^2 w : C$*, alors* $\Gamma \vdash^{\mathsf{c}} w : C$*.*

Preuve immédiate : λ^2 ne contient pas toutes les règles de CC. □

Les résultats précédents pour CC (sec. 1.4.2) restent valides : ils sont mentionnés ici pour λ^2 pour des raisons de complétude et leur preuve est omise.

Lemme 3.1.2 (voir lem. 1.4.1).

(i) Si $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^2$*, alors pour tout* i*,* $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$ *et* $x_i \not\equiv x_j$ *pour tout* $i \neq j$ *;*

(ii) Si $x_1 : A_1, \ldots, x_n : A_n \vdash^2 w : C$*, alors pour tout* i*,* $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$ *et* $x_i \not\equiv x_j$ *pour tout* $i \neq j$ *et* $\mathcal{V}(w, C) \subseteq \{x_1, \ldots, x_n\}$*.*

Lemme 3.1.3 (voir lem. 1.4.2). *Si* $\Gamma\,\mathsf{wf}^2$ *ou* $\Gamma \vdash^2 w : C$*, alors* $\mathrm{Type} \notin \mathcal{S}(\Gamma)$ *et* $\mathrm{Type} \notin \mathcal{S}(w)$*.*

Lemme 3.1.4 (validité des environnements, voir lem. 1.4.3).

(i) si $\Gamma\,\mathsf{wf}^2$*, alors pour tout environnement* $\Gamma' \preccurlyeq \Gamma$*,* $\Gamma'\,\mathsf{wf}^2$ *en est une sous-dérivation ;*

(ii) si $\Gamma \vdash^2 w : C$*, alors pour tout environnement* $\Gamma' \preccurlyeq \Gamma$*,* $\Gamma'\,\mathsf{wf}^2$ *en est une sous-dérivation stricte.*

Lemme 3.1.5 (validité des types des environnements, voir lem. 1.4.4).

Si $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^2$*, alors pour tout* i *il existe* κ *tel que* $x_1 : A_1, \ldots, x_i : A_i \vdash^2 A_{i+1} : \kappa$ *en est une sous-dérivation stricte.*

Lemme 3.1.6 (affaiblissement, voir lem. 1.4.5).

Si $\Gamma \vdash^2 w : C$*,* $\Gamma'\,\mathsf{wf}^2$ *et* $\Gamma \subseteq \Gamma'$*, alors* $\Gamma' \vdash^2 w : C$*.*

Lemme 3.1.7 (unicité du type, voir lem. 1.4.6). *Si* $\Gamma \vdash^2 w : C$ *et* $\Gamma \vdash^2 w : C'$*, alors* $C \equiv C'$*.*

Remarque Ici les types sont syntaxiquement égaux, et pas seulement β-convertibles comme dans le cas de CC. En effet, les types de λ^2 sont toujours en forme normale (cor. 3.1.14).

Lemme 3.1.8 (de substitution, voir lem. 1.4.7).

(i) Si $\Gamma, y : C, \Gamma'\,\mathsf{wf}^2$ *et* $\Gamma \vdash^2 w : C$*, alors* $\Gamma, \Gamma'[y \leftarrow w]\,\mathsf{wf}^2$ *;*

(ii) Si $\Gamma, y : C, \Gamma' \vdash^2 t : D$ *et* $\Gamma \vdash^2 w : C$*, alors* $\Gamma, \Gamma'[y \leftarrow w] \vdash^2 t[y \leftarrow w] : D[y \leftarrow w]$*.*

Lemme 3.1.9 (génération, voir lem. 1.4.8). *Soit* $\Gamma \vdash^2 t : T$*, un de ces cas se présente :*

(i) si $t \equiv \mathrm{Prop}$*, alors* $T \equiv \mathrm{Type}$ *;*

(ii) si $t \equiv x$*, alors il existe* $(x : A) \in \Gamma$ *avec* $T \equiv A$ *;*

(iii) si $t \equiv \lambda x^A.u$*, alors il existe* B *tels que* $\Gamma, x : A \vdash^2 u : B : \mathrm{Prop}$ *est une sous-dérivation stricte avec* $T \equiv \forall x^A.B$ *;*

(iv) si $t \equiv u\ v$*, alors il existe* A *et* B *tels que* $\Gamma \vdash^2 u : \forall x^A.B$ *et* $\Gamma \vdash^2 v : A$ *sont des sous-dérivations strictes avec* $T \equiv B[x \leftarrow v]$ *;*

(v) si $t \equiv \forall x^A.B$*, alors* $\Gamma, x : A \vdash^2 B : \mathrm{Prop}$ *est une sous-dérivation stricte avec* $T \equiv \mathrm{Prop}$*.*

Remarque

λ^2 n'ayant pas de règle de conversion des types, ce lemme de génération est plus précis : T n'est pas seulement β-convertible à un type donné mais lui est syntaxiquement égal.

Lemme 3.1.10 (correction des types, voir lem. 1.4.9).

Si $\Gamma \vdash^2 w : C$*, alors* $C \equiv \mathrm{Type}$ *ou bien il existe* κ *tel que* $\Gamma \vdash^2 C : \kappa$*.*

Lemme 3.1.11.

(i) Si $\Gamma \vdash^2 C : \mathrm{Type}$ *alors* $C \equiv \mathrm{Prop}$ *et la dernière règle de la dérivation est (ax) ;*

(ii) Si $\Gamma \vdash^2 C : \mathrm{Prop}$ *alors la dernière règle de la dérivation est (var) ou (prod).*

Preuve par cas sur la dernière règle utilisée.

(i) **(var)** Cas impossible car Type ne peut se trouver dans l'environnement (lem. 3.1.3).

(app) On a deux cas (lem. 1.2.2) :

- $B \equiv x$ et $v \equiv \mathrm{Type}$: ce qui est impossible (lem. 3.1.3) ;
- $B \equiv \mathrm{Type}$: d'où $\Gamma \vdash^2 \forall x^A.\,\mathrm{Type} : \kappa$ (lem. 3.1.10), ce qui est impossible (lem. 3.1.3).

(ii) **(app)** $\dfrac{\Gamma \vdash^2 u : \forall x^A.B \quad \Gamma \vdash^2 v : A}{\Gamma \vdash^2 u\ v : B[x \leftarrow v\,]}$

On a $\Gamma \vdash^2 \forall x^A.B : \kappa$ (lem. 3.1.10) puis par génération (lem. 3.1.9) $\Gamma, x : A \vdash^2 B : \mathrm{Prop}$.

De $B[x \leftarrow v\,] \equiv \mathrm{Prop}$, on a deux cas (lem. 1.2.2) :

- $B \equiv x$ et $v \equiv \mathrm{Prop}$: par génération (lem. 3.1.9) sur la seconde prémisse $A \equiv \mathrm{Type}$, ce qui est impossible (lem. 3.1.3) ;
- $B \equiv \mathrm{Prop}$: par génération (lem. 3.1.9) $\mathrm{Type} \equiv \mathrm{Prop}$ qui est impossible. □

Lemme 3.1.12.

(i) Si $x_1 : A_1, \ldots, x_n : A_n\ \mathsf{wf}^2$ *alors le symbole* λ *n'apparaît dans aucun* A_i *;*

(ii) Si $\Gamma \vdash^2 C : \kappa$ *alors le symbole* λ *n'apparaît pas dans* C*.*

Preuve par récurrence structurelle sur la dérivation ; les cas de (abs) et (app) ne sont pas à considérer (lem. 3.1.11).

(env$_2$) $\dfrac{\Gamma \vdash^2 A : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\ \mathsf{wf}^2}$

Comme $\Gamma\ \mathsf{wf}^2$ est une sous-dérivation (lem. 3.1.4) alors les types de Γ ne contiennent pas λ par hypothèse de récurrence ; de même pour A.

(prod) $\dfrac{\Gamma, x : A \vdash^2 B : \mathrm{Prop}}{\Gamma \vdash^2 \forall x^A.B : \mathrm{Prop}}$

Comme $\Gamma \vdash^2 A : \kappa$ est une sous-dérivation, alors par hypothèse de récurrence ni A ni B ne contiennent λ, et donc $\forall x^A.B$ non plus. □

Lemme 3.1.13. *Tout terme brut* u *réductible contient le symbole* λ*.*

Preuve par récurrence structurelle sur la dérivation de $u \rightsquigarrow_\beta u'$. □

Corollaire 3.1.14.

(i) Si $\Gamma \vdash^2 w : C$ *alors* C *est en forme normale ;*

(ii) Si $\Gamma \vdash^2 C : \kappa$ *alors* C *est en forme normale ;*

(iii) Si $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^2$ *alors tous les* A_i *sont en forme normale.*

Preuve

(i) On a soit $\Gamma \vdash^2 C : \kappa$, soit $C \equiv \mathrm{Type}$ (lem. 3.1.10). Dans les deux cas, C ne contient pas le symbole λ (lem. 3.1.12).

(ii) De même, C ne contient pas λ (lem. 3.1.12).

(iii) Comme $x_1 : A_1, \ldots, x_i : A_i \vdash^2 A_{i+1} : \kappa$ (lem. 3.1.5), alors les A_i ne contiennent pas λ. □

Lemme 3.1.15. *Si* $\Gamma \vdash^2 C : \mathrm{Prop}$*, alors pour tout* $x \in \mathcal{V}(C)$*,* $(x : \mathrm{Prop}) \in \Gamma$.

Preuve par récurrence structurelle sur la dérivation ; seules les règles (var) et (prod) sont à considérer (lem. 3.1.11). □

Théorème 3.1.16 (stabilité par réduction, voir thm. 1.4.11).

Si $\Gamma \vdash^2 t : C$ *et* $t \rightsquigarrow_\beta t'$*, alors* $\Gamma \vdash^2 t' : C$.

Preuve usuelle par récurrence structurelle sur la dérivation puis par cas sur la définition de $\rightsquigarrow_\beta$ (similaire à celle du thm. 1.4.11).

(abs) $\dfrac{\Gamma, x : A \vdash^2 u : B : \mathrm{Prop}}{\Gamma \vdash^2 \lambda x^A.u : \forall x^A.B}$

A étant en forme normale (cor. 3.1.14), seul le cas $u \rightsquigarrow_\beta u'$ est à traiter : immédiat par application de l'hypothèse de récurrence sur la première prémisse.

(app) $\dfrac{\Gamma \vdash^2 u : \forall x^A.B \quad \Gamma \vdash^2 v : A}{\Gamma \vdash^2 u\ v : B[x \leftarrow v\,]}$

On a trois cas :

- $u \rightsquigarrow_\beta u'$: immédiat par hypothèse de récurrence puis (app).
- $v \rightsquigarrow_\beta v'$: on a trois cas (lem. 3.1.10) :
  - $A \equiv \mathrm{Type}$: impossible (lem. 3.1.10 et 3.1.3) ;
  - $\Gamma \vdash^2 A : \mathrm{Type}$: alors $A \equiv \mathrm{Prop}$ (lem. 3.1.11) donc v est irréductible (cor. 3.1.14) ;
  - $\Gamma \vdash^2 A : \mathrm{Prop}$: alors $A \not\equiv \mathrm{Prop}$ (lem. 3.1.9) et donc $x \notin \mathcal{V}(B)$ (lem. 3.1.15) d'où on a $B[x \leftarrow v'\,] \equiv B \equiv B[x \leftarrow v\,]$, et il suffit d'appliquer l'hypothèse de récurrence sur la seconde prémisse puis (app).
- $u \equiv \lambda x^C.w$ et $u\ v \rightsquigarrow_\beta w[x \leftarrow v\,]$: par génération (lem. 3.1.9) on a $\Gamma, x : A \vdash^2 w : B$ et par substitution (lem. 3.1.8) on obtient le résultat.

(prod) Un terme de type Prop n'est pas réductible (cor. 3.1.14). □

3.2 Avec motivations totales explicites – λ_e^2

3.2.1 Définition du système

Termes bruts

◊ On ajoute aux termes bruts de λ^2 les constantes suivantes : o et $\top$.

Règles d'inférence de λ_e^2

◊ On définit la relation $\vdash^{2_e}$ entre un environnement, deux termes bruts et une substitution, ainsi que la relation wf^{2_e} entre un environnement et une substitution par les règles suivantes :

$$\frac{}{[\,]\,\mathsf{wf}^{2_e}_{[\,]}}\ (\text{e-env}_1) \qquad \frac{\Gamma \vdash^{2_e}_{\sigma} A : \kappa \quad \vdash^{2_e}_{[\,]} a : \sigma(A) \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^{2_e}_{\sigma::(x \mapsto a)}}\ (\text{e-env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^{2_e}_{\sigma}}{\Gamma \vdash^{2_e}_{\sigma} o : \top : \mathrm{Prop} : \mathrm{Type}}\ (\text{e-ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^{2_e}_{\sigma}}{\Gamma, x : A, \Gamma' \vdash^{2_e}_{\sigma} x : A}\ (\text{e-var})$$

$$\frac{\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_e}_{\sigma} \lambda x^A.u : \forall x^A.B}\ (\text{e-abs}) \qquad \frac{\Gamma \vdash^{2_e}_{\sigma} u : \forall x^A.B \quad \Gamma \vdash^{2_e}_{\sigma} v : A}{\Gamma \vdash^{2_e}_{\sigma} u\ v : B[x \leftarrow v]}\ (\text{e-app})$$

$$\frac{\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} B : \mathrm{Prop} \quad \vdash^{2_e}_{[\,]} t : \sigma(\forall x^A.B)}{\Gamma \vdash^{2_e}_{\sigma} \forall x^A.B : \mathrm{Prop}}\ (\text{e-prod})$$

Remarques

La prémisse supplémentaire sur (e-env$_2$) n'est pas une contrainte additionnelle : par construction le a est contenu dans la dérivation de la prémisse $\Gamma \vdash^{2_e}_{\sigma} A : \kappa$ (voir lem. 3.2.12).

Les substitutions et les environnements en relation par wf^{2_e} ou $\vdash^{2_e}$ se correspondent : ils ont même taille, et à chaque variable de l'environnement correspond un terme brut situé en même position dans la substitution (voir lem. 3.2.2).

3.2.2 Résultats

Théorème 3.2.1 (λ_e^2 est un sous-système de λ^2).

(i) si $\Gamma\,\mathsf{wf}^{2_e}_{\sigma}$, alors $\Gamma\,\mathsf{wf}^2$;

(ii) si $\Gamma \vdash^{2_e}_{\sigma} w : C$, alors $\Gamma \vdash^{2} w : C$.

Preuve immédiate par récurrence structurelle sur la dérivation : il suffit d'« oublier » les motivations explicites et d'interpréter les constantes o par $\lambda A^{\mathrm{Prop}}.\lambda x^A.x$ et $\top$ par $\forall A^{\mathrm{Prop}}.A \to A$.□

Lemme 3.2.2 (voir lem. 3.1.2).

(i) Si $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^{2_e}_{(y_1 \mapsto t_1)::\ldots::(y_m \mapsto t_m)}$, alors :
- *pour tout i, $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$;*
- *pour tout $i \neq j$ $x_i \not\equiv x_j$;*
- *pour tout i $x_i \equiv y_i$ et t_i est clos ;*

– $m = n$.

(ii) Si $x_1 : A_1, \ldots, x_n : A_n \vdash^{2_e}_{(y_1 \mapsto t_1)::\ldots::(y_m \mapsto t_m)} w : C$, *alors :*
– *pour tout* i, $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$;
– *pour tout* $i \neq j$ $x_i \not\equiv x_j$;
– $\mathcal{V}(w, C) \subseteq \{x_1, \ldots, x_n\}$;
– *pour tout* i $x_i \equiv y_i$ *et* t_i *est clos ;*
– $m = n$.

Lemme 3.2.3 (voir lem. 3.1.3). *Si* $\Gamma\, \mathsf{wf}^{2_e}_{\sigma}$ *ou* $\Gamma \vdash^{2_e}_{\sigma} w : C$, *alors* $\mathrm{Type} \notin \mathcal{S}(\Gamma)$ *et* $\mathrm{Type} \notin \mathcal{S}(w)$.

Lemme 3.2.4 (validité des environnements, voir lem. 3.1.4).

(i) si $\Gamma\, \mathsf{wf}^{2_e}_{\sigma}$, *alors pour tout environnement* $\Gamma' \preccurlyeq \Gamma$, $\Gamma'\, \mathsf{wf}^{2_e}_{\sigma'}$ *en est une sous-dérivation avec* $\sigma' \preccurlyeq \sigma$ *correspondant à* Γ' ;

(ii) si $\Gamma \vdash^{2_e}_{\sigma} w : C$, *alors pour tout environnement* $\Gamma' \preccurlyeq \Gamma$, $\Gamma'\, \mathsf{wf}^{2_e}_{\sigma'}$ *en est une sous-dérivation stricte avec* $\sigma' \preccurlyeq \sigma$ *correspondant à* Γ'.

Lemme 3.2.5 (validité des types des environnements, voir lem. 3.1.5).

Si $x_1 : A_1, \ldots, x_n : A_n\, \mathsf{wf}^{2_e}_{\sigma}$, *alors pour tout* i *il existe* κ *tel que* $x_1 : A_1, \ldots, x_i : A_i \vdash^{2_e}_{\sigma_{\leq i}} A_{i+1} : \kappa$ *en est une sous-dérivation stricte.*

Lemme 3.2.6 (unicité du type, voir lem. 3.1.7). *Si* $\Gamma \vdash^{2_e}_{\sigma} w : C$ *et* $\Gamma \vdash^{2_e}_{\sigma} w : C'$, *alors* $C \equiv C'$.

Preuve est une conséquence immédiate que λ^2_e est un sous-système de λ^2 (thm. 3.2.1). □

Lemme 3.2.7 (génération, voir lem. 3.1.9). *Soit* $\Gamma \vdash^{2_e}_{\sigma} t : T$, *un de ces cas se présente :*

(i) si $t \equiv o$, *alors* $T \equiv \top$;

(ii) si $t \equiv \top$, *alors* $T \equiv \mathrm{Prop}$;

(iii) si $t \equiv \mathrm{Prop}$, *alors* $T \equiv \mathrm{Type}$;

(iv) si $t \equiv x$, *alors il existe* $(x : A) \in \Gamma$ *avec* $T \equiv A$;

(v) si $t \equiv \lambda x^A.u$, *alors il existe* B *et* a *tels que* $\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} u : B : \mathrm{Prop}$ *est une sous-dérivation stricte avec* $T \equiv \forall x^A.B$;

(vi) si $t \equiv u\ v$, *alors il existe* A *et* B *tels que* $\Gamma \vdash^{2} u : \forall x^A.B$ *et* $\Gamma \vdash^{2} v : A$ *sont des sous-dérivations strictes avec* $T \equiv B[x \leftarrow v]$;

(vii) si $t \equiv \forall x^A.B$, *alors il existe* a *et* t *tels que* $\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} B : \mathrm{Prop}$ *et* $\vdash^{2_e}_{[]} t : \sigma(\forall x^A.B)$ *sont des sous-dérivations strictes avec* $T \equiv \mathrm{Prop}$.

Lemme 3.2.8.

(i) Si $\Gamma \vdash^{2_e}_{\sigma} C : \mathrm{Type}$ *alors* $C \equiv \mathrm{Prop}$ *et la dernière règle de la dérivation est (e-ax) ;*

(ii) Si $\Gamma \vdash^{2_e}_{\sigma} C : \mathrm{Prop}$ *alors la dernière règle de la dérivation est (e-ax) ou (e-var) ou (e-prod).*

Preuve par cas sur la dernière règle utilisée, similaire à celle pour λ^2 (lem. 3.1.11) : pour montrer qu'une dérivation est impossible pour λ^2_e, il suffit d'utiliser le fait que λ^2_e est un sous-système de λ^2 (thm. 3.2.1) et de montrer que la dérivation correspondante est déjà impossible dans λ^2.

(i) **(e-var)** Impossible (lem. 3.2.3).

(e-app) On aurait $\Gamma \vdash^{2_e}_{\sigma} u\ v : \mathrm{Type}$ d'où $\Gamma \vdash^{2} u\ v : \mathrm{Type}$ (thm. 3.2.1) puis $u\ v \equiv \mathrm{Prop}$ (lem. 3.1.11) ce qui est impossible.

(ii) **(e-app)** $\dfrac{\Gamma \vdash^{2_e}_{\sigma} u : \forall x^A.B \quad \Gamma \vdash^{2_e}_{\sigma} v : A}{\Gamma \vdash^{2_e}_{\sigma} u\ v : B[x \leftarrow v]}$ avec $B[x \leftarrow v] \equiv \mathrm{Prop}$.

On a deux cas (lem. 1.2.2) :

- $B \equiv x$ et $v \equiv \mathrm{Prop}$: par génération (lem. 3.2.7) $A \equiv \mathrm{Type}$ et donc (thm. 3.2.1) $\Gamma \vdash^{2} u : \forall x^{\mathrm{Type}}.B$ ce qui est impossible.
- $B \equiv \mathrm{Prop}$: or $\Gamma \vdash^{2} u : \forall x^A.\,\mathrm{Prop}$ est impossible. □

Lemme 3.2.9 (voir cor. 3.1.14).

(i) Si $\Gamma \vdash^{2_e}_{\sigma} w : C$ alors C est en forme normale ;

(ii) Si $\Gamma \vdash^{2_e}_{\sigma} C : \kappa$ alors C est en forme normale ;

(iii) Si $x_1 : A_1, \ldots, x_n : A_n\ \mathsf{wf}^{2_e}_{\sigma}$ alors tous les A_i sont en forme normale.

Preuve est une conséquence immédiate que λ_e^2 est un sous-système de λ^2 (thm. 3.2.1). □

Lemme 3.2.10 (voir lem. 3.1.15). *Si $\Gamma \vdash^{2_e}_{\sigma} C : \mathrm{Prop}$, alors pour tout $x \in \mathcal{V}(C)$, $(x : \mathrm{Prop}) \in \Gamma$.*

Preuve est une conséquence immédiate que λ_e^2 est un sous-système de λ^2 (thm. 3.2.1). □

Proposition 3.2.11 (λ_e^2 vérifie le critère de Poincaré).

Si $x_1 : A_1, \ldots, x_n : A_n\ \mathsf{wf}^{2_e}_{\sigma}$, alors

$$\forall i \quad \vdash^{2_e}_{[]} \sigma(x_i) : \sigma(A_i)$$

*en **sont des sous-dérivations strictes**.*

Preuve par récurrence structurelle sur la dérivation de $x_1 : A_1, \ldots, x_n : A_n\ \mathsf{wf}^{2_e}_{\sigma}$.

(e-env$_2$) $\dfrac{\Gamma \vdash^{2_e}_{\sigma} A : \kappa \quad \vdash^{2_e}_{[]} a : \sigma(A) \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\ \mathsf{wf}^{2_e}_{\sigma::(x \mapsto a)}}$

De $\Gamma \vdash^{2_e}_{\sigma} A : \kappa$ on a $\Gamma\ \mathsf{wf}^{2_e}_{\sigma}$ en est une sous-dérivation (lem. 3.2.4), d'où par hypothèse de récurrence, avec $\Gamma := y_1 : B_1, \ldots, y_n : B_n$

$$\forall i \quad \vdash^{2_e}_{[]} \sigma(y_i) : \sigma(B_i)$$

sont des sous-dérivations strictes de $\Gamma \vdash^{2_e}_{\sigma} A : \kappa$. La seconde prémisse permet de conclure. □

Lemme 3.2.12. *Si $\Gamma \vdash^{2_e}_{\sigma} C : \kappa$, alors il existe un terme t tel que $\vdash^{2_e}_{[]} t : \sigma(C)$.*

Preuve par récurrence structurelle sur la dérivation. Les seules règles à regarder sont (e-ax), (e-prod) et (e-var) (lem. 3.2.8), et seul le cas de (e-var) est non immédiat :

(e-var) $\dfrac{\Gamma, x : \mathrm{Prop}, \Gamma'\ \mathsf{wf}^{2_e}_{\sigma}}{\Gamma, x : \mathrm{Prop}, \Gamma' \vdash^{2_e}_{\sigma} x : \mathrm{Prop}}$

D'après le critère de Poincaré (prop. 3.2.11) appliqué à la prémisse, $\vdash^{2_e}_{[]} \sigma(x) : \mathrm{Prop}$ en est une sous-dérivation stricte. D'où, par hypothèse de récurrence, il existe un terme t tel que $\vdash^{2_e}_{[]} t : \sigma(x)$. □

Remarque

Ce lemme justifie le fait que la prémisse additionnelle de la règle (e-env$_2$) n'est pas une contrainte additionnelle.

Lemme 3.2.13 (affaiblissement). *Si* $\Gamma \vdash^{2_e}_{\sigma} w : C$, $\Gamma' \,\mathsf{wf}^{2_e}_{\sigma'}$, $\Gamma \subseteq \Gamma'$ *et* $\sigma \subseteq \sigma'$, *alors* $\Gamma' \vdash^{2_e}_{\sigma'} w : C$.

Preuve par récurrence structurelle sur la dérivation ; similaire à celle du lemme 3.1.6.

(e-abs) $\dfrac{\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_e}_{\sigma} \lambda x^A.u : \forall x^A.B}$ Soient $\Gamma' \,\mathsf{wf}^{2_e}_{\sigma'}$ avec $\Gamma \subseteq \Gamma'$ et $\sigma \subseteq \sigma'$.

D'une prémisse, on déduit que $\Gamma \vdash^{2_e}_{\sigma} A : \kappa$ en est une sous-dérivation (lem. 3.2.5), sur laquelle on applique l'hypothèse de récurrence pour obtenir

$$\Gamma' \vdash^{2_e}_{\sigma'} A : \kappa$$

et comme aussi (prop. 3.2.11)

$$\vdash^{2_e}_{[]} a : \sigma(A) \text{ d'où } \vdash^{2_e}_{[]} a : \sigma'(A)$$

alors finalement par (e-env$_2$) on obtient

$$\Gamma', x : A \,\mathsf{wf}^{2_e}_{\sigma'::(x \mapsto a)}$$

L'hypothèse de récurrence sur les prémisses donne $\Gamma', x : A \vdash^{2_e}_{\sigma'::(x \mapsto a)} u : B : \mathrm{Prop}$ et (e-abs) le résultat.

(e-prod) On effectue de même que pour (e-abs) afin de pouvoir appliquer l'hypothèse de récurrence et en déduire le résultat par (e-prod). □

Lemme 3.2.14.

(i) si $\Gamma[z \leftarrow w] \,\mathsf{wf}^{2_e}_{\sigma}$ *et* $\vdash^{2_e}_{[]} w : C : \kappa$ *avec* $z \notin \mathrm{dom}(\Gamma)$, *alors* $z : C, \Gamma \,\mathsf{wf}^{2_e}_{(z \mapsto w)::\sigma}$;

(ii) si $\Gamma[z \leftarrow w] \vdash^{2_e}_{\sigma} D[z \leftarrow w] : \kappa'$ *et* $\vdash^{2_e}_{[]} w : C : \kappa$ *avec* $z \notin \mathrm{dom}(\Gamma)$, *alors* $z : C, \Gamma \vdash^{2_e}_{(z \mapsto w)::\sigma} D : \kappa'$.

Preuve par récurrence structurelle sur la dérivation.

(i)

(e-env$_1$) De $\vdash^{2_e}_{[]} w : C : \kappa$ par (e-env$_2$) on a bien $z : C \,\mathsf{wf}^{2_e}_{[(z \mapsto w)]}$.

(e-env$_2$) $\dfrac{\Gamma[z \leftarrow w] \vdash^{2_e}_{\sigma} A[z \leftarrow w] : \kappa'' \quad \vdash^{2_e}_{[]} a : \sigma(A[z \leftarrow w]) \quad x \notin \mathrm{dom}(\Gamma[z \leftarrow w])}{\Gamma[z \leftarrow w], x : A[z \leftarrow w] \,\mathsf{wf}^{2_e}_{\sigma::(x \mapsto a)}}$

Par hypothèse de récurrence on a

$$z : C, \Gamma \vdash^{2_e}_{(z \mapsto w)::\sigma} A : \kappa''$$

et comme de plus (lem. 1.2.5)

$$\vdash^{2_e}_{[]} a : (z \mapsto w)::\sigma(A)$$

puisque w clos (lem. 3.2.2), alors par (e-env$_2$) on a bien le résultat.

(ii) Seules les règles (e-ax), (e-var) et (e-prod) sont à considérer (lem. 3.2.8).

Pour toutes les règles suivantes, on peut toujours traiter le cas où $D \equiv z$ de la façon suivante :

De $\Gamma[z \leftarrow w] \vdash^{2_e}_{\sigma} D[z \leftarrow w] : \kappa'$ on a que

$$\Gamma[z \leftarrow w]\, \mathsf{wf}^{2_e}_{\sigma}$$

en est une sous-dérivation stricte (lem. 3.2.4), puis par hypothèse de récurrence

$$z : C, \Gamma\, \mathsf{wf}^{2_e}_{(z \mapsto w) :: \sigma}$$

et par la règle (e-var) on obtient

$$z : C, \Gamma \vdash^{2_e}_{(z \mapsto w) :: \sigma} z : C$$

Or $C \equiv \kappa'$ par unicité du type (lem. 3.2.6) puisque :

- on a $\Gamma[z \leftarrow w] \vdash^{2_e}_{\sigma} w : \kappa'$ par hypothèse ;
- de $\vdash^{2_e}_{[]} w : C$ on a $\Gamma[z \leftarrow w] \vdash^{2_e}_{\sigma} w : C$ par affaiblissement (lem. 3.2.13).

(e-ax) $\dfrac{\Gamma[z \leftarrow w]\, \mathsf{wf}^{2_e}_{\sigma}}{\Gamma[z \leftarrow w] \vdash^{2_e}_{\sigma} \top : \mathrm{Prop}}$ avec $D[z \leftarrow w] \equiv \top$ et $D \not\equiv z$, d'où $D \equiv \top$.

Par hypothèse de récurrence $z : C, \Gamma\, \mathsf{wf}^{2_e}_{(z \mapsto w) :: \sigma}$ et par (e-ax) $z : C, \Gamma \vdash^{2_e}_{(z \mapsto w) :: \sigma} \top : \mathrm{Prop}$. De même pour $\Gamma[z \leftarrow w] \vdash^{2_e}_{\sigma} \mathrm{Prop} : \mathrm{Type}$.

(e-var) $\dfrac{\Gamma[z \leftarrow w], x : \kappa', \Gamma'[z \leftarrow w]\, \mathsf{wf}^{2_e}_{\sigma :: (x \mapsto t) :: \sigma'}}{\Gamma[z \leftarrow w], x : \kappa', \Gamma'[z \leftarrow w] \vdash^{2_e}_{\sigma :: (x \mapsto t) :: \sigma'} x : \kappa'}$ avec $D[z \leftarrow w] \equiv x$ et $D \not\equiv z$, d'où $D \equiv x$.

L'hypothèse de récurrence donne $z : C, \Gamma, x : \kappa', \Gamma'\, \mathsf{wf}^{2_e}_{(z \mapsto w) :: \sigma :: (x \mapsto t) :: \sigma'}$ puis (e-var) le résultat.

(e-prod) $\dfrac{\Gamma[z \leftarrow w], x : A[z \leftarrow w] \vdash^{2_e}_{\sigma :: (x \mapsto a)} B[z \leftarrow w] : \mathrm{Prop} \quad \vdash^{2_e}_{[]} t : \sigma(\forall x^{A[z \leftarrow w]}.B[z \leftarrow w])}{\Gamma[z \leftarrow w] \vdash^{2_e}_{\sigma} \forall x^{A[z \leftarrow w]}.B[z \leftarrow w] : \mathrm{Prop}}$

Par hypothèse de récurrence on a

$$z : C, \Gamma, x : A \vdash^{2_e}_{(z \mapsto w) :: \sigma :: (x \mapsto a)} B : \mathrm{Prop}$$

de plus $\vdash^{2_e}_{[]} t : \sigma(\forall x^{A[z \leftarrow w]}.B[z \leftarrow w])$ s'écrit aussi (lem. 1.2.5)

$$\vdash^{2_e}_{[]} t : (z \mapsto w) :: \sigma(\forall x^{A}.B)$$

d'où le résultat par (e-prod). □

Proposition 3.2.15 (λ_e^2 vérifie la réciproque du critère de Poincaré).
Si

$$\vdash^{2_e}_{[]} t_1 : A_1 : \kappa_1 \quad \vdash^{2_e}_{[]} t_2 : A_2[x_1 \leftarrow t_1] : \kappa_2 \quad \ldots \quad \vdash^{2_e}_{[]} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}] : \kappa_n$$

(avec les x_i distinctes deux à deux), alors

$$x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n\, \mathsf{wf}^{2_e}_{(x_1 \mapsto t_1) :: (x_2 \mapsto t_2) :: \ldots :: (x_n \mapsto t_n)}$$

Preuve par récurrence sur n.

Par hypothèse, $\vdash^{2_e}_{[]} A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}] : \kappa_n$, or les x_i étant distinctes deux à deux et les t_i clos (lem. 3.2.2), cela s'écrit aussi (lem 1.2.5)

$$\vdash^{2_e}_{[]} A_n[x_1, \ldots, x_{n-2} \leftarrow t_1, \ldots, t_{n-2}][x_{n-1} \leftarrow t_{n-1}] : \kappa_n$$

et comme

$$\vdash^{2_e}_{[]} t_{n-1} : A_{n-1}[x_1, \ldots, x_{n-2} \leftarrow t_1, \ldots, t_{n-2}] : \kappa_{n-1}$$

alors (lem. 3.2.14) on obtient

$$x_{n-1} : A_{n-1}[x_1, \ldots, x_{n-2} \leftarrow t_1, \ldots, t_{n-2}] \vdash^{2_e}_{(x_{n-1} \mapsto t_{n-1})} A_n[x_1, \ldots, x_{n-2} \leftarrow t_1, \ldots, t_{n-2}] : \kappa_n$$

$$\vdots$$

$$x_1 : A_1, \ldots, x_{n-1} : A_{n-1} \vdash^{2_e}_{(x_1 \mapsto t_1)::\ldots::(x_{n-1} \mapsto t_{n-1})} A_n : \kappa_n$$

puis, comme

$$\vdash^{2_e}_{[]} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$$

alors par (e-env$_2$) on obtient bien

$$x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{2_e}_{(x_1 \mapsto t_1)::\ldots::(x_n \mapsto t_n)}$$

Remarque

Dans la preuve précédente, les « $\vdots$ » cachent une récurrence qui peut être explicitée de la façon suivante : soit $\sigma := (x_1 \mapsto t_1)::\ldots::(x_n \mapsto t_n)$, et soit la propriété

$$P(k) := \left.\begin{array}{c} x_{n-k} : \sigma_{<n-k}(A_{n-k}), \\ x_{n-k+1} : \sigma_{<n-k}(A_{n-k+1}), \\ \vdots \\ x_{n-k+(k-1)} : \sigma_{<n-k}(A_{n-k+(k-1)}) \end{array}\right\} \vdash^{2_e}_{\sigma_{[\![n-k,n-1]\!]}} \sigma_{<n-k}(A_n) : \kappa_n$$

avec $\sigma_{[\![i,j]\!]} := (x_i \mapsto t_i)::\ldots::(x_j \mapsto t_j)$.

La preuve cachée est alors la démonstration par récurrence sur k que la propriété $P(k)$ est correcte pour $1 \leq k \leq n-1$.

Lemme 3.2.16.

(i) Si $z : C, \Gamma \,\mathsf{wf}^{2_e}_{(z \mapsto c)::\sigma}$ *alors* $\Gamma[z \leftarrow c] \,\mathsf{wf}^{2_e}_{\sigma}$*;*

(ii) Si $z : C, \Gamma \vdash^{2_e}_{(z \mapsto c)::\sigma} w : D$ *alors* $\Gamma[z \leftarrow c] \vdash^{2_e}_{\sigma} w[z \leftarrow c] : D[z \leftarrow c]$.

Preuve par récurrence structurelle sur la dérivation.

(e-var) Le seul cas non trivial est : $\dfrac{z : C, \Gamma \,\mathsf{wf}^{2_e}_{(z \mapsto c)::\sigma}}{z : C, \Gamma \vdash^{2_e}_{(z \mapsto c)::\sigma} z : C}$

Par hypothèse de récurrence, on a $\Gamma[z \leftarrow c] \,\mathsf{wf}^{2_e}_{\sigma}$.

Or $\vdash^{2_e}_{[]} c : C$ par le critère de Poincaré (prop. 3.2.11), puis par affaiblissement (lem. 3.2.13) on obtient bien $\Gamma[z \leftarrow c] \vdash^{2_e}_{\sigma} c : C$.

$$\textbf{(e-app)}\ \frac{z : C, \Gamma \vdash^{2_e}_{(z \mapsto c)::\sigma} u : \forall x^A.B \quad z : C, \Gamma \vdash^{2_e}_{(z \mapsto c)::\sigma} v : A}{z : C, \Gamma \vdash^{2_e}_{(z \mapsto c)::\sigma} u\ v : B[x \leftarrow v]}$$

Par hypothèse de récurrence on a

$$\Gamma[z \leftarrow c] \vdash^{2_e}_{\sigma} u[z \leftarrow c] : \forall x^{A[z \leftarrow c]}.B[z \leftarrow c] \qquad \text{et} \qquad \Gamma[z \leftarrow c] \vdash^{2_e}_{\sigma} v[z \leftarrow c] : A[z \leftarrow c]$$

d'où par (e-app)

$$\Gamma[z \leftarrow c] \vdash^{2_e}_{\sigma} u[z \leftarrow c]\ v[z \leftarrow c] : B[z \leftarrow c][x \leftarrow v[z \leftarrow c]]$$

or comme c clos (lem. 3.2.2), alors (lem. 1.2.6)

$$B[z \leftarrow c][x \leftarrow v[z \leftarrow c]] \equiv B[x \leftarrow v][z \leftarrow c]$$

$$\textbf{(e-prod)}\ \frac{z : C, \Gamma, x : A \vdash^{2_e}_{(z \mapsto c)::\sigma::(x \mapsto a)} B : \mathrm{Prop} \quad \vdash^{2_e}_{[]} t : (z \mapsto c)::\sigma(\forall x^A.B)}{z : C, \Gamma \vdash^{2_e}_{(z \mapsto c)::\sigma} \forall x^A.B : \mathrm{Prop}}$$

Par hypothèse de récurrence

$$\Gamma[z \leftarrow c], x : A[z \leftarrow c] \vdash^{2_e}_{\sigma::(x \mapsto a)} B[z \leftarrow c] : \mathrm{Prop}$$

Or comme c clos et $z \notin \mathrm{dom}(\sigma)$ (lem. 3.2.2), alors (lem. 1.2.5)

$$(z \mapsto c)::\sigma(\forall x^A.B) \equiv \sigma((\forall x^A.B)[z \leftarrow c])$$

et donc on a

$$\vdash^{2_e}_{[]} t : \sigma(\forall x^{A[z \leftarrow c]}.B[z \leftarrow c])$$

ce qui permet de conclure en appliquant la règle (e-prod). □

Lemme 3.2.17. *Si* $\Gamma \vdash^{2_e}_{\sigma} w : C$ *alors* $\vdash^{2_e}_{[]} \sigma(w) : \sigma(C)$.

Preuve par récurrence sur la taille de l'environnement.

Si $\Gamma := x_1 : A_1, \ldots, x_n : A_n$ et $\sigma := (x_1 \mapsto t_1)::\ldots::(x_n \mapsto t_n)$, en substituant n fois les motivations (lem. 3.2.16), on obtient

$$\vdash^{2_e}_{[]} w[x_1 \leftarrow t_1] \ldots [x_n \leftarrow t_n] : D[x_1 \leftarrow t_1] \ldots [x_n \leftarrow t_n]$$

Or puisque les t_i sont clos et les x_i tous différents (lem. 3.2.2), alors (lem. 1.2.5)

$$w[x_1 \leftarrow t_1] \ldots [x_n \leftarrow t_n] \equiv w[x_1, \ldots, x_n \leftarrow t_1, \ldots, t_n]$$

$$D[x_1 \leftarrow t_1] \ldots [x_n \leftarrow t_n] \equiv D[x_1, \ldots, x_n \leftarrow t_1, \ldots, t_n]$$

□

Lemme 3.2.18. *Si* $\Gamma, z : C, \Gamma' \vdash^{2_e}_{\sigma} w : D$ *et* $z \notin \mathcal{V}(\Gamma', w)$, *alors* $z \notin \mathcal{V}(D)$.

Preuve immédiate par récurrence structurelle sur la dérivation. □

Lemme 3.2.19 (renforcement).

(i) Si $\Gamma, z : C, \Gamma' \ \mathsf{wf}^{2_e}_{\sigma::(z \mapsto c)::\sigma'}$ *et* $z \notin \mathcal{V}(\Gamma')$, *alors* $\Gamma, \Gamma' \ \mathsf{wf}^{2_e}_{\sigma::\sigma'}$ *;*

(ii) Si $\Gamma, z : C, \Gamma' \vdash^{2_e}_{\sigma::(z \mapsto c)::\sigma'} w : D$ *et* $z \notin \mathcal{V}(\Gamma', w)$*, alors* $\Gamma, \Gamma' \vdash^{2_e}_{\sigma::\sigma'} w : D$.

Preuve par récurrence structurelle sur la dérivation ; similaire à (Luo, 1990, lem. 3.2.9).

Le seul cas non trivial est le suivant :

$$\textbf{(e-abs)}\ \frac{\Gamma, z : C, \Gamma', x : A \vdash^{2_e}_{\sigma::(z \mapsto c)::\sigma'::(x \mapsto a)} u : B : \mathrm{Prop}}{\Gamma, z : C, \Gamma' \vdash^{2_e}_{\sigma::(z \mapsto c)::\sigma'} \lambda x^A.u : \forall x^A.B} \quad \text{avec } z \notin \mathcal{V}(\Gamma', \lambda x^A.u).$$

Alors $z \notin \mathcal{V}(\Gamma', A, u)$, et donc aussi $z \notin \mathcal{V}(B)$ (lem. 3.2.18). On peut alors appliquer l'hypothèse de récurrence pour obtenir $\Gamma, \Gamma', x : A \vdash^{2_e}_{\sigma::\sigma'::(x \mapsto a)} u : B : \mathrm{Prop}$ puis par (e-abs) le résultat. □

Lemme 3.2.20. *Si* $\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} u : B : \mathrm{Prop}$*, alors* $\Gamma \vdash^{2_e}_{\sigma} \lambda x^A.u : \forall x^A.B : \mathrm{Prop}$.

Preuve

Par (e-abs) sur l'hypothèse on a $\Gamma \vdash^{2_e}_{\sigma} \lambda x^A.u : \forall x^A.B$, d'où on obtient $\vdash^{2_e}_{[]} \sigma(\lambda x^A.u) : \sigma(\forall x^A.B)$ (lem. 3.2.17), ce qui permet d'appliquer (e-prod) pour avoir finalement le résultat. □

Lemme 3.2.21. *Si* $\Gamma\, \mathsf{wf}^{2_e}_{\sigma}$ *et* $\vdash^{2_e}_{[]} c : \sigma(C) : \kappa$ *avec* $z \notin \mathrm{dom}(\Gamma)$*, alors* $\Gamma, z : C\, \mathsf{wf}^{2_e}_{\sigma::(z \mapsto c)}$.

Preuve

Posons $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$ et $\sigma \equiv (x_1 \mapsto a_1)::\ldots::(x_n \mapsto a_n)$. D'après le critère de Poincaré (prop. 3.2.11)

$$\vdash^{2_e}_{[]} a_1 : A_1 \quad \vdash^{2_e}_{[]} a_2 : A_2[x_1 \leftarrow a_1] \quad \ldots \quad \vdash^{2_e}_{[]} a_n : A_n[x_1, \ldots, x_{n-1} \leftarrow a_1, \ldots, a_{n-1}]$$

et comme pour tout i (lem. 3.2.5)

$$x_1 : A_1, \ldots, x_{i-1} : A_{i-1} \vdash^{2_e}_{\sigma_{<i}} A_i : \kappa_i$$

alors (lem. 3.2.17)

$$\vdash^{2_e}_{[]} A_i[x_1, \ldots, x_{i-1} \leftarrow a_1, \ldots, a_{i-1}] : \kappa_i$$

Le résultat s'obtient en appliquant la réciproque du critère de Poincaré (prop. 3.2.15) sur :

$$\vdash^{2_e}_{[]} a_1 : A_1 : \kappa_1 \quad \vdash^{2_e}_{[]} a_2 : A_2[x_1 \leftarrow a_1] : \kappa_2 \quad \ldots \quad \vdash^{2_e}_{[]} a_n : A_n[x_1, \ldots, x_{n-1} \leftarrow a_1, \ldots, a_{n-1}] : \kappa_n$$

$$\vdash^{2_e}_{[]} c : C[x_1, \ldots, x_n \leftarrow a_1, \ldots, a_n] : \kappa$$

□

Théorème 3.2.22 (remplacement d'équivalents).

Si $\Gamma \vdash^{2_e}_{\sigma} w : E[z_1, \ldots, z_n \leftarrow C_1, \ldots, C_n] : \mathrm{Prop}$ *et il existe des termes* $(f_i)_{1 \leq i \leq n}$ *et* $(g_i)_{1 \leq i \leq n}$ *tels que pour tout i*

$$\Gamma \vdash^{2_e}_{\sigma} f_i : C_i \to D_i \quad \Gamma \vdash^{2_e}_{\sigma} g_i : D_i \to C_i$$

et

$$\Gamma \vdash^{2_e}_{\sigma} C_i : \mathrm{Prop} \quad \Gamma \vdash^{2_e}_{\sigma} D_i : \mathrm{Prop}$$

alors il existe un terme w' *tel que* $\Gamma \vdash^{2_e}_{\sigma} w' : E[z_1, \ldots, z_n \leftarrow D_1, \ldots, D_n] : \mathrm{Prop}$.

Preuve par récurrence sur le terme brut E (généralise (Colson et Michel, 2009, lem. 14)).

Constatons tout d'abord que si $E \equiv z_i$, alors $w' := f_i\ w$ convient.

Traitons le cas où E est différent de tous les z_i. Procédons par cas sur la dernière règle utilisée pour obtenir $\Gamma \vdash^{2_e}_{\sigma} E[z_1, \ldots, z_n \leftarrow C_1, \ldots, C_n] : \mathrm{Prop}$, ce qui limite l'analyse à trois règles (lem. 3.2.8) :

(e-ax) Dans ce cas $E \equiv \top$ et alors $w' := w$ convient.

(e-var) Dans ce cas $E \equiv y$ une variable différente des z_i et alors $w' := w$ convient.

(e-prod) On note $F[\vec{z} \leftarrow \vec{C}]$ pour $F[z_1, \ldots, z_n \leftarrow C_1, \ldots, C_n]$ afin d'alléger les notations.

$$\frac{\Gamma, x : A[\vec{z} \leftarrow \vec{C}] \vdash^{2_e}_{\sigma::(x \mapsto a)} B[\vec{z} \leftarrow \vec{C}] : \mathrm{Prop} \qquad \vdash^{2_e}_{[]} t : \sigma(\forall x^{A[\vec{z} \leftarrow \vec{C}]}.B[\vec{z} \leftarrow \vec{C}])}{\Gamma \vdash^{2_e}_{\sigma} \forall x^{A[\vec{z} \leftarrow \vec{C}]}.B[\vec{z} \leftarrow \vec{C}] : \mathrm{Prop}}$$

Comme $\Gamma \vdash^{2_e}_{\sigma} A[\vec{z} \leftarrow \vec{C}] : \kappa$ (lem. 3.2.4 et 3.2.5), on distingue deux cas selon la valeur de κ :

- $\kappa \equiv \mathrm{Type}$: alors $A[\vec{z} \leftarrow \vec{C}] \equiv \mathrm{Prop}$ (lem. 3.2.8).
  Si $A \equiv z_i$ alors $\Gamma \vdash^{2_e}_{\sigma} C_i : \mathrm{Type}$, ce qui n'est pas possible par unicité du type (lem. 3.2.6).
  Nécessairement, $A \not\equiv z_i$ pour tout i, et donc $A \equiv \mathrm{Prop}$.
  La règle s'écrit alors plus simplement ainsi :

$$\frac{\Gamma, x : \mathrm{Prop} \vdash^{2_e}_{\sigma::(x \mapsto a)} B[\vec{z} \leftarrow \vec{C}] : \mathrm{Prop} \qquad \vdash^{2_e}_{[]} t : \sigma(\forall x^{\mathrm{Prop}}.B[\vec{z} \leftarrow \vec{C}])}{\Gamma \vdash^{2_e}_{\sigma} \forall x^{\mathrm{Prop}}.B[\vec{z} \leftarrow \vec{C}] : \mathrm{Prop}}$$

  Par affaiblissement (lem. 3.2.13) sur $\Gamma \vdash^{2_e}_{\sigma} w : \forall x^{\mathrm{Prop}}.B[\vec{z} \leftarrow \vec{C}] : \mathrm{Prop}$ avec l'environnement $\Gamma, x : \mathrm{Prop}\ \mathsf{wf}^{2_e}_{\sigma::(x \mapsto a)}$ (lem. 3.2.4), on obtient

$$\Gamma, x : \mathrm{Prop} \vdash^{2_e}_{\sigma::(x \mapsto a)} w : \forall x^{\mathrm{Prop}}.B[\vec{z} \leftarrow \vec{C}] : \mathrm{Prop}$$

  puis par (e-var) et (e-app)

$$\Gamma, x : \mathrm{Prop} \vdash^{2_e}_{\sigma::(x \mapsto a)} w\ x : B[\vec{z} \leftarrow \vec{C}]$$

  et comme aussi

$$\Gamma, x : \mathrm{Prop} \vdash^{2_e}_{\sigma::(x \mapsto a)} B[\vec{z} \leftarrow \vec{C}] : \mathrm{Prop}$$

  alors par hypothèse de récurrence il existe un terme u tel que

$$\Gamma, x : \mathrm{Prop} \vdash^{2_e}_{\sigma::(x \mapsto a)} u : B[\vec{z} \leftarrow \vec{D}] : \mathrm{Prop}$$

  d'où (lem. 3.2.20)

$$\Gamma \vdash^{2_e}_{\sigma} \lambda x^{\mathrm{Prop}}.u : \forall x^{\mathrm{Prop}}.B[\vec{z} \leftarrow \vec{D}] : \mathrm{Prop}$$

  c'est-à-dire que $w' := \lambda x^{\mathrm{Prop}}.u$ convient.

- $\kappa \equiv \mathrm{Prop}$: alors $A[\vec{z} \leftarrow \vec{C}] \not\equiv \mathrm{Prop}$ (lem. 3.2.7) et donc $x \notin \mathcal{V}(B[\vec{z} \leftarrow \vec{C}])$ (lem. 3.2.10).
  De la première prémisse, on a (prop. 3.2.11)

$$\vdash^{2_e}_{[]} a : \sigma(A[\vec{z} \leftarrow \vec{C}]) : \mathrm{Prop}$$

ce qui s'écrit aussi (lem. 1.2.5)

$$\vdash^{2_e}_{[]} a : A[\vec{z}, \vec{y} \leftarrow \sigma(\vec{C}), \sigma(\vec{y})\,] : \mathrm{Prop}$$

avec $\vec{y}$ les variables libres de $A[\vec{z} \leftarrow \vec{C}\,]$. Or nous avons (lem. 3.2.17)

$$\vdash^{2_e}_{[]} \sigma(f_i) : \sigma(C_i) \to \sigma(D_i) \qquad \vdash^{2_e}_{[]} \sigma(g_i) : \sigma(D_i) \to \sigma(C_i)$$
$$\vdash^{2_e}_{[]} \sigma(C_i) : \mathrm{Prop} \qquad \vdash^{2_e}_{[]} \sigma(D_i) : \mathrm{Prop}$$

ainsi que (lem. 3.2.10, 3.2.4 et 3.2.17)

$$\vdash^{2_e}_{[]} \sigma(y_i) : \mathrm{Prop} \qquad \vdash^{2_e}_{[]} \lambda z^{\sigma(y_i)}.z : \sigma(y_i) \to \sigma(y_i)$$

et on peut alors appliquer l'hypothèse de récurrence pour obtenir un terme a' tel que

$$\vdash^{2_e}_{[]} a' : A[\vec{z}, \vec{y} \leftarrow \sigma(\vec{D}), \sigma(\vec{y})\,] : \mathrm{Prop}$$

c'est-à-dire (lem. 1.2.5)

$$\vdash^{2_e}_{[]} a' : \sigma(A[\vec{z} \leftarrow \vec{D}\,]) : \mathrm{Prop}$$

et comme $\Gamma\, \mathsf{wf}^{2_e}_{\sigma}$ (lem. 3.2.4), on en déduit (lem. 3.2.21)

$$\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,]\, \mathsf{wf}^{2_e}_{\sigma::(x \mapsto a')}$$

Ainsi par (e-var) on a

$$\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,] \vdash^{2_e}_{\sigma::(x \mapsto a')} x : A[\vec{z} \leftarrow \vec{D}\,]$$

et donc aussi (lem. 3.2.5, 3.2.8 et 3.2.7)

$$\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,] \vdash^{2_e}_{\sigma::(x \mapsto a')} A[\vec{z} \leftarrow \vec{D}\,] : \mathrm{Prop}$$

d'où par hypothèse de récurrence on construit un terme u tel que

$$\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,] \vdash^{2_e}_{\sigma::(x \mapsto a')} u : A[\vec{z} \leftarrow \vec{C}\,] : \mathrm{Prop}$$

Par affaiblissement sur l'hypothèse (lem. 3.2.13) et (e-app) on a

$$\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,] \vdash^{2_e}_{\sigma::(x \mapsto a')} w\ u : B[\vec{z} \leftarrow \vec{C}\,]$$

et de la première prémisse

$$\Gamma, x : A[\vec{z} \leftarrow \vec{C}\,] \vdash^{2_e}_{\sigma::(x \mapsto a)} B[\vec{z} \leftarrow \vec{C}\,] : \mathrm{Prop}$$

alors en supprimant x de l'environnement par renforcement (lem. 3.2.19), puis en ajoutant $x : A[\vec{z} \leftarrow \vec{D}\,]$ par affaiblissement (lem. 3.2.13) on obtient

$$\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,] \vdash^{2_e}_{\sigma::(x \mapsto a')} B[\vec{z} \leftarrow \vec{C}\,] : \mathrm{Prop}$$

d'où par hypothèse de récurrence on a un terme v tel que

$$\Gamma, x : A[\vec{z} \leftarrow \vec{D}\,] \vdash^{2_e}_{\sigma::(x \mapsto a')} v : B[\vec{z} \leftarrow \vec{D}\,] : \mathrm{Prop}$$

puis finalement (lem. 3.2.20)

$$\Gamma \vdash^{2_e}_{\sigma} \lambda x^{A[\vec{z} \leftarrow \vec{D}\,]}.v : A[\vec{z} \leftarrow \vec{D}\,] \to B[\vec{z} \leftarrow \vec{D}\,] : \mathrm{Prop}$$

□

Lemme 3.2.23.
Si $\Gamma \vdash^{2_e}_{\sigma} C : \mathrm{Prop}$, $\Gamma \vdash^{2_e}_{\sigma} D : \mathrm{Prop}$ *avec* C *et* D *clos, alors il existe deux termes* f *et* g *tels que*

$$\Gamma \vdash^{2_e}_{\sigma} f : C \to D : \mathrm{Prop} \quad \Gamma \vdash^{2_e}_{\sigma} g : D \to C : \mathrm{Prop}$$

Preuve
Comme C et D clos, alors par renforcement (lem. 3.2.19)

$$\vdash^{2_e}_{[]} C : \mathrm{Prop} \quad \vdash^{2_e}_{[]} D : \mathrm{Prop}$$

et donc il existe des termes u et v (lem. 3.2.12)

$$\vdash^{2_e}_{[]} u : C : \mathrm{Prop} \quad \vdash^{2_e}_{[]} v : D : \mathrm{Prop}$$

et par (e-env$_2$)

$$z : C \, \mathsf{wf}^{2_e}_{(z \mapsto u)} \quad z : D \, \mathsf{wf}^{2_e}_{(z \mapsto v)}$$

puis par affaiblissement (lem. 3.2.13)

$$z : C \vdash^{2_e}_{(z \mapsto u)} v : D : \mathrm{Prop} \quad z : D \vdash^{2_e}_{(z \mapsto v)} u : C : \mathrm{Prop}$$

et par (e-abs) et (e-prod) (lem. 3.2.20)

$$\vdash^{2_e}_{[]} \lambda z^C.v : C \to D : \mathrm{Prop} \quad \vdash^{2_e}_{[]} \lambda z^D.u : D \to C : \mathrm{Prop}$$

puis finalement en affaiblissant (lem. 3.2.13 et 3.2.4)

$$\Gamma \vdash^{2_e}_{\sigma} \lambda z^C.v : C \to D : \mathrm{Prop} \quad \Gamma \vdash^{2_e}_{\sigma} \lambda z^D.u : D \to C : \mathrm{Prop}$$

□

Lemme 3.2.24 (échange des motivations). *Si* $\Gamma \vdash^{2_e}_{\sigma} w : C$ *et* $\Gamma \, \mathsf{wf}^{2_e}_{\sigma'}$, *alors* $\Gamma \vdash^{2_e}_{\sigma'} w : C$.

Preuve par récurrence structurelle sur la dérivation de $\Gamma \vdash^{2_e}_{\sigma} w : C$.

(e-abs) $$\frac{\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_e}_{\sigma} \lambda x^A.u : \forall x^A.B}$$

Comme $\Gamma \vdash^{2_e}_{\sigma} A : \kappa$ est une sous-dérivation stricte (lem. 3.2.4 et 3.2.5), alors par hypothèse de récurrence $\Gamma \vdash^{2_e}_{\sigma'} A : \kappa$, et donc on a a' (lem. 3.2.12 et (e-env$_2$)) tel que

$$\Gamma, x : A \, \mathsf{wf}^{2_e}_{\sigma'::(x \mapsto a')}$$

Il suffit alors d'appliquer l'hypothèse de récurrence sur les deux prémisses puis la règle (e-abs) pour obtenir le résultat.

(e-prod) $$\frac{\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} B : \mathrm{Prop} \quad \vdash^{2_e}_{[]} t : \sigma(\forall x^A.B)}{\Gamma \vdash^{2_e}_{\sigma} \forall x^A.B : \mathrm{Prop}}$$

De même que précédemment, on a $\Gamma, x : A \, \mathsf{wf}^{2_e}_{\sigma'::(x \mapsto a')}$ d'où par hypothèse de récurrence

$$\Gamma, x : A \vdash^{2_e}_{\sigma'::(x \mapsto a')} B : \mathrm{Prop}$$

On peut écrire la seconde prémisse ainsi

$$\vdash^{2_e}_{[]} t : (\forall x^A.B)[y_1, \ldots, y_m \leftarrow \sigma(y_1), \ldots, \sigma(y_m)\,]$$

avec les $(y_i)_{1 \leq i \leq m}$ étant les variables libres de $\forall x^A.B$. Comme de plus $(y_i : \mathrm{Prop}) \in \Gamma$ (lem. 3.2.10), alors aussi (prop. 3.2.11)

$$\vdash^{2_e}_{[]} \sigma(y_i) : \mathrm{Prop} \quad \vdash^{2_e}_{[]} \sigma'(y_i) : \mathrm{Prop}$$

Puisque les $(\sigma(y_i))_{1 \leq i \leq m}$ et les $(\sigma'(y_i))_{1 \leq i \leq m}$ sont clos (lem. 3.2.2), alors (lem. 3.2.23) on a des termes $(f_i)_{1 \leq i \leq m}$ et $(g_i)_{1 \leq i \leq m}$ tels que

$$\vdash^{2_e}_{[]} f_i : \sigma'(y_i) \to \sigma(y_i) \quad \vdash^{2_e}_{[]} g_i : \sigma(y_i) \to \sigma'(y_i)$$

d'où par remplacement d'équivalents (thm. 3.2.22) on a un terme t'

$$\vdash^{2_e}_{[]} t' : (\forall x^A.B)[y_1, \ldots, y_m \leftarrow \sigma'(y_1), \ldots, \sigma'(y_m)\,]$$

c'est-à-dire

$$\vdash^{2_e}_{[]} t' : \sigma'(\forall x^A.B) : \mathrm{Prop}$$

On peut alors conclure en appliquant (e-prod). □

Lemme 3.2.25 (de substitution).

(i) Si $\Gamma, y : C, \Gamma' \,\mathsf{wf}^{2_e}_{\sigma::(y \mapsto c)::\sigma'}$ *et* $\Gamma \vdash^{2_e}_{\sigma} w : C$*, alors il existe une substitution* ρ *telle qu'on ait* $\Gamma, \Gamma'[y \leftarrow w\,] \,\mathsf{wf}^{2_e}_{\sigma::\rho}$*;*

(ii) Si $\Gamma, y : C, \Gamma' \vdash^{2_e}_{\sigma::(y \mapsto c)::\sigma'} d : D$ *et* $\Gamma \vdash^{2_e}_{\sigma} w : C$*, alors il existe une substitution* ρ *telle que* $\Gamma, \Gamma'[y \leftarrow w\,] \vdash^{2_e}_{\sigma::\rho} d[y \leftarrow w\,] : D[y \leftarrow w\,]$.

Preuve par récurrence structurelle sur la première dérivation (similaire à 3.1.8).

(e-env$_2$) immédiat en utilisant l'hypothèse de récurrence sur la première prémisse puis (e-env$_2$) (avec lem. 3.2.12).

(e-var) immédiat en utilisant l'hypothèse de récurrence sur la prémisse, et en affaiblissant (lem. 3.2.13) si la variable extraite de l'environnement est y.

(e-abs) $$\frac{\Gamma, y : C, \Gamma', x : A \vdash^{2_e}_{\sigma::(y \mapsto c)::\sigma'::(x \mapsto a)} u : B : \mathrm{Prop}}{\Gamma, y : C, \Gamma' \vdash^{2_e}_{\sigma::(y \mapsto c)::\sigma'} \lambda x^A.u : \forall x^A.B}$$

Par hypothèse de récurrence sur la première prémisse on a

$$\Gamma, \Gamma'[y \leftarrow w\,], x : A[y \leftarrow w\,] \vdash^{2_e}_{\sigma::\rho'::(x \mapsto a')} u[y \leftarrow w\,] : B[y \leftarrow w\,]$$

et sur la deuxième prémisse

$$\Gamma, \Gamma'[y \leftarrow w\,], x : A[y \leftarrow w\,] \vdash^{2_e}_{\sigma::\rho''::(x \mapsto a'')} B[y \leftarrow w\,] : \mathrm{Prop}$$

d'où par échange des motivations (lem. 3.2.24 et 3.2.4)

$$\Gamma, \Gamma'[y \leftarrow w\,], x : A[y \leftarrow w\,] \vdash^{2_e}_{\sigma::\rho'::(x \mapsto a')} B[y \leftarrow w\,] : \mathrm{Prop}$$

et finalement on obtient le résultat par la règle (e-abs) avec $\rho := \rho'$.

(e-app) De la même façon que précédemment, puisque l'hypothèse de récurrence appliquée aux deux prémisses donnent deux substitutions ρ' et ρ'' potentiellement différentes, on en choisit une et on déduit le résultat par la règle (e-app).

$$\textbf{(e-prod)}\ \frac{\Gamma, y : C, \Gamma', x : A \vdash^{2_e}_{\sigma::(y \mapsto c)::\sigma'::(x \mapsto a)} B : \mathrm{Prop} \qquad \vdash^{2_e}_{[]} t : \sigma::(y \mapsto c)::\sigma'(\forall x^A.B)}{\Gamma, y : C, \Gamma' \vdash^{2_e}_{\sigma::(y \mapsto c)::\sigma'} \forall x^A.B : \mathrm{Prop}}$$

Déjà, par hypothèse de récurrence, on a une substitution ρ' et un terme a' tels que

$$\Gamma, \Gamma'[y \leftarrow w], x : A[y \leftarrow w] \vdash^{2_e}_{\sigma::\rho'::(x \mapsto a')} B[y \leftarrow w] : \mathrm{Prop}$$

D'autre part en transmettant la motivation en conclusion (lem. 3.2.17)

$$\vdash^{2_e}_{[]} t : \sigma::(y \mapsto c)::\sigma'(\forall x^A.B) : \mathrm{Prop} \qquad (*)$$

Pour toute variable libre z de $\forall x^A.B$ on a $(z : \mathrm{Prop}) \in \Gamma, y : C, \Gamma'$ (lem. 3.2.10) et alors :

- $z \not\equiv y$: le critère de Poincaré (prop. 3.2.11) sur les environnements bien formés précédents (lem. 3.2.4) donne

$$\vdash^{2_e}_{[]} \sigma::(y \mapsto c)::\sigma'(z) : \mathrm{Prop} \qquad \vdash^{2_e}_{[]} \sigma::\rho'(z) : \mathrm{Prop}$$

- $z \equiv y$: le critère de Poincaré (prop. 3.2.11), et la transmission de la motivation en conclusion (lem. 3.2.17) donnent

$$\vdash^{2_e}_{[]} \sigma::(y \mapsto c)::\sigma'(z) : \mathrm{Prop} \qquad \vdash^{2_e}_{[]} \sigma(w) : \mathrm{Prop}$$

Tout ces types étant équivalents car clos (lem. 3.2.23), on peut les échanger (thm. 3.2.22) dans $(*)$ pour obtenir un terme t' tel que

$$\vdash^{2_e}_{[]} t' : \sigma::(y \mapsto \sigma(w))::\rho'(\forall x^A.B) : \mathrm{Prop}$$

c'est-à-dire

$$\vdash^{2_e}_{[]} t' : \sigma::\rho'((\forall x^A.B)[y \leftarrow w]) : \mathrm{Prop}$$

ce qui permet de conclure par (e-prod). □

Proposition 3.2.26 (stabilité par réduction). *Si* $\Gamma \vdash^{2_e}_{\sigma} t : C$ *et* $t \rightsquigarrow_\beta t'$, *alors* $\Gamma \vdash^{2_e}_{\sigma} t' : C$.

Preuve par récurrence structurelle sur la dérivation puis par cas sur la définition de $\rightsquigarrow_\beta$ (similaire au thm. 3.1.16).

On peut passer par λ^2 (thm. 3.2.1) pour éliminer certains cas (termes irréductibles, dérivations impossibles, etc.) de la même façon que dans la preuve pour λ^2 : il suffira alors d'utiliser le lemme de substitution (lem. 3.2.25) pour le cas où $t \equiv (\lambda x^A.w)\ v \rightsquigarrow_\beta w[x \leftarrow w] \equiv t'$, et l'hypothèse de récurrence pour le reste. □

Lemme 3.2.27 (correction des types, voir lem. 1.4.9).
Si $\Gamma \vdash^{2_e}_{\sigma} w : C$ *alors* $C \equiv \mathrm{Type}$ *ou bien il existe* κ *tel que* $\Gamma \vdash^{2_e}_{\sigma} C : \kappa$.

Preuve par récurrence structurelle sur la dérivation (similaire à lem. 1.4.9).

(e-abs) Le lemme 3.2.20 donne immédiatement le résultat. □

Proposition 3.2.28 ($\boldsymbol{\lambda_e^2}$ est un pseudo sous-système pédagogique de CC).

λ_e^2 vérifie les propriétés suivantes :
(i) λ_e^2 est un sous-système de CC ;
(ii) Si $\Gamma \vdash_{\sigma}^{2_e} t : C$ et $t \rightsquigarrow_\beta t'$, alors $\Gamma \vdash_{\sigma}^{2_e} t' : C$.
*(iii) $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{2_e}_{(x_1 \mapsto t_1)::\ldots::(x_n \mapsto t_n)}$ **si et seulement si** $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{\mathsf{c}}$ et il existe des termes $t_1, \ldots, t_n$ tels que*

$$\vdash_{[]}^{2_e} t_1 : A_1 \qquad \vdash_{[]}^{2_e} t_2 : A_2[x_1 \leftarrow t_1] \quad \ldots \quad \vdash_{[]}^{2_e} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$$

Preuve

(i) λ_e^2 est un sous-système de λ^2 (thm. 3.2.1) lui-même sous-système de CC (thm. 3.1.1).

(ii) C'est exactement la proposition 3.2.26.

(iii) $\boxed{\Rightarrow}$ C'est exactement la proposition 3.2.11.
$\boxed{\Leftarrow}$ De

$$\vdash_{[]}^{2_e} t_i : A_i[x_1, \ldots, x_{i-1} \leftarrow t_1, \ldots, t_{i-1}]$$

et comme $A_i \not\equiv \text{Type}$ puisque $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{\mathsf{c}}$ (lem. 1.4.2) ainsi que $t_i \not\equiv \text{Type}$ (lem 3.2.3), on déduit que

$$\vdash_{[]}^{2_e} A_i[x_1, \ldots, x_{i-1} \leftarrow t_1, \ldots, t_{i-1}] : \kappa_i$$

et on peut alors appliquer la proposition 3.2.15 pour obtenir le résultat. □

Remarque

Il s'agit en fait d'une définition adaptée d'un sous-système pédagogique. En toute rigueur λ_e^2 n'est pas un sous-système de CC : il y a deux constantes supplémentaires o et $\top$, et tous les jugements sont indicés par une motivation explicite.

3.3 Avec motivations totales – λ_t^2

3.3.1 Définition du système

Termes bruts

◊ On ajoute aux termes bruts de λ^2 les constantes suivantes : o et $\top$.

Motivation d'un environnement

◊ Une substitution σ *motive* un environnement $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$, noté $\sigma \operatorname{mot} \Gamma$, si :

$$\forall i \quad \vdash^{2_t} \sigma(x_i) : \sigma(A_i)$$

◊ Selon le contexte, $\sigma \operatorname{mot} \Gamma$ dénote les dérivations précédentes, ou le fait que l'environnement Γ est *motivable* par σ.

Motivation d'un type par rapport à un environnement

◊ Une substitution σ *motive un type* C *sous un environnement* Γ, noté $\sigma \operatorname{mot}_\Gamma C$ si :

(i) $\sigma \operatorname{mot} \Gamma$;

(ii) et il existe un terme t tel que $\vdash^{2_t} t : \sigma(C)$.

◊ Selon le contexte, $\sigma \operatorname{mot}_\Gamma C$ dénote les dérivations précédentes, ou le fait que C est motivable par σ dans l'environnement Γ.

Remarques

Ces définitions de motivations dépendent du système exposé ci-après, ce qui peut sembler problématique.

Pour résoudre l'apparente circularité, on peut découpler chacune des définitions en (a) une abréviation commode à la définition du système, (b) une définition effective une fois les règles d'inférence du système définies.

Règles d'inférence de λ_t^2

◊ On définit la relation $\vdash^{2_t}$ entre un environnement et deux termes bruts, ainsi que la relation wf^{2_t} sur les environnements par les règles suivantes :

$$\frac{}{[\,]\,\mathsf{wf}^{2_t}}\ (\text{t-env}_1) \qquad \frac{\Gamma \vdash^{2_t} A : \kappa \quad x \notin \operatorname{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^{2_t}}\ (\text{t-env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^{2_t}}{\Gamma \vdash^{2_t} o : \top : \mathrm{Prop} : \mathrm{Type}}\ (\text{t-ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^{2_t}}{\Gamma, x : A, \Gamma' \vdash^{2_t} x : A}\ (\text{t-var})$$

$$\frac{\Gamma, x : A \vdash^{2_t} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_t} \lambda x^A.u : \forall x^A.B}\ (\text{t-abs}) \qquad \frac{\Gamma \vdash^{2_t} u : \forall x^A.B \quad \Gamma \vdash^{2_t} v : A}{\Gamma \vdash^{2_t} u\ v : B[x \leftarrow v\,]}\ (\text{t-app})$$

$$\frac{\Gamma, x : A \vdash^{2_t} B : \mathrm{Prop} \quad \sigma \operatorname{mot}_\Gamma \forall x^A.B}{\Gamma \vdash^{2_t} \forall x^A.B : \mathrm{Prop}}\ (\text{t-prod})$$

3.3.2 Résultats

Théorème 3.3.1 ($\boldsymbol{\lambda^2_t}$ est un sous-système de $\boldsymbol{\lambda^2}$).

(i) si $\Gamma\,\mathsf{wf}^{2_\mathrm{t}}$*, alors* $\Gamma\,\mathsf{wf}^{2}$ *;*

(ii) si $\Gamma \vdash^{2_\mathrm{t}} w : C$*, alors* $\Gamma \vdash^{2} w : C$*.*

Preuve immédiate par récurrence structurelle sur la dérivation : il suffit d'identifier la constante o avec $\lambda A^{\mathrm{Prop}}.\lambda x^A.x$ et $\top$ avec $\forall A^{\mathrm{Prop}}.A \to A$. □

Lemme 3.3.2 (voir lem. 3.1.2).

(i) Si $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^{2_\mathrm{t}}$*, alors pour tout* i*,* $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$ *et* $x_i \not\equiv x_j$ *pour tout* $i \neq j$ *;*

(ii) Si $x_1 : A_1, \ldots, x_n : A_n \vdash^{2_\mathrm{t}} w : C$*, alors pour tout* i*,* $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$ *et* $x_i \not\equiv x_j$ *pour tout* $i \neq j$ *et* $\mathcal{V}(w, C) \subseteq \{x_1, \ldots, x_n\}$*.*

Lemme 3.3.3 (validité des environnements, voir lem. 3.1.4).

(i) si $\Gamma\,\mathsf{wf}^{2_\mathrm{t}}$*, alors pour tout environnement* $\Gamma' \preccurlyeq \Gamma$*,* $\Gamma'\,\mathsf{wf}^{2_\mathrm{t}}$ *en est une sous-dérivation ;*

(ii) si $\Gamma \vdash^{2_\mathrm{t}} w : C$*, alors pour tout environnement* $\Gamma' \preccurlyeq \Gamma$*,* $\Gamma'\,\mathsf{wf}^{2_\mathrm{t}}$ *en est une sous-dérivation stricte.*

Lemme 3.3.4 (validité des types des environnements, voir lem. 3.1.5).

Si $x_1 : A_1, \ldots, x_n : A_n\,\mathsf{wf}^{2_\mathrm{t}}$*, alors pour tout* i *il existe* κ *tel que* $x_1 : A_1, \ldots, x_i : A_i \vdash^{2_\mathrm{t}} A_{i+1} : \kappa$ *en est une sous-dérivation stricte.*

Lemme 3.3.5 (voir lem. 3.2.8).

(i) Si $\Gamma \vdash^{2_\mathrm{t}} C : \mathrm{Type}$ *alors* $C \equiv \mathrm{Prop}$ *et la dernière règle de la dérivation est (t-ax) ;*

(ii) Si $\Gamma \vdash^{2_\mathrm{t}} C : \mathrm{Prop}$ *alors la dernière règle de la dérivation est (t-ax), (t-var) ou (t-prod).*

Théorème 3.3.6 ($\boldsymbol{\lambda^2_e}$ est un sous-système de $\boldsymbol{\lambda^2_t}$). *Pour toute substitution* σ *:*

(i) si $\Gamma\,\mathsf{wf}^{2_\mathrm{e}}_\sigma$*, alors* $\Gamma\,\mathsf{wf}^{2_\mathrm{t}}$ *;*

(ii) si $\Gamma \vdash^{2_\mathrm{e}}_\sigma w : C$*, alors* $\Gamma \vdash^{2_\mathrm{t}} w : C$*.*

Preuve par récurrence structurelle sur la dérivation.

Tous les cas sauf (e-prod) sont immédiats : dès lors qu'on *oublie* la motivation explicite, les règles sont les mêmes ou plus contraintes.

$$\textbf{(e-prod)}\quad \frac{\Gamma, x : A \vdash^{2_\mathrm{e}}_{\sigma::(x \mapsto a)} B : \mathrm{Prop} \qquad \vdash^{2_\mathrm{e}}_{[]} t : \sigma(\forall x^A.B)}{\Gamma \vdash^{2_\mathrm{e}}_\sigma \forall x^A.B : \mathrm{Prop}}$$

De la première prémisse on a $\Gamma\,\mathsf{wf}^{2_\mathrm{e}}_\sigma$ (lem. 3.2.4), et par le critère de Poincaré (prop. 3.2.11)

$$\forall i \quad \vdash^{2_\mathrm{e}}_{[]} \sigma(x_i) : \sigma(A_i)$$

sont des sous-dérivations strictes, sur lesquelles on peut appliquer l'hypothèse de récurrence pour obtenir

$$\forall i \quad \vdash^{2_\mathrm{t}} \sigma(x_i) : \sigma(A_i)$$

c'est-à-dire σ mot Γ. De plus, l'hypothèse de récurrence appliquée à la seconde prémisse donne

$$\vdash^{2_\mathrm{t}} t : \sigma(\forall x^A.B)$$

et on a alors $\sigma\,\mathrm{mot}_\Gamma\,\forall x^A.B$. Par hypothèse de récurrence sur la première prémisse et la règle (t-prod) on obtient bien le résultat. □

Théorème 3.3.7 (λ_t^2 est un sous-système de λ_e^2).

(i) si $\Gamma\,\mathsf{wf}^{2_t}$, alors il existe une substitution σ telle que $\Gamma\,\mathsf{wf}^{2_e}_{\sigma}$;

(ii) si $\Gamma \vdash^{2_t} w : C$, alors il existe une substitution σ telle que $\Gamma \vdash^{2_e}_{\sigma} w : C$.

Preuve par récurrence structurelle sur la dérivation.

(t-env$_1$) Trivial : $\sigma := [\,]$ convient.

(t-env$_2$) $$\frac{\Gamma \vdash^{2_t} A : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^{2_t}}$$

Par hypothèse de récurrence, on a une substitution σ' telle que $\Gamma \vdash^{2_e}_{\sigma'} A : \kappa$, et donc (lem. 3.2.12) on a un terme a tel que $\vdash^{2_e}_{[\,]} a : \sigma'(A)$. D'où par (e-env$_2$) on obtient bien le résultat avec $\sigma := \sigma' ::(x \mapsto a)$.

(t-ax) Immédiat par hypothèse de récurrence et (e-ax).

(t-var) Immédiat par hypothèse de récurrence et (e-var).

(t-abs) $$\frac{\Gamma, x : A \vdash^{2_t} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_t} \lambda x^A.u : \forall x^A.B}$$

Par hypothèse de récurrence, on a σ_1, σ_2, a_1 et a_2 tels que

$$\Gamma, x : A \vdash^{2_e}_{\sigma_1::(x \mapsto a_1)} u : B \qquad \text{et} \qquad \Gamma, x : A \vdash^{2_e}_{\sigma_2::(x \mapsto a_2)} B : \mathrm{Prop}$$

d'où, en échangeant les motivations (lem. 3.2.4 et 3.2.24), on a aussi

$$\Gamma, x : A \vdash^{2_e}_{\sigma_1::(x \mapsto a_1)} B : \mathrm{Prop}$$

et finalement le résultat par (e-abs).

(t-app) On effectue de même que pour (t-abs).

(t-prod) $$\frac{\Gamma, x : A \vdash^{2_t} B : \mathrm{Prop} \quad \sigma\,\mathrm{mot}_\Gamma\,\forall x^A.B}{\Gamma \vdash^{2_t} \forall x^A.B : \mathrm{Prop}}$$

Dans la suite, (HR) désignera l'hypothèse de récurrence, applicable à toute sous-dérivation stricte de $\Gamma \vdash^{2_t} \forall x^A.B : \mathrm{Prop}$.

Soit $\Gamma \equiv y_1 : D_1, \ldots, y_n : D_n$. On va d'abord montrer par récurrence sur i que

$$\forall i \quad y_1 : D_1, \ldots, y_i : D_i\,\mathsf{wf}^{2_e}_{\sigma_{\leq i}}$$

- $i = 0$: par (e-env$_1$) on a bien $[\,]\,\mathsf{wf}^{2_e}_{[\,]}$.
- Supposons
$$y_1 : D_1, \ldots, y_i : D_i\,\mathsf{wf}^{2_e}_{\sigma_{\leq i}} \tag{HRi}$$

  Par définition de $\sigma\,\mathrm{mot}_\Gamma\,\forall x^A.B$ on a $\vdash^{2_t} \sigma(y_{i+1}) : \sigma(D_{i+1})$ comme sous-dérivation, sur laquelle on peut appliquer l'hypothèse de récurrence (HR) pour obtenir

$$\vdash^{2_e}_{[\,]} \sigma(y_{i+1}) : \sigma(D_{i+1})$$

Comme $y_1 : D_1, \ldots, y_i : D_i \vdash^{2_t} D_{i+1} : \kappa$ (lem. 3.3.3 et lem. 3.3.4) est une sous-dérivation de la première prémisse, par hypothèse de récurrence (HR) il existe une substitution ρ telle que $y_1 : D_1, \ldots, y_i : D_i \vdash^{2_e}_{\rho} D_{i+1} : \kappa$, et donc par échange de motivations (lem. 3.2.24) avec (HRi) $y_1 : D_1, \ldots, y_i : D_i \vdash^{2_e}_{\sigma_{\leq i}} D_{i+1} : \kappa$, ce qui, en transmettant la motivation en conclusion (lem. 3.2.17) donne

$$\vdash^{2_e}_{[]} \sigma_{\leq i}(D_{i+1}) : \kappa$$

Finalement de

$$y_1 : D_1, \ldots, y_i : D_i \,\mathsf{wf}^{2_e}_{\sigma_{\leq i}} \tag{HRi}$$

et

$$\vdash^{2_e}_{[]} \sigma(y_{i+1}) : \sigma_{\leq i}(D_{i+1}) : \kappa$$

on déduit (lem. 3.2.21) que

$$y_1 : D_1, \ldots, y_i : D_i, y_{i+1} : D_{i+1} \,\mathsf{wf}^{2_e}_{\sigma_{\leq i+1}}$$

ce qui termine cette sous-preuve par récurrence sur i.

Quand $i = n$, on a $\Gamma \,\mathsf{wf}^{2_e}_{\sigma}$.
L'hypothèse de récurrence (HR) sur la première prémisse donne ρ et a' tels que

$$\Gamma, x : A \vdash^{2_e}_{\rho::(x \mapsto a')} B : \mathrm{Prop}$$

On a alors $\Gamma \vdash^{2_e}_{\rho} A : \kappa$ (lem. 3.2.4 et 3.2.5), donc $\Gamma \vdash^{2_e}_{\sigma} A : \kappa$ par échange de motivations (lem. 3.2.24), d'où il existe a tel que $\vdash^{2_e}_{[]} a : \sigma(A)$ (lem. 3.2.12), et par (e-env$_2$)

$$\Gamma, x : A \,\mathsf{wf}^{2_e}_{\sigma::(x \mapsto a)}$$

Par échange de motivations (lem. 3.2.24) on a alors

$$\Gamma, x : A \vdash^{2_e}_{\sigma::(x \mapsto a)} B : \mathrm{Prop}$$

Or par définition de $\sigma \,\mathrm{mot}_\Gamma \,\forall x^A.B$, il existe un terme t tel que $\vdash^{2_t} t : \sigma(\forall x^A.B)$ en est une sous-dérivation, sur laquelle on peut appliquer l'hypothèse de récurrence (HR) pour obtenir

$$\vdash^{2_e}_{[]} t : \sigma(\forall x^A.B)$$

Finalement la règle (e-prod) nous donne bien le résultat. □

Proposition 3.3.8 ($\boldsymbol{\lambda^2_t}$ est un pseudo sous-système pédagogique de CC).
λ^2_t vérifie les propriétés suivantes :
(i) λ^2_t est un sous-système de CC ;
(ii) Si $\Gamma \vdash^{2_t} t : C$ et $t \rightsquigarrow_\beta t'$, alors $\Gamma \vdash^{2_t} t' : C$.
*(iii) $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{2_t}$ **si et seulement si** $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{\mathsf{c}}$ et il existe des termes $t_1, \ldots, t_n$ tels que*

$$\vdash^{2_t} t_1 : A_1 \qquad \vdash^{2_t} t_2 : A_2[x_1 \leftarrow t_1] \quad \ldots \quad \vdash^{2_t} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$$

Preuve

(i) λ_t^2 est un sous-système de λ^2 (thm. 3.3.1) lui-même sous-système de CC (thm. 3.1.1).

(ii) De

$$\Gamma \vdash^{2_t} t : C$$

on a (thm. 3.3.7) une substitution σ telle que

$$\Gamma \vdash^{2_e}_{\sigma} t : C$$

et comme $t \rightsquigarrow_\beta t'$, alors (prop. 3.2.28)

$$\Gamma \vdash^{2_e}_{\sigma} t' : C$$

d'où (thm. 3.3.6)

$$\Gamma \vdash^{2_t} t' : C$$

(iii) $\boxed{\Rightarrow}$ De

$$x_1 : A_1, \ldots, x_n : A_n \ \mathsf{wf}^{2_t}$$

on construit (thm. 3.3.7) une substitution $\sigma := [(x_1 \mapsto t_1)::\ldots::(x_n \mapsto t_n)]$ telle que

$$x_1 : A_1, \ldots, x_n : A_n \ \mathsf{wf}^{2_e}_{(x_1 \mapsto t_1)::\ldots::(x_n \mapsto t_n)}$$

d'où l'on déduit (prop. 3.2.28)

$$\forall i \quad \vdash^{2_e}_{[]} t_{i+1} : A_{i+1}[x_1, \ldots, x_i \leftarrow t_1, \ldots, t_i\,]$$

et finalement (thm. 3.3.6)

$$\forall i \quad \vdash^{2_t} t_{i+1} : A_{i+1}[x_1, \ldots, x_i \leftarrow t_1, \ldots, t_i\,]$$

$\boxed{\Leftarrow}$ De

$$\forall i \quad \vdash^{2_t} t_{i+1} : A_{i+1}[x_1, \ldots, x_i \leftarrow t_1, \ldots, t_i\,]$$

on a (thm. 3.3.7 et lem. 3.2.2)

$$\forall i \quad \vdash^{2_e}_{[]} t_{i+1} : A_{i+1}[x_1, \ldots, x_i \leftarrow t_1, \ldots, t_i\,]$$

et comme aussi $x_1 : A_1, \ldots, x_n : A_n \ \mathsf{wf}^{\mathsf{c}}$, alors (prop. 3.2.28)

$$x_1 : A_1, \ldots, x_n : A_n \ \mathsf{wf}^{2_e}_{(x_1 \mapsto t_1)::\ldots::(x_n \mapsto t_n)}$$

et donc (thm.3.3.6)

$$x_1 : A_1, \ldots, x_n : A_n \ \mathsf{wf}^{2_t}$$

□

Remarque

Comme pour λ_e^2, il ne s'agit pas de la vrai définition d'un sous-système pédagogique de CC à cause des constantes o et $\top$ ajoutées au calcul.

3.4 Avec motivations partielles – λ_p^2

3.4.1 Définition du système

Application d'une motivation partielle à un environnement

◊ L'application de la substitution σ à l'environnement Γ, dont le résultat est noté $\sigma(\Gamma)$, est définie par :

$$\sigma([\,]) := [\,]$$
$$\sigma(\Gamma, x : A) := \begin{cases} \sigma(\Gamma) & \text{si } x \in \mathrm{dom}(\sigma) \\ \sigma(\Gamma), x : \sigma(A) & \text{sinon} \end{cases}$$

Motivation partielle d'un environnement

◊ Une substitution σ *motive partiellement* l'environnement $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$, noté $\sigma \ \widetilde{\mathrm{mot}}\ \Gamma$, si :

$$\forall i \quad x_i \in \mathrm{dom}(\sigma) \Rightarrow \sigma(\Gamma_{<i}) \vdash^{2_{\mathrm{P}}} \sigma(x_i) : \sigma(A_i)$$

Selon le contexte, $\sigma \ \widetilde{\mathrm{mot}}\ \Gamma$ dénote les dérivations précédentes, ou le fait que l'environnement Γ est *partiellement motivable* par σ.

Motivation partielle d'un type par rapport à un environnement

◊ Une substitution σ *motive partiellement un type* C sous un environnement Γ, noté $\sigma \ \widetilde{\mathrm{mot}}_\Gamma\ C$ si :

(i) $\sigma \ \widetilde{\mathrm{mot}}\ \Gamma$;

(ii) et il existe un terme t tel que $\sigma(\Gamma) \vdash^{2_{\mathrm{P}}} t : \sigma(C)$.

◊ Selon le contexte, $\sigma \ \widetilde{\mathrm{mot}}_\Gamma\ C$ dénote les dérivations précédentes, ou le fait que C est partiellement motivable par σ dans l'environnement Γ.

Remarques

Encore une fois, les deux définitions suivantes dépendent des règles du système exposé ci-après. Les mêmes remarques que précédemment (pour λ_t^2) valent ici aussi.

Quand $\mathrm{dom}(\Gamma) \subseteq \mathrm{dom}(\sigma)$ on retombe sur les définitions précédentes des motivations totales de λ_t^2.

Pour tout environnement Γ, on a toujours $[\,] \ \widetilde{\mathrm{mot}}\ \Gamma$. En revanche, pour un type C, on n'a pas toujours $[\,] \ \widetilde{\mathrm{mot}}_\Gamma\ C$.

Règles d'inférence de λ_p^2

◊ On définit la relation $\vdash^{2_\mathrm{p}}$ entre un environnement et deux termes bruts, ainsi que la relation $\mathsf{wf}^{2_\mathrm{p}}$ sur les environnements par les règles suivantes :

$$\frac{}{[\,]\,\mathsf{wf}^{2_\mathrm{p}}}\ (\text{p-env}_1) \qquad \frac{\Gamma \vdash^{2_\mathrm{p}} A : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^{2_\mathrm{p}}}\ (\text{p-env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^{2_\mathrm{p}}}{\Gamma \vdash^{2_\mathrm{p}} \mathrm{Prop} : \mathrm{Type}}\ (\text{p-ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^{2_\mathrm{p}}}{\Gamma, x : A, \Gamma' \vdash^{2_\mathrm{p}} x : A}\ (\text{p-var})$$

$$\frac{\Gamma, x : A \vdash^{2_\mathrm{p}} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_\mathrm{p}} \lambda x^A.u : \forall x^A.B}\ (\text{p-abs}) \qquad \frac{\Gamma \vdash^{2_\mathrm{p}} u : \forall x^A.B \quad \Gamma \vdash^{2_\mathrm{p}} v : A}{\Gamma \vdash^{2_\mathrm{p}} u\ v : B[x \leftarrow v\,]}\ (\text{p-app})$$

$$\frac{\Gamma, x : A \vdash^{2_\mathrm{p}} B : \mathrm{Prop} \quad \sigma\ \widetilde{\mathrm{mot}}_\Gamma\ \forall x^A.B}{\Gamma \vdash^{2_\mathrm{p}} \forall x^A.B : \mathrm{Prop}}\ (\text{p-prod})$$

Remarques

Si $\mathrm{dom}(\sigma) = \emptyset$ dans la règle (p-prod), alors $\sigma\ \widetilde{\mathrm{mot}}_\Gamma\ \forall x^A.B$ revient à trouver un terme t tel que $\Gamma \vdash^{2_\mathrm{p}} t : \forall x^A.B$, autrement dit (quasiment) ce qu'il se passait dans CC_r (voir sec. 2.2).

Il s'agit ici d'un vrai sous-système de CC puisqu'aucune constante n'est ajoutée au calcul.

3.4.2 Résultats

Dans cette section, on identifiera les constantes o et $\top$ des systèmes précédents avec leur définition respective, soit $o := \lambda A^{\mathrm{Prop}}.\lambda x^A.x$ et $\top := \forall A^{\mathrm{Prop}}.A \to A$.

Lemme 3.4.1. *Dans λ_p^2, on a les règles dérivées suivantes :*

$$\frac{\Gamma\,\mathsf{wf}^{2_\mathrm{p}}}{\Gamma \vdash^{2_\mathrm{p}} o : \top : \mathrm{Prop} : \mathrm{Type}}$$

Preuve similaire à celle pour CC_r (lem. 2.2.15) en utilisant une substitution vide à chaque utilisation de la règle (p-prod). □

Théorème 3.4.2 (λ_p^2 est un sous-système de λ^2).

(i) si $\Gamma\,\mathsf{wf}^{2_\mathrm{p}}$, alors $\Gamma\,\mathsf{wf}^2$;

(ii) si $\Gamma \vdash^{2_\mathrm{p}} w : C$, alors $\Gamma \vdash^2 w : C$.

Preuve immédiate par récurrence structurelle sur la dérivation. □

Théorème 3.4.3 (λ_t^2 est un sous-système de λ_p^2).

(i) si $\Gamma\,\mathsf{wf}^{2_\mathrm{t}}$, alors $\Gamma\,\mathsf{wf}^{2_\mathrm{p}}$;

(ii) si $\Gamma \vdash^{2_\mathrm{t}} w : C$, alors $\Gamma \vdash^{2_\mathrm{p}} w : C$.

Preuve immédiate par récurrence structurelle sur la dérivation.

Le cas de (t-ax) est le lemme 3.4.1.

Dans le cas de (t-prod), il suffit de remarquer que si $\sigma\,\mathrm{mot}_\Gamma\,\forall x^A.B$, alors $\sigma\ \widetilde{\mathrm{mot}}_\Gamma\ \forall x^A.B$ en appliquant autant de fois que nécessaire l'hypothèse de récurrence sur les dérivations de $\sigma\,\mathrm{mot}_\Gamma\,\forall x^A.B$. □

Théorème 3.4.4 ($\boldsymbol{\lambda^2_p}$ est un sous-système de $\boldsymbol{\lambda^2_t}$).

(i) si $\Gamma\,\mathsf{wf}^{2_\mathrm{p}}$, *alors* $\Gamma\,\mathsf{wf}^{2_\mathrm{t}}$ *;*

(ii) si $\Gamma \vdash^{2_\mathrm{p}} w : C$, *alors* $\Gamma \vdash^{2_\mathrm{t}} w : C$.

Preuve par récurrence structurelle sur la dérivation.

$$\textbf{(p-prod)}\ \frac{\Gamma, x : A \vdash^{2_\mathrm{p}} B : \mathrm{Prop} \qquad \sigma\ \widetilde{\mathrm{mot}}_\Gamma\ \forall x^A.B}{\Gamma \vdash^{2_\mathrm{p}} \forall x^A.B : \mathrm{Prop}} \quad \text{avec } \Gamma \equiv y_1 : C_1, \ldots, y_n : C_n.$$

Par définition de $\sigma\ \widetilde{\mathrm{mot}}_\Gamma\ \forall x^A.B$, on a un terme t tel que

$$\sigma(\Gamma) \vdash^{2_\mathrm{p}} t : \sigma(\forall x^A.B)$$

est une sous-dérivation, et donc par hypothèse de récurrence

$$\sigma(\Gamma) \vdash^{2_\mathrm{t}} t : \sigma(\forall x^A.B)$$

d'où il existe (thm. 3.3.7) une substitution ρ telle que

$$\sigma(\Gamma) \vdash^{2_\mathrm{e}}_{\rho} t : \sigma(\forall x^A.B) \qquad (*)$$

On a alors $\rho \odot \sigma\,\mathrm{mot}_\Gamma\,\forall x^A.B$ puisque :

- quand $y_i \in \mathrm{dom}(\sigma)$, alors par définition de $\sigma\ \widetilde{\mathrm{mot}}_\Gamma\ \forall x^A.B$

  $$\sigma(\Gamma_{<i}) \vdash^{2_\mathrm{p}} \sigma(y_i) : \sigma(C_i)$$

  est une sous-dérivation, et donc par hypothèse de récurrence

  $$\sigma(\Gamma_{<i}) \vdash^{2_\mathrm{t}} \sigma(y_i) : \sigma(C_i)$$

  d'où il existe ρ' (thm. 3.3.7) telle que

  $$\sigma(\Gamma_{<i}) \vdash^{2_\mathrm{e}}_{\rho'} \sigma(y_i) : \sigma(C_i)$$

  et donc par échange de motivations (lem. 3.2.24 et lem. 3.2.4)

  $$\sigma(\Gamma_{<i}) \vdash^{2_\mathrm{e}}_{\rho_{<i}} \sigma(y_i) : \sigma(C_i)$$

  puis en transférant la motivation en conclusion (lem. 3.2.17)

  $$\vdash^{2_\mathrm{e}}_{[]} \rho(\sigma(y_i)) : \rho(\sigma(C_i))$$

  et donc aussi (thm. 3.3.6 et lem. 1.2.5)

  $$\vdash^{2_\mathrm{t}} \rho \odot \sigma(y_i) : \rho \odot \sigma(C_i)$$

- quand $y_i \notin \mathrm{dom}(\sigma)$, alors par définition $y_i \in \mathrm{dom}(\sigma(\Gamma))$, et donc de $(*)$ par le critère de Poincaré (prop. 3.2.11 et lem. 3.2.4)

$$\vdash^{2_e}_{[]} \rho(y_i) : \rho(\sigma(C_i))$$

c'est-à-dire, puisque $y_i \notin \mathrm{dom}(\sigma)$,

$$\vdash^{2_e}_{[]} \rho(\sigma(y_i)) : \rho(\sigma(C_i))$$

d'où (thm. 3.3.6 et lem. 1.2.5)

$$\vdash^{2_t} \rho \odot \sigma(y_i) : \rho \odot \sigma(C_i)$$

- De $(*)$, en transmettant la motivation en conclusion (lem. 3.2.17)

$$\vdash^{2_e}_{[]} \rho(u) : \rho(\sigma(\forall x^A.B))$$

d'où (thm. 3.3.6 et lem. 1.2.5)

$$\vdash^{2_t} \rho(u) : \rho \odot \sigma(\forall x^A.B)$$

Finalement, comme par hypothèse de récurrence sur la prémisse on a

$$\Gamma, x : A \vdash^{2_t} B : \mathrm{Prop}$$

alors la règle (t-prod) nous permet de conclure. □

Proposition 3.4.5 (λ_p^2 est un sous-système pédagogique de CC).

λ_p^2 vérifie les propriétés suivantes :
(i) λ_p^2 est un sous-système de CC ;
(ii) Si $\Gamma \vdash^{2_p} t : C$ et $t \rightsquigarrow_\beta t'$, alors $\Gamma \vdash^{2_p} t' : C$.
*(iii) $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{2_p}$ **si et seulement si** $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^{\mathsf{c}}$ et il existe des termes $t_1, \ldots, t_n$ tels que*

$$\vdash^{2_p} t_1 : A_1 \qquad \vdash^{2_p} t_2 : A_2[x_1 \leftarrow t_1] \quad \ldots \quad \vdash^{2_p} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$$

Preuve

(i) λ_p^2 est un sous-système de λ^2 (thm. 3.4.2) lui-même sous-système de CC (thm. 3.1.1).
(ii) et (iii) Il suffit d'utiliser le fait que les deux systèmes λ_t^2 et λ_p^2 sont équivalents (thm. 3.4.3 et thm. 3.4.4) afin de récupérer les résultats du premier (prop. 3.3.8) dans le second. □

Remarque

Cette fois-ci, λ_p^2 est un vrai sous-système pédagogique de CC (sec 2.1).

3.5 Système F pédagogique – P-Prop^2

On commence par rappeler le système P-Prop^2 proposé par Colson et Michel (2009) : le λ-calcul du second ordre pédagogique. On montre qu'il est contenu dans les systèmes précédents, en particulier λ_t^2.

3.5.1 Définition du système

Types de P-Prop^2

◊ Les *types* de P-Prop^2 sont construits selon les règles suivantes :
- $\top$ est un type ;
- les variables de type $\alpha, \beta, \gamma, \ldots$ sont des types ;
- si A et B sont des types, alors $A \rightarrow B$ est un type ;
- si α est une variable de type et A un type, alors $\forall \alpha.A$ est un type.

◊ L'ensemble des variables libres d'un type A, noté $\mathcal{V}(A)$, est défini de façon usuelle.

Termes de P-Prop^2

◊ Les *termes* de P-Prop^2 sont construits selon les règles suivantes :
- $\boldsymbol{o}$ est un terme ;
- les variables de terme $x, y, z, \ldots$ sont des termes ;
- si x est une variable de terme, A un type et t un terme, alors $\boldsymbol{\lambda} x^A.t$ est un terme ;
- si α est une variable de type et t un terme, alors $\boldsymbol{\Lambda} \alpha.t$ est un terme ;
- si t et u sont des termes, alors $t\ u$ est un terme ;
- si t est un terme et U un type, alors $t\ U$ est un terme.

Substitutions de P-Prop^2

◊ Une *substitution* de P-Prop^2 est une application des variables de type dans les types.

◊ L'application d'une substitution σ à un type A, définie de manière usuelle, est notée $\sigma \cdot A$.

◊ Une substitution partout identique sauf en un nombre fini de points $\alpha_1, \ldots, \alpha_n$, associés aux types $V_1, \ldots, V_n$, est notée $[\alpha_1, \ldots, \alpha_n \leftarrow V_1, \ldots, V_n]$.

Contextes de P-Prop^2

◊ Un *contexte* Δ de P-Prop^2 est un ensemble fini de couples $x : A$ où x est une variable de terme et A un type. De plus, si $x : A$ et $x : B$ sont dans Δ alors $A = B$.

◊ Le contexte $\{x_1 : A_1, \ldots, x_n : A_n\}$ est noté plus simplement $x_1 : A_1, \ldots, x_n : A_n$.

◊ L'ensemble des variables libres d'un contexte $\Delta = x_1 : A_1, \ldots, x_n : A_n$, noté $\mathcal{V}(\Delta)$, est défini de façon usuelle comme l'union des $\mathcal{V}(A_i)$.

Les définitions de motivation suivantes font référence au système défini par la suite.

Motivations de P-Prop^2

◊ Une substitution σ de P-Prop^2 *motive* un type A, noté $\vdash^{\mathsf{pf}} \sigma \cdot A$, s'il existe un terme t tel que $\vdash^{\mathsf{pf}} t : \sigma \cdot A$.

◊ Par extension, une substitution σ *motive un contexte* $\Delta = x_1 : A_1, \ldots, x_n : A_n$, noté $\vdash^{\mathsf{pf}} \sigma \cdot \Delta$, si pour tout i on a $\vdash^{\mathsf{pf}} \sigma \cdot A_i$.

Règles d'inférence de P-Prop^2

◊ La relation $\vdash^{\mathsf{pf}}$ entre un contexte, un terme et un type de P-Prop^2 est définie par les règles suivantes :

$$\frac{\vdash^{\mathsf{pf}} \sigma \cdot \Delta}{\Delta \vdash^{\mathsf{pf}} \boldsymbol{o} : \top}\ (\text{P-Ax}) \qquad \frac{x : F \in \Delta \quad \vdash^{\mathsf{pf}} \sigma \cdot \Delta}{\Delta \vdash^{\mathsf{pf}} x : F}\ (\text{P-Hyp})$$

$$\frac{\Delta, x : A \vdash^{\mathsf{pf}} u : B}{\Delta \vdash^{\mathsf{pf}} \boldsymbol{\lambda} x^A.u : A \to B}\ (\to_i) \qquad \frac{\Delta \vdash^{\mathsf{pf}} u : A \to B \quad \Delta \vdash^{\mathsf{pf}} v : A}{\Delta \vdash^{\mathsf{pf}} u\ v : B}\ (\to_e)$$

$$\frac{\Delta \vdash^{\mathsf{pf}} u : B \quad \alpha \notin \mathcal{V}(\Delta)}{\Delta \vdash^{\mathsf{pf}} \boldsymbol{\Lambda} \alpha.u : \forall \alpha.B}\ (\forall_i) \qquad \frac{\Delta \vdash^{\mathsf{pf}} u : \forall \alpha.B \quad \vdash^{\mathsf{pf}} \sigma \cdot V}{\Delta \vdash^{\mathsf{pf}} u\ V : [\alpha \leftarrow V] \cdot B}\ (\text{P-}\forall_e)$$

3.5.2 Résultats

Traduction des termes de P-Prop^2 vers ceux de λ_t^2

◊ Soit la traduction $[\![\cdot]\!]$ des types et termes de P-Prop^2 vers les termes bruts de λ_t^2 définie par :

$$\begin{aligned}
[\![\top]\!] &:= \top \\
[\![\alpha]\!] &:= \alpha \text{ où } \alpha \text{ est une variable de type} \\
[\![A \to B]\!] &:= [\![A]\!] \to [\![B]\!] \\
[\![\forall \alpha.A]\!] &:= \forall \alpha^{\text{Prop}}.[\![A]\!] \\
[\![\boldsymbol{o}]\!] &:= o \\
[\![x]\!] &:= x \text{ où } x \text{ est une variable de terme} \\
[\![\boldsymbol{\lambda} x^A.t]\!] &:= \lambda x^{[\![A]\!]}.[\![t]\!] \\
[\![\boldsymbol{\Lambda} \alpha.t]\!] &:= \lambda \alpha^{\text{Prop}}.[\![t]\!] \\
[\![t\ u]\!] &:= [\![t]\!]\ [\![u]\!] \\
[\![t\ U]\!] &:= [\![t]\!]\ [\![U]\!]
\end{aligned}$$

Remarque

On suppose que les variables des termes bruts de λ_t^2 contiennent les variables de type et de terme de P-Prop^2.

Lemme 3.5.1. *Pour tout type A, B et toute variable de type α de P-Prop^2*

$$[\![[\alpha \leftarrow B] \cdot A]\!] \equiv [\![A]\!][\alpha \leftarrow [\![B]\!]\,]$$

Preuve par récurrence structurelle sur le type A. □

Notons quelques résultats simplifiant l'extension de la traduction des contexte de P-Prop^2 vers les environnements de λ_t^2.

Lemme 3.5.2 (échange dans λ_e^2).

(i) Si $\Gamma, y : C, z : D, \Gamma' \,\mathsf{wf}^{2_e}_{\sigma::(y \mapsto c)::(z \mapsto d)::\sigma'}$ *et* $y \notin \mathcal{V}(D)$,
alors $\Gamma, z : D, y : C, \Gamma' \,\mathsf{wf}^{2_e}_{\sigma::(z \mapsto d)::(y \mapsto c)::\sigma'}$ *;*

(ii) Si $\Gamma, y : C, z : D, \Gamma' \vdash^{2_e}_{\sigma::(y \mapsto c)::(z \mapsto d)::\sigma'} w : E$ *et* $y \notin \mathcal{V}(D)$,
alors $\Gamma, z : D, y : C, \Gamma' \vdash^{2_e}_{\sigma::(z \mapsto d)::(y \mapsto c)::\sigma'} w : E$.

Preuve par récurrence structurelle sur la dérivation.

$$\textbf{(e-env}_2\textbf{)}\ \frac{\Gamma, y : C \vdash^{2_e}_{\sigma::(y \mapsto c)} D : \kappa \quad \vdash^{2_e}_{[]} d : \sigma::(y \mapsto c)(D) \quad z \notin \mathrm{dom}(\Gamma, y)}{\Gamma, y : C, z : D \,\mathsf{wf}^{2_e}_{\sigma::(y \mapsto c)::(z \mapsto d)}}$$

Comme $y \notin \mathcal{V}(D)$ par renforcement (lem. 3.2.19) sur la première prémisse on obtient $\Gamma \vdash^{2_e}_{\sigma} D : \kappa$ puis $\vdash^{2_e}_{[]} d : \sigma(D)$ (lem. 3.2.17). Donc par (e-env$_2$) on a $\Gamma, z : D \,\mathsf{wf}^{2_e}_{\sigma::(z \mapsto d)}$.

De la première prémisse, on déduit $\Gamma \vdash^{2_e}_{\sigma} C : \kappa$ (lem. 3.2.5), qu'on affaiblit (lem. 3.2.13) pour obtenir une dérivation de $\Gamma, z : D \vdash^{2_e}_{\sigma::(z \mapsto d)} C : \kappa$.

Comme $\vdash^{2_e}_{[]} c : \sigma(C)$ (prop. 3.2.11) et $z \notin \mathcal{V}(C)$ (lem. 3.2.2), alors aussi $\vdash^{2_e}_{[]} c : \sigma::(z \mapsto d)(C)$, et par (e-env$_2$) on obtient finalement le résultat $\Gamma, z : D, y : C \,\mathsf{wf}^{2_e}_{\sigma::(z \mapsto d)::(y \mapsto c)}$. □

Lemme 3.5.3 (échange dans λ_t^2).

(i) Si $\Gamma, y : C, z : D, \Gamma' \,\mathsf{wf}^{2_t}$ *et* $y \notin \mathcal{V}(D)$, *alors* $\Gamma, z : D, y : C, \Gamma' \,\mathsf{wf}^{2_t}$ *;*

(ii) Si $\Gamma, y : C, z : D, \Gamma' \vdash^{2_t} w : E$ *et* $y \notin \mathcal{V}(D)$, *alors* $\Gamma, z : D, y : C, \Gamma' \vdash^{2_t} w : E$.

Preuve immédiate comme corollaire du lemme 3.5.2 précédent, puisque λ_t^2 et λ_e^2 sont équivalents (thm. 3.3.6 et 3.3.7). □

Lemme 3.5.4. *Si* $\Gamma \vdash^{2_t} w : C$, *alors on peut diviser* Γ *en* Γ_1 *et* Γ_2 *tels que :*
- Γ *est une permutation de* Γ_1, Γ_2 *;*
- $\Gamma_1, \Gamma_2 \vdash^{2_t} w : C$ *;*
- *pour tout* $y : D \in \Gamma_1$, $D \equiv \mathrm{Prop}$ *;*
- *pour tout* $y : D \in \Gamma_2$, $D \not\equiv \mathrm{Prop}$.

Preuve

Posons $\Gamma \equiv x_1 : A_1, \ldots, x_n : A_n$. Comme on a $x_1 : A_1, \ldots, x_i : A_i \vdash^{2_t} A_{i+1} : \kappa$ (lem. 3.3.4) :

- ou bien $\kappa \equiv \mathrm{Type}$ et alors $A_{i+1} \equiv \mathrm{Prop}$ (lem. 3.3.5) ;
- ou bien $\kappa \equiv \mathrm{Prop}$ et alors $A_{i+1} \not\equiv \mathrm{Prop}$ (thm. 3.3.7 et lem. 3.2.7).

On peut alors placer tous les $x_i : A_i$ avec $A_i \equiv \mathrm{Prop}$ en début d'environnement (lem. 3.5.3) pour former la partie Γ_1, les autres se trouvant à la suite pour constituer Γ_2. □

Remarques

Les éléments de Γ_1 peuvent apparaître dans n'importe quel ordre (lem. 3.5.3).

De même pour Γ_2 car les A_i ne dépendent que des variables $x_j : \mathrm{Prop}$ de Γ_1 (lem. 3.2.10 et thm. 3.3.7).

Dans la suite, on considère que dès que $\Gamma\,\mathsf{wf}^{2_t}$ ou $\Gamma \vdash^{2_t} w : C$, Γ est en fait composé de deux parties Γ_1 et Γ_2 où Γ_1 contient les variables de type Prop.

On se passe alors de mentionner Γ_1 : on y met toutes les variables libres de Γ_2, w et C. Ceci est possible par les propriétés de renforcement (lem. 3.2.19) et d'affaiblissement (lem. 3.2.13), qui nous permettent d'ajouter et de supprimer à loisir des éléments de type Prop dans Γ_1.

Traduction d'un contexte de P-Prop^2

◊ La *traduction d'un contexte* de P-Prop^2 dans un environnement de λ_t^2 est définie par :

$$[\![x_1 : A_1, \ldots, x_n : A_n]\!] := x_1 : [\![A_1]\!], \ldots, x_n : [\![A_n]\!]$$

Lemme 3.5.5 (correction des types dans λ_t^2).
Si $\Gamma \vdash^{2_t} w : C$, alors $C \equiv$ Type ou bien il existe κ tel que $\Gamma \vdash^{2_t} C : \kappa$.

Preuve immédiate par le fait que λ_t^2 et λ_e^2 sont équivalents (thm. 3.3.6 et 3.3.7) et le lemme 3.2.27 précédent. □

Lemme 3.5.6. *Si $\Gamma \vdash^{2_t} w : [\![C]\!]$, alors $\Gamma \vdash^{2_t} [\![C]\!]$: Prop.*

Preuve
De $\Gamma \vdash^{2_t} w : [\![C]\!]$ on déduit qu'il existe κ tel que $\Gamma \vdash^{2_t} [\![C]\!] : \kappa$ (lem. 3.5.5). Or si $\kappa \equiv$ Type, alors $[\![C]\!] \equiv$ Prop (lem. 3.3.5), ce qui n'est pas possible par définition de $[\![\cdot]\!]$. En conséquence $\kappa \equiv$ Prop. □

Motivation triviale universelle

◊ La *motivation triviale universelle* τ est la substitution constante qui à toute variable de type associe $\top$.

Théorème 3.5.7. *Si $\Delta \vDash^{\mathsf{pf}} u : F$, alors pour toute sous-formule G de Δ, F on a $\vDash^{\mathsf{pf}} \tau \cdot G$.*

Preuve dans (Colson et Michel, 2009, thm. 19). □

Lemme 3.5.8. *Si $\Delta \vDash^{\mathsf{pf}} u : F$, alors il existe une dérivation de $\Delta \vDash^{\mathsf{pf}} u : F$ n'utilisant que la motivation τ dans les prémisses des règles (P-Ax), (P-Hyp) et (P-$\forall_e$).*

Preuve par récurrence structurelle sur la dérivation.

Pour chacune de ces trois règles, toutes les formules motivées apparaissent comme sous-formules dans la conclusion. Par conséquent elles sont toutes motivables aussi par τ (thm. 3.5.7). On peut alors remplacer la prémisse $\vDash^{\mathsf{pf}} \sigma \cdot \Delta$ par $\vDash^{\mathsf{pf}} \tau \cdot \Delta$. □

Lemme 3.5.9.
Si $\Delta \vDash^{\mathsf{pf}} w : C$ est une dérivation n'utilisant que τ comme motivation, alors $[\![\Delta]\!] \vdash^{2_t} [\![w]\!] : [\![C]\!]$.

Preuve par récurrence structurelle sur la dérivation.

$$\textbf{(P-Ax)}\ \frac{\vDash^{\mathsf{pf}} \tau \cdot \Delta}{\Delta \vDash^{\mathsf{pf}} \boldsymbol{o} : \top} \quad \text{avec } \Delta = \{x_1 : A_1, \ldots, x_n : A_n\}.$$

Par hypothèse on a des termes $(t_i)_{1 \leq i \leq n}$ tels que

$$\vDash^{\mathsf{pf}} t_i : \tau \cdot A_i$$

d'où par hypothèse de récurrence

$$\vdash^{2_t} [\![t_i]\!] : [\![\tau \cdot A_i]\!]$$

or $[\![\tau \cdot A_i]\!] \equiv [\![A_i]\!][\vec{y} \leftarrow \top]$ (lem. 3.5.1) avec les $\vec{y}$ étant les variables libres de A_i. Et comme aussi (lem. 3.5.6)

$$\vdash^{2_t} [\![\tau \cdot A_i]\!] : \mathrm{Prop}$$

alors par la réciproque du critère de Poincaré (prop. 3.3.8)

$$x_1 : [\![A_1]\!], \ldots, x_n : [\![A_n]\!]\ \mathsf{wf}^{2_t}$$

et donc par (t-ax) on obtient le résultat.

(P-Hyp) $\dfrac{x : F \in \Delta \quad \vdash^{\mathsf{pf}} \tau \cdot \Delta}{\Delta \vdash^{\mathsf{pf}} x : F}$

De même que pour (P-Ax), on montre que $[\![\Delta]\!]\ \mathsf{wf}^{2_t}$. Or comme $x : F \in \Delta$ implique que $x : [\![F]\!] \in [\![\Delta]\!]$, alors par (t-var) on a bien $[\![\Delta]\!] \vdash^{2_t} x : [\![F]\!]$.

$(\rightarrow_i)$ $\dfrac{\Delta, x : A \vdash^{\mathsf{pf}} u : B}{\Delta \vdash^{\mathsf{pf}} \boldsymbol{\lambda} x^A.u : A \rightarrow B}$

Par hypothèse de récurrence on a

$$[\![\Delta]\!], x : [\![A]\!] \vdash^{2_t} [\![u]\!] : [\![B]\!]$$

et donc (lem. 3.5.6) aussi

$$[\![\Delta]\!], x : [\![A]\!] \vdash^{2_t} [\![B]\!] : \mathrm{Prop}$$

D'où par (t-abs) on a bien

$$[\![\Delta]\!] \vdash^{2_t} \lambda x^{[\![A]\!]}.[\![u]\!] : [\![A]\!] \rightarrow [\![B]\!]$$

(car comme $x \notin \mathcal{V}(B)$ alors $x \notin \mathcal{V}([\![B]\!])$).

$(\rightarrow_e)$ $\dfrac{\Delta \vdash^{\mathsf{pf}} u : A \rightarrow B \quad \Delta \vdash^{\mathsf{pf}} v : A}{\Delta \vdash^{\mathsf{pf}} u\ v : B}$

Il suffit d'appliquer l'hypothèse de récurrence sur les deux prémisses pour conclure par (t-app).

$(\forall_i)$ $\dfrac{\Delta \vdash^{\mathsf{pf}} u : B \quad \alpha \notin \mathcal{V}(\Delta)}{\Delta \vdash^{\mathsf{pf}} \boldsymbol{\Lambda} \alpha.u : \forall \alpha.B}$

Par hypothèse de récurrence on a

$$[\![\Delta]\!] \vdash^{2_t} [\![u]\!] : [\![B]\!]$$

On a deux cas selon que $\alpha \in \mathcal{V}(B)$ ou non :

- $\alpha \in \mathcal{V}(B)$: alors $\alpha : \mathrm{Prop}$ se trouve dans l'environnement caché Γ_1, et comme il n'apparaît pas dans $\mathcal{V}(\Delta)$, alors il n'apparaît pas non plus dans $\mathcal{V}([\![\Delta]\!])$. On peut alors faire *remonter* $\alpha : \mathrm{Prop}$ en tête d'environnement par permutations successives (lem. 3.5.3) pour obtenir

$$[\![\Delta]\!], \alpha : \mathrm{Prop} \vdash^{2_t} [\![u]\!] : [\![B]\!]$$

- $\alpha \notin \mathcal{V}(B)$: alors α n'apparaît pas dans l'environnement caché, et donc on peut ajouter $\alpha : \mathrm{Prop}$ à $[\![\Delta]\!]$ par affaiblissement (lem. 3.2.13 et thm. 3.3.6 et 3.3.7) pour obtenir

$$[\![\Delta]\!], \alpha : \mathrm{Prop} \vdash^{2_t} [\![u]\!] : [\![B]\!]$$

Dans ces deux cas, on a aussi (lem. 3.5.6)

$$[\![\Delta]\!], \alpha : \mathrm{Prop} \vdash^{2_t} [\![B]\!] : \mathrm{Prop}$$

et (t-abs) permet de conclure.

$$\textbf{(P-}\forall_e\textbf{)}\ \frac{\Delta \vdash^{\mathsf{pf}} u : \forall\alpha.B \quad \vdash^{\mathsf{pf}} \tau \cdot V}{\Delta \vdash^{\mathsf{pf}} u\ V : [\alpha \leftarrow V] \cdot B}$$

De même que pour (P-Ax) et (P-Hyp), de $\vdash^{\mathsf{pf}} \tau \cdot V$ on déduit

$$z : [\![V]\!]\ \mathsf{wf}^{2_t}$$

avec z une variable fraîche pour toutes les formules considérées. On a alors (lem. 3.3.4)

$$\vdash^{2_t} [\![V]\!] : \kappa$$

avec $\kappa \equiv \mathrm{Prop}$ (sinon $[\![V]\!] \equiv \mathrm{Prop}$ par lem. 3.2.10 et thm. 3.3.6 et 3.3.7).

Par hypothèse de récurrence

$$[\![\Delta]\!] \vdash^{2_t} [\![u]\!] : \forall\alpha^{\mathrm{Prop}}.[\![B]\!]$$

et donc $[\![\Delta]\!]\ \mathsf{wf}^{2_t}$ (lem. 3.3.3). D'où par affaiblissement on a aussi

$$[\![\Delta]\!] \vdash^{2_t} [\![V]\!] : \mathrm{Prop}$$

et par la règle (t-app)

$$[\![\Delta]\!] \vdash^{2_t} [\![u]\!]\ [\![V]\!] : [\![B]\!][\alpha \leftarrow [\![V]\!]\,]$$

or $[\![B]\!][\alpha \leftarrow [\![V]\!]\,] \equiv [\![[\alpha \leftarrow V] \cdot B]\!]$ (lem. 3.5.1). □

Lemme 3.5.10.
Si $\Gamma \vdash^{2_t} w : C$ avec $w \not\equiv \mathrm{Prop}$ alors il existe un terme ou un type w' de P-Prop^2 tel que $[\![w']\!] \equiv w$.

Preuve par récurrence structurelle sur la dérivation.

$$\textbf{(t-abs)}\ \frac{\Gamma, x : A \vdash^{2_t} u : B : \mathrm{Prop}}{\Gamma \vdash^{2_t} \lambda x^A.u : \forall x^A.B}$$

Déjà l'hypothèse de récurrence nous donne un terme u' tel que $[\![u']\!] \equiv u$.
Comme $\Gamma \vdash^{2_t} A : \kappa$ (lem. 3.3.4) est une sous-dérivation, on a deux cas :

- si $\kappa \equiv \mathrm{Type}$, alors $A \equiv \mathrm{Prop}$ (lem. 3.3.5) et dans ce cas $w' := \mathbf{\Lambda} x.u'$ convient ;
- si $\kappa \equiv \mathrm{Prop}$, alors $A \not\equiv \mathrm{Prop}$, et on peut appliquer l'hypothèse de récurrence pour récupérer un terme A' tel que $[\![A']\!] \equiv A$; d'où $w' := \boldsymbol{\lambda} x^{A'}.u'$ convient.

$$\textbf{(t-prod)}\ \frac{\Gamma, x : A \vdash^{2_t} B : \mathrm{Prop} \quad \sigma\ \mathrm{mot}_\Gamma\ \forall x^A.B}{\Gamma \vdash^{2_t} \forall x^A.B : \mathrm{Prop}}$$

L'hypothèse de récurrence donne un terme B' tel que $[\![B']\!] \equiv B$.
Ici aussi on a deux cas à considérer dans la sous-dérivation $\Gamma \vdash^{2_t} A : \kappa$:

- si $\kappa \equiv \mathrm{Type}$, alors $A \equiv \mathrm{Prop}$ (lem. 3.3.5) et dans ce cas $w' := \forall x.B'$ convient ;

- si $\kappa \equiv \mathrm{Prop}$, alors $A \not\equiv \mathrm{Prop}$ et donc $x \notin \mathcal{V}(B)$ (lem. 3.2.10), et l'hypothèse de récurrence donne un terme A' tel que $[\![A']\!] \equiv A$; d'où $w' := A' \to B'$ convient. □

Corollaire 3.5.11. *Si $y_1 : D_1, \ldots, y_n : D_n \vdash^{2_t} w : C$ alors :*
(i) si $D_i \not\equiv \mathrm{Prop}$, alors il existe un terme ou un type D'_i de P-Prop^2 tel que $[\![D'_i]\!] \equiv D_i$;
(ii) si $w \not\equiv \mathrm{Prop}$, alors il existe un terme ou un type w' de P-Prop^2 tel que $[\![w']\!] \equiv w$;
(iii) si $C \not\equiv \mathrm{Prop}, \mathrm{Type}$, alors il existe un terme ou un type C' de P-Prop^2 tel que $[\![C']\!] \equiv C$.

Preuve

(i) De $y_1 : D_1, \ldots, y_i : D_i \vdash^{2_t} D_{i+1} : \kappa$ (lem. 3.3.4) on a deux cas :
- si $\kappa \equiv \mathrm{Type}$, alors $D_{i+1} \equiv \mathrm{Prop}$ (lem. 3.3.5) ;
- si $\kappa \equiv \mathrm{Prop}$, alors $D_{i+1} \not\equiv \mathrm{Prop}$ et donc le lemme 3.5.10 permet de conclure.

(ii) C'est exactement le lemme 3.5.10.

(iii) On a trois cas (lem. 3.5.5) :
- $C \equiv \mathrm{Type}$: alors c'est bon ;
- $y_1 : D_1, \ldots, y_n : D_n \vdash^{2_t} C : \mathrm{Type}$: alors $C \equiv \mathrm{Prop}$ (lem. 3.3.5) et c'est bon ;
- $y_1 : D_1, \ldots, y_n : D_n \vdash^{2_t} C : \mathrm{Prop}$: alors $C \not\equiv \mathrm{Prop}$ et le lemme 3.5.10 permet de conclure.

□

Lemme 3.5.12.
(i) Si $[\![\Delta]\!]\,\mathsf{wf}^{2_t}$ alors il existe une substitution ρ telle que $\vDash^{\mathsf{pf}} \rho \cdot \Delta$;
(ii) Si $[\![\Delta]\!] \vdash^{2_t} [\![C]\!] : \mathrm{Prop}$ alors il existe une substitution ρ telle que $\vDash^{\mathsf{pf}} \rho \cdot \Delta$ et $\vDash^{\mathsf{pf}} \rho \cdot C$;
(iii) Si $[\![\Delta]\!] \vdash^{2_t} [\![w]\!] : [\![C]\!]$ alors $\Delta \vDash^{\mathsf{pf}} w : C$.

Preuve par récurrence structurelle sur la dérivation.

(t-env$_1$) $\dfrac{}{[\![\emptyset]\!]\,\mathsf{wf}^{2_t}}$ Avec ρ la substitution vide, on a trivialement $\vDash^{\mathsf{pf}} \rho \cdot \emptyset$.

(t-env$_2$) $\dfrac{[\![\Delta]\!] \vdash^{2_t} A : \kappa \quad x \notin \mathrm{dom}([\![\Delta]\!])}{[\![\Delta]\!], x : A\,\mathsf{wf}^{2_t}}$

On a deux cas :
- si $\kappa \equiv \mathrm{Prop}$, alors $A \not\equiv \mathrm{Prop}$, et A est l'image d'un A' par $[\![\cdot]\!]$ (cor. 3.5.11) ; l'hypothèse de récurrence sur la prémisse nous donne alors bien une substitution ρ vérifiant $\vDash^{\mathsf{pf}} \rho \cdot (\Delta, A')$;
- si $\kappa \equiv \mathrm{Type}$, alors $A \equiv \mathrm{Prop}$ (lem. 3.3.5) et $x : A$ est dans la partie cachée de l'environnement ; or comme $[\![\Delta]\!]\,\mathsf{wf}^{2_t}$ (lem. 3.3.3) est une sous-dérivation, alors l'hypothèse de récurrence nous donne bien une substitution ρ telle que $\vDash^{\mathsf{pf}} \rho \cdot \Delta$.

(t-ax) $\dfrac{[\![\Delta]\!]\,\mathsf{wf}^{2_t}}{[\![\Delta]\!] \vdash^{2_t} [\![\boldsymbol{o}]\!] : [\![\top]\!] : \mathrm{Prop}}$

Par hypothèse de récurrence sur la prémisse on a une substitution ρ telle que $\vDash^{\mathsf{pf}} \rho \cdot \Delta$, on peut alors dériver $\Delta \vDash^{\mathsf{pf}} \boldsymbol{o} : \top$ par (P-Ax), et on a aussi $\vDash^{\mathsf{pf}} \rho \cdot (\Delta, \top)$.

(t-var) $\dfrac{[\![\Delta, x : A, \Delta']\!]\,\mathsf{wf}^{2_t}}{[\![\Delta, x : A, \Delta']\!] \vdash^{2_t} [\![x]\!] : [\![A]\!]}$

Par hypothèse de récurrence on a une substitution ρ telle que $\vDash^{\mathsf{pf}} \rho \cdot (\Delta, x : A, \Delta')$ et donc par (P-Hyp) on a bien $\Delta, x : A, \Delta' \vDash^{\mathsf{pf}} x : A$.

(t-abs) Selon le type de x, on est dans un des deux cas suivants :

$$\frac{[\![\Delta]\!], x : [\![A]\!] \vdash^{2_t} [\![u]\!] : [\![B]\!] : \mathrm{Prop}}{[\![\Delta]\!] \vdash^{2_t} [\![\boldsymbol{\lambda} x^A.u]\!] : [\![A \to B]\!]} \qquad \frac{[\![\Delta]\!], x : \mathrm{Prop} \vdash^{2_t} [\![u]\!] : [\![B]\!] : \mathrm{Prop}}{[\![\Delta]\!] \vdash^{2_t} [\![\boldsymbol{\Lambda} x.u]\!] : [\![\forall x.B]\!]}$$

Chacun se résout facilement en appliquant l'hypothèse de récurrence à la première prémisse et en utilisant respectivement les règles $(\to_i)$ et $(\forall_i)$.

(t-app) De même, selon le type de x on se trouve dans l'un des deux cas suivants :

$$\frac{[\![\Delta]\!] \vdash^{2_t} [\![u]\!] : [\![A \to B]\!] \quad [\![\Delta]\!] \vdash^{2_t} [\![v]\!] : [\![A]\!]}{[\![\Delta]\!] \vdash^{2_t} [\![u\ v]\!] : [\![B]\!]} \qquad \frac{[\![\Delta]\!] \vdash^{2_t} [\![u]\!] : [\![\forall x.B]\!] \quad [\![\Delta]\!] \vdash^{2_t} [\![v]\!] : \mathrm{Prop}}{[\![\Delta]\!] \vdash^{2_t} [\![u\ v]\!] : [\![B]\!][x \leftarrow [\![v]\!]\,]}$$

Chacun se résout facilement en appliquant l'hypothèse de récurrence aux prémisses et en utilisant respectivement les règles $(\to_e)$ et $(\forall_e)$.

(t-prod) Encore une fois, selon le type de x on doit traiter deux cas :

- $\dfrac{[\![\Delta]\!], x : [\![A]\!] \vdash^{2_t} [\![B]\!] : \mathrm{Prop} \quad \sigma\,\mathrm{mot}_{[\![\Delta]\!]}[\![A \to B]\!]}{[\![\Delta]\!] \vdash^{2_t} [\![A \to B]\!] : \mathrm{Prop}}$ avec $\Delta \equiv y_1 : D_1, \ldots, y_n : D_n$.

  Par définition de $\sigma\,\mathrm{mot}_{[\![\Delta]\!]}[\![A \to B]\!]$, on a :
  - des termes t_i tels que $\vdash^{2_t} t_i : [\![D_i]\!][\vec{\alpha} \leftarrow \vec{E}\,]$ avec les $\vec{\alpha}$ les variables libres des D_i et donc les E_j tels que $\vdash^{2_t} E_j : \mathrm{Prop}$ (lem. 3.2.10).
    On a alors des termes E'_j et t'_i tels que $[\![E'_j]\!] \equiv E_j$ et $[\![t'_i]\!] \equiv t_i$ (cor. 3.5.11), d'où

    $$\vdash^{2_t} [\![t'_i]\!] : [\![D_i]\!][\vec{\alpha} \leftarrow [\![\vec{E'}]\!]\,] \quad \text{c'est-à-dire (lem. 3.5.1)} \quad \vdash^{2_t} [\![t'_i]\!] : [\![[\vec{\alpha} \leftarrow \vec{E'}\,] \cdot D_i]\!]$$

    ce qui, par hypothèse de récurrence, donne

    $$\vDash^{\mathsf{pf}} t'_i : [\vec{\alpha} \leftarrow \vec{E'}\,] \cdot D_i \quad \text{autrement dit} \quad \vDash^{\mathsf{pf}} \rho \cdot D_i$$

    avec $\rho \equiv [\vec{\alpha} \leftarrow \vec{E'}\,]$.
  - et un terme u tel que

    $$\vdash^{2_t} u : [\![A \to B]\!][\vec{\alpha} \leftarrow \vec{E}\,]$$

    ce qui par le même raisonnement nous conduit à

    $$\vDash^{\mathsf{pf}} \rho \cdot (A \to B)$$

- $\dfrac{[\![\Delta]\!], x : \mathrm{Prop} \vdash^{2_t} [\![B]\!] : \mathrm{Prop} \quad \sigma\,\mathrm{mot}_{[\![\Delta]\!]}[\![\forall x.B]\!]}{[\![\Delta]\!] \vdash^{2_t} [\![\forall x.B]\!] : \mathrm{Prop}}$ On conclut par le même raisonnement.

□

Théorème 3.5.13. *$\Delta \vDash^{\mathsf{pf}} w : C$ si et seulement si $[\![\Delta]\!] \vdash^{2_t} [\![w]\!] : [\![C]\!]$.*

Preuve

$\boxed{\Rightarrow}$ De $\Delta \vdash^{\mathsf{pf}} w : C$, on construit une dérivation n'utilisant que τ comme motivation (lem. 3.5.8), et alors $[\![\Delta]\!] \vdash^{2_t} [\![w]\!] : [\![C]\!]$ (lem. 3.5.9).

$\boxed{\Leftarrow}$ C'est exactement le (iii) du lemme 3.5.12 précédent. □

Traduction des termes de λ_t^2 vers ceux de P-Prop^2

- À une dérivation $\Gamma \vdash^{2_t} w : C$ où $C \not\equiv \kappa$, on peut associer un unique antécédent de w et de C par $[\![\cdot]\!]$ (cor. 3.5.11). On note cet antécédent par $[\![\cdot]\!]^{-1}$.
- De même il existe un unique antécédent de Γ, qu'on désigne par $[\![\Gamma]\!]^{-1}$.

Théorème 3.5.14. $\Gamma \vdash^{2_t} w : C$ *avec* $C \not\equiv \mathrm{Prop}, \mathrm{Type}$ *si et seulement si* $[\![\Gamma]\!]^{-1} \vdash^{\mathsf{pf}} [\![w]\!]^{-1} : [\![C]\!]^{-1}$.

Preuve $\Gamma \vdash^{2_t} w : C$ se récrit en $[\![[\![\Gamma]\!]^{-1}]\!] \vdash^{2_t} [\![[\![w]\!]^{-1}]\!] : [\![[\![C]\!]^{-1}]\!]$ d'où l'on déduit l'équivalence (thm. 3.5.13) avec $[\![\Gamma]\!]^{-1} \vdash^{\mathsf{pf}} [\![w]\!]^{-1} : [\![C]\!]^{-1}$. □

Corollaire 3.5.15. *On peut plonger le calcul propositionnel du second ordre Prop^2 et λ^2 dans les calculs λ_e^2, λ_t^2 et λ_p^2.*

Preuve Le lemme 3.6.1 suivant donne un plongement de Prop^2 dans P-Prop^2, ce qui est suffisant puisqu'on peut plonger P-Prop^2 dans λ_t^2 (thm. 3.5.13), λ_t^2 étant équivalent à λ_e^2 et λ_p^2 (thm. 3.3.6, 3.3.7 et 3.4.3, 3.4.4).

D'autre part, λ^2 et Prop^2 sont deux formalisations (équivalentes) du même calcul. □

3.6 Indécidabilité de la vérification du typage

Habitation de type (*type inhabitation*)

◊ Pour un système fonctionnel donné, *le problème de l'habitation de type* est le suivant :

entrée : un contexte (ou environnement) Γ, et un type A ;

sortie : « vrai » s'il existe un terme t tel que $\Gamma \vdash^{\star} t : A$, et « faux » sinon.

Lemme 3.6.1.

L'habitation de type pour Prop^2 est réductible à l'habitation de type pour P-Prop^2, i.e. quels que soient Δ et A :

$$\textit{il existe } t \textit{ tel que } \Delta \vdash^{\mathsf{f}} t : A \text{ si et seulement si } \textit{il existe un terme } t' \textit{ tel que } \Delta^{\gamma} \vdash^{\mathsf{pf}} t' : A^{\gamma}$$

avec $^{\gamma}$ une traduction associant toute formule de Prop^2 à une formule de P-Prop^2.

Preuve

Le résultat est prouvé (constructivement) dans Colson et Michel (2008) à propos des systèmes formels correspondant aux systèmes fonctionnels Prop^2 et P-Prop^2. La traduction $^{\gamma}$, inspirée de la A-traduction de Friedman (1978), consiste à remplacer partout dans les types les occurences de variables de type α par $\alpha \vee \gamma$ où γ est une variable de type fraîche. □

Lemme 3.6.2. *L'habitation de type pour Prop^2 est indécidable.*

Preuve par Urzyczyn (1997). □

Lemme 3.6.3. *L'habitation de type pour P-Prop^2 est indécidable.*

Preuve par contradiction.

Supposons que l'habitation de type pour P-Prop^2 soit décidée par un algorithme D, i.e.

$$D(\Delta, A) = \text{vrai } \textit{si et seulement si} \text{ il existe un terme } t \text{ tel que } \Delta \vdash^{\mathsf{pf}} t : A$$

On peut alors construire un algorithme D' capable de décider l'habitation de type pour Prop^2 :

$$D'(\Delta, A) := D(\Delta^{\gamma}, A^{\gamma})$$

En effet :

$$\begin{aligned} D'(\Delta, A) = \text{vrai} \quad & \textit{ssi} \quad D(\Delta^{\gamma}, A^{\gamma}) = \text{vrai} \\ & \textit{ssi} \quad \text{il existe } t' \text{ tel que } \Delta^{\gamma} \vdash^{\mathsf{pf}} t' : A^{\gamma} \\ & \textit{ssi} \quad \text{il existe } t \text{ tel que } \Delta \vdash^{\mathsf{f}} t : A \qquad \text{(lem. 3.6.1)} \end{aligned}$$

Or l'habitation de type pour Prop^2 est indécidable (lem. 3.6.2). □

Vérification du typage (*type checking*)

◊ Pour un système fonctionnel donné, *le problème de la vérification du typage* est le suivant :

entrée : un contexte (ou environnement) Γ, un terme t et un type A ;

sortie : « vrai » s'il existe une dérivation $\Gamma \vdash^{\star} t : A$, et « faux » sinon.

Lemme 3.6.4. *L'habitation de type pour P-Prop^2 avec un contexte vide est réductible à la vérification du typage pour λ_t^2 avec un environnement vide, i.e. quel que soit le type A*

$$\textit{il existe } t \textit{ tel que } \vdash^{\mathsf{pf}} t : A \textit{ avec } A \textit{ clos} \text{ si et seulement si } \vdash^{2_t} [\![A]\!] : \mathrm{Prop}$$

Preuve

$\boxed{\Rightarrow}$ De $\vdash^{\mathsf{pf}} t : A$ on déduit $\vdash^{2_t} [\![t]\!] : [\![A]\!]$ (thm. 3.5.13), et par correction des types (lem. 3.5.5) $\vdash^{2_t} [\![A]\!] : \kappa$. Or $\kappa \not\equiv \mathrm{Type}$ car sinon $[\![A]\!] \equiv \mathrm{Prop}$ (lem. 3.3.5) ce qui n'est pas possible par définition de $[\![\cdot]\!]$, donc $\kappa \equiv \mathrm{Prop}$.

$\boxed{\Leftarrow}$ De $\vdash^{2_t} [\![A]\!] : \mathrm{Prop}$ on peut construire un terme a tel que $\vdash^{2_t} a : [\![A]\!]$ (lem. 3.2.12 et thm. 3.3.6 et 3.3.7). Or a est l'image d'un terme t par $[\![\cdot]\!]$ (cor. 3.5.11), i.e. $[\![t]\!] \equiv a$, d'où $\vdash^{2_t} [\![t]\!] : [\![A]\!]$ et finalement $\vdash^{\mathsf{f}} t : A$ (thm. 3.5.13). □

Lemme 3.6.5. *L'habitation de type pour P-Prop^2 est réductible à l'habitation de type pour P-Prop^2 avec un contexte vide, i.e. pour tout type A*

$$\textit{il existe } t \textit{ tel que } \Delta \vdash^{\mathsf{pf}} t : A \text{ si et seulement si } \textit{il existe } t' \textit{ tel que } \vdash^{\mathsf{pf}} t' : \forall\vec{\alpha}.\Delta{\to}A$$

où $\forall\vec{\alpha}.\Delta{\to}A$ est clos ; $\vec{\alpha}$ sont les variables libres de Δ et A ; et $\Delta{\to}A$ dénote $B_1{\to}\ldots{\to}B_n{\to}A$ avec $\Delta = \{y_1 : B_1, \ldots, y_n : B_n\}$.

Preuve

$\boxed{\Rightarrow}$ De $\Delta \vdash^{\mathsf{pf}} t : A$ on a $\vdash^{\mathsf{pf}} \boldsymbol{\lambda}\Delta.t : \Delta{\to}A$ par $(\to_i)$ puis $\vdash^{\mathsf{pf}} \boldsymbol{\Lambda}\vec{\alpha}.\boldsymbol{\lambda}\Delta.t : \forall\vec{\alpha}.\Delta{\to}A$ par $(\forall_i)$. Donc $t' := \boldsymbol{\Lambda}\vec{\alpha}.\boldsymbol{\lambda}\Delta.t$ convient.

$\boxed{\Leftarrow}$ Inversement de $\vdash^{\mathsf{pf}} t' : \forall\vec{\alpha}.\Delta{\to}A$ par $(\forall_e)$ on a $\vdash^{\mathsf{pf}} t'\ \vec{\alpha} : \Delta{\to}A$ puisque les $\vec{\alpha}$ sont motivables par $\top$, puis par affaiblissement on a $\Delta \vdash^{\mathsf{pf}} t'\ \vec{\alpha} : \Delta{\to}A$ et finalement par $(\to_e)$ on obtient $\Delta \vdash^{\mathsf{pf}} t'\ \vec{\alpha}\ \Delta : A$, c'est-à-dire que $t := t'\ \vec{\alpha}\ \Delta$ convient.

L'affaiblissement dans P-Prop^2 est prouvé par (Colson et Michel, 2009, prop. 21) sous réserve de motiver la formule introduite : ici les formules de Δ sont toutes motivables par la substitution triviale τ puisqu'elles apparaissent comme sous-formules dans $\forall\vec{\alpha}.\Delta{\to}A$ (thm. 3.5.7).□

Théorème 3.6.6. *La vérification du typage pour λ_t^2 est indécidable.*

Preuve par contradiction.

Supposons que la vérification du typage pour λ_t^2 soit décidée par un algorithme D, i.e.

$$D(\Gamma, t, A) = \text{vrai } \textit{si et seulement si } \Gamma \vdash^{2_t} t : A$$

On peut alors construire un algorithme D' pour décider l'habitation de type pour P-Prop^2 :

$$D'(\Delta, A) := D([\,], [\![\forall\vec{\alpha}.\Delta{\to}A]\!], \mathrm{Prop})$$

avec $\vec{\alpha}$ les variables libres de Δ et de A.

En effet :

$$\begin{aligned} D'(\Delta, A) = \text{vrai ssi } & D([\,], [\![\forall\vec{\alpha}.\Delta{\to}A]\!], \mathrm{Prop}) = \text{vrai} \\ \text{ssi } & \vdash^{2_t} [\![\forall\vec{\alpha}.\Delta{\to}A]\!] : \mathrm{Prop} \\ \text{ssi il existe } t \text{ tel que } & \vdash^{\mathsf{pf}} t : \forall\vec{\alpha}.\Delta{\to}A \quad \text{(lem. 3.6.4)} \\ \text{ssi il existe } t' \text{ tel que } & \Delta \vdash^{\mathsf{pf}} t' : A \quad \text{(lem. 3.6.5)} \end{aligned}$$

Or l'habitation de type pour P-Prop^2 est indécidable (lem. 3.6.3). □

Corollaire 3.6.7. *La vérification du typage pour λ_p^2 est indécidable.*

Preuve est conséquence immédiate du fait que λ_t^2 et λ_p^2 sont équivalents (thm. 3.3.6 et 3.3.7).□

Vérification du typage (*type checking*) pour motivations explicites

◊ Pour un système fonctionnel à motivations explicites donné, *le problème de la vérification du typage pour motivations explicites* est le suivant :

entrée : un contexte (ou environnement) Γ, une substitution σ, un terme t et un type A ;

sortie : «vrai» s'il existe une dérivation $\Gamma \vdash_\sigma^\star t : A$, et «faux» sinon.

Théorème 3.6.8. *La vérification du typage pour λ_e^2 est indécidable.*

Preuve par contradiction.

Supposons que la vérification du typage pour λ_e^2 soit décidée par un algorithme D, i.e.

$$D(\Gamma, \sigma, t, A) = \text{vrai } \textit{si et seulement si } \Gamma \vdash_\sigma^{2_e} t : A$$

On peut alors construire un algorithme D' pour décider la vérification du typage pour λ_t^2 :

$$D'(\Gamma, t, A) := \begin{cases} D([\,], [\,], \forall\Gamma.\top, \text{Prop}) & \text{si } A \equiv \text{Type et } t \equiv \text{Prop} \\ \text{faux} & \text{si } A \equiv \text{Type et } t \not\equiv \text{Prop} \\ \text{faux} & \text{si } A \equiv \text{Prop et } t \equiv \text{Prop} \\ D([\,], [\,], \forall\Gamma.\forall z^t.\top, \text{Prop}) & \text{si } A \equiv \text{Prop et } t \not\equiv \text{Prop} \\ D([\,], [\,], \lambda\Gamma.t, \forall\Gamma.A) & \text{sinon} \end{cases}$$

avec $\lambda\Gamma.A \equiv \lambda y_1^{B_1}.\ldots.\lambda y_n^{B_n}.A$ si $\Gamma \equiv y_1 : B_1, \ldots, y_n : B_n$, de même pour $\forall\Gamma.A$.

Tout d'abord on montre que

$$D([\,], [\,], \forall\Gamma.\top, \text{Prop}) = \text{vrai} \quad \textit{ssi} \quad \text{il existe } \sigma \text{ telle que } \Gamma\, \mathsf{wf}_\sigma^{2_e}$$

$\boxed{\Rightarrow}$ De $\vdash_{[\,]}^{2_e} \forall\Gamma.\top : \text{Prop}$ par génération (lem. 3.2.7) on obtient une substitution σ telle que $\Gamma \vdash_\sigma^{2_e} \top : \kappa$, et finalement (lem. 3.2.4) $\Gamma\, \mathsf{wf}_\sigma^{2_e}$.

$\boxed{\Leftarrow}$ De $\Gamma\, \mathsf{wf}_\sigma^{2_e}$ par (e-ax) on a $\Gamma \vdash_\sigma^{2_e} o : \top : \text{Prop}$ puis par (e-abs) et (e-prod) (lem. 3.2.20) $\vdash_{[\,]}^{2_e} \lambda\Gamma.o : \forall\Gamma.\top : \text{Prop}$ et donc $D([\,], [\,], \forall\Gamma.\top, \text{Prop}) = \text{vrai}$.

Nous pouvons maintenant montrer que D' décide bien la vérification du typage pour λ_t^2 :

$$D'(\Gamma, t, A) = \text{vrai} \quad \textit{ssi} \quad \Gamma \vdash^{2_t} t : A$$

- $A \equiv \text{Type}$ et $t \equiv \text{Prop}$:

$$\begin{aligned} D'(\Gamma, t, A) = \text{vrai} \;\; & \textit{ssi} \;\; D([\,], [\,], \forall\Gamma.\top, \text{Prop}) = \text{vrai} \\ & \textit{ssi} \;\; \text{il existe } \sigma\ \Gamma\, \mathsf{wf}_\sigma^{2_e} \\ & \textit{ssi} \;\; \text{il existe } \sigma\ \Gamma \vdash_\sigma^{2_e} \text{Prop} : \text{Type} \quad \text{((e-ax) et lem. 3.2.4)} \\ & \textit{ssi} \;\; \Gamma \vdash^{2_t} \text{Prop} : \text{Type} \quad \text{(thm. 3.3.6 et 3.3.7)} \end{aligned}$$

- $A \equiv \text{Type}$ et $t \not\equiv \text{Prop}$:

$$D'(\Gamma, t, A) = \text{faux} \;\; \textit{ssi} \;\; \Gamma \not\vdash^{2_t} t : \text{Type} \quad \text{(lem. 3.2.8)}$$

- $A \equiv \mathrm{Prop}$ et $t \equiv \mathrm{Prop}$:

$$D'(\Gamma, t, A) = \text{faux} \quad \textit{ssi} \quad \Gamma \not\vdash^{2_\mathrm{t}} \mathrm{Prop} : \mathrm{Prop} \quad \text{(lem. 3.2.7)}$$

- $A \equiv \mathrm{Prop}$ et $t \not\equiv \mathrm{Prop}$:

$$\begin{array}{rll}
D'(\Gamma, t, A) = \text{vrai} & \textit{ssi} & D([\,], [\,], \forall\Gamma.\forall z^t.\top, \mathrm{Prop}) = \text{vrai} \\
& \textit{ssi} & \text{il existe } \sigma \text{ et } w \ \Gamma, z : t \ \mathsf{wf}^{2_\mathrm{e}}_{\sigma::(z \mapsto w)} \\
& \textit{ssi} & \text{il existe } \sigma \text{ et } w \ \Gamma \vdash^{2_\mathrm{e}}_{\sigma} t : \kappa \qquad \text{(lem. 3.2.5, (e-env}_2\text{) et 3.2.12)} \\
& \textit{ssi} & \text{il existe } \sigma \text{ et } w \ \Gamma \vdash^{2_\mathrm{e}}_{\sigma} t : \mathrm{Prop} \qquad \text{(lem. 3.2.8)} \\
& \textit{ssi} & \Gamma \vdash^{2_\mathrm{t}} t : \mathrm{Prop} \qquad \text{(thm. 3.3.6 et 3.3.7)}
\end{array}$$

- $A \not\equiv \kappa$:

$$\begin{array}{rll}
D'(\Gamma, t, A) = \text{vrai} & \textit{ssi} & D([\,], [\,], \lambda\Gamma.t, \forall\Gamma.A) = \text{vrai} \\
& \textit{ssi} & \vdash^{2_\mathrm{e}}_{[\,]} \lambda\Gamma.t : \forall\Gamma.A \\
& \textit{ssi} & \vdash^{2_\mathrm{e}}_{[\,]} \lambda\Gamma.t : \forall\Gamma.A : \kappa \qquad \text{(lem. 3.2.27)} \\
& \textit{ssi} & \vdash^{2_\mathrm{e}}_{[\,]} \lambda\Gamma.t : \forall\Gamma.A : \mathrm{Prop} \qquad \text{(lem. 3.2.8)} \\
& \textit{ssi} & \text{il existe } \sigma \ \Gamma \vdash^{2_\mathrm{e}}_{\sigma} t : A : \mathrm{Prop} \qquad \text{(lem. 3.2.7, lem. 3.2.20)} \\
& \textit{ssi} & \Gamma \vdash^{2_\mathrm{t}} t : A \qquad \text{(thm. 3.3.6 et 3.3.7)}
\end{array}$$

Or la vérification du typage pour λ^2_t est indécidable (thm. 3.6.6). □

Chapitre 4

Sous-système pédagogique d'ordre supérieur

4.1 Lambda-calcul d'ordre supérieur – λ^ω 119
4.2 Avec motivations totales explicites – λ_e^ω 122

4.1 Lambda-calcul d'ordre supérieur – λ^ω

4.1.1 Définition du système

Règles d'inférence de λ^ω

◊ On définit la relation $\vDash^\omega$ entre un environnement et deux termes bruts, ainsi que la relation wf^ω sur les environnements par les règles suivantes :

$$\frac{}{[\,]\,\mathsf{wf}^\omega}\ (\omega\text{-env}_1) \qquad \frac{\Gamma \vDash^\omega A : \kappa \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^\omega}\ (\omega\text{-env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^\omega}{\Gamma \vDash^\omega \mathrm{Prop} : \mathrm{Type}}\ (\omega\text{-ax}) \qquad \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^\omega}{\Gamma, x : A, \Gamma' \vDash^\omega x : A}\ (\omega\text{-var})$$

$$\frac{\Gamma, x : A \vDash^\omega u : B : \mathrm{Prop}}{\Gamma \vDash^\omega \lambda x^A.u : \forall x^A.B}\ (\omega\text{-abs}_\star) \qquad \frac{\Gamma, x : A \vDash^\omega u : B : \mathrm{Type} \quad \Gamma \vDash^\omega A : \mathrm{Type}}{\Gamma \vDash^\omega \lambda x^A.u : A \to B}\ (\omega\text{-abs}_\Box)$$

$$\frac{\Gamma \vDash^\omega u : \forall x^A.B : \mathrm{Prop} \quad \Gamma \vDash^\omega v : A}{\Gamma \vDash^\omega u\ v : B[x \leftarrow v\,]}\ (\omega\text{-app}_\star) \qquad \frac{\Gamma \vDash^\omega u : A \to B : \mathrm{Type} \quad \Gamma \vDash^\omega v : A}{\Gamma \vDash^\omega u\ v : B}\ (\omega\text{-app}_\Box)$$

$$\frac{\Gamma, x : A \vDash^\omega B : \mathrm{Prop}}{\Gamma \vDash^\omega \forall x^A.B : \mathrm{Prop}}\ (\omega\text{-prod}_\star) \qquad \frac{\Gamma, x : A \vDash^\omega B : \mathrm{Type} \quad \Gamma \vDash^\omega A : \mathrm{Type}}{\Gamma \vDash^\omega A \to B : \mathrm{Type}}\ (\omega\text{-prod}_\Box)$$

$$\frac{\Gamma \vDash^\omega u : A \quad A =_\beta A' \quad \Gamma \vDash^\omega A' : \mathrm{Prop}}{\Gamma \vDash^\omega u : A'}\ (\omega\text{-conv})$$

4.1.2 Résultats

Ordre

- ◊ Un *ordre* est un type simple sur Prop.
- ◊ On notera $\mathcal{O}, \mathcal{O}', \ldots, \mathcal{O}_1, \mathcal{O}_2, \ldots$ pour désigner un ordre.

Lemme 4.1.1 (voir lem. 1.4.1).

(i) Si $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^\omega$*, alors pour tout* i*,* $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$ *et* $x_i \not\equiv x_j$ *pour tout* $i \neq j$ *;*

(ii) Si $x_1 : A_1, \ldots, x_n : A_n \vdash^\omega w : C$*, alors pour tout* i*,* $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$ *et* $x_i \not\equiv x_j$ *pour tout* $i \neq j$ *et* $\mathcal{V}(w, C) \subseteq \{x_1, \ldots, x_n\}$*.*

Lemme 4.1.2 (voir lem. 1.4.2). *Si* $\Gamma \,\mathsf{wf}^\omega$ *ou* $\Gamma \vdash^\omega w : C$*, alors* $\mathrm{Type} \notin \mathcal{S}(\Gamma)$ *et* $\mathrm{Type} \notin \mathcal{S}(w)$*.*

Lemme 4.1.3 (validité des environnements, voir lem. 1.4.3).

(i) si $\Gamma \,\mathsf{wf}^\omega$*, alors pour tout environnement* $\Gamma' \preccurlyeq \Gamma$*,* $\Gamma' \,\mathsf{wf}^\omega$ *en est une sous-dérivation ;*

(ii) si $\Gamma \vdash^\omega w : C$*, alors pour tout environnement* $\Gamma' \preccurlyeq \Gamma$*,* $\Gamma' \,\mathsf{wf}^\omega$ *en est une sous-dérivation stricte.*

Lemme 4.1.4 (validité des types des environnements, voir lem. 1.4.4).

Si $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}^\omega$*, alors pour tout* i *il existe* κ *tel que* $x_1 : A_1, \ldots, x_i : A_i \vdash^\omega A_{i+1} : \kappa$ *en est une sous-dérivation stricte.*

Lemme 4.1.5 (affaiblissement, voir lem. 1.4.5).

Si $\Gamma \vdash^\omega w : C$*,* $\Gamma' \,\mathsf{wf}^\omega$ *et* $\Gamma \subseteq \Gamma'$*, alors* $\Gamma' \vdash^\omega w : C$*.*

Lemme 4.1.6 (unicité du type, voir lem. 1.4.6). *Si* $\Gamma \vdash^\omega w : C$ *et* $\Gamma \vdash^\omega w : C'$*, alors* $C =_\beta C'$*.*

Lemme 4.1.7 (de substitution, voir lem. 1.4.7).

(i) Si $\Gamma, y : C, \Gamma' \,\mathsf{wf}^\omega$ *et* $\Gamma \vdash^\omega w : C$*, alors* $\Gamma, \Gamma'[y \leftarrow w] \,\mathsf{wf}^\omega$ *;*

(ii) Si $\Gamma, y : C, \Gamma' \vdash^\omega t : D$ *et* $\Gamma \vdash^\omega w : C$*, alors* $\Gamma, \Gamma'[y \leftarrow w] \vdash^\omega t[y \leftarrow w] : D[y \leftarrow w]$*.*

Lemme 4.1.8 (génération, voir lem. 1.4.8).

Soit $\Gamma \vdash^\omega t : T$*, un de ces cas se présente :*

(i) si $t \equiv \mathrm{Prop}$*, alors* $T =_\beta \mathrm{Type}$ *;*

(ii) si $t \equiv x$*, alors il existe* $(x : A) \in \Gamma$ *avec* $T =_\beta A$ *;*

(iii) si $t \equiv \lambda x^A.u$*, alors il existe* B *et* κ *tels que* $\Gamma, x : A \vdash^\omega u : B : \kappa$ *est une sous-dérivation stricte avec* $T =_\beta \forall x^A.B$ *;*

(iv) si $t \equiv u\ v$*, alors il existe* A *et* B *tels que* $\Gamma \vdash^\omega u : \forall x^A.B$ *et* $\Gamma \vdash^\omega v : A$ *sont des sous-dérivations strictes avec* $T =_\beta B[x \leftarrow v]$ *;*

(v) si $t \equiv \forall x^A.B$*, alors il existe* κ *tel que* $\Gamma, x : A \vdash^\omega B : \kappa$ *est une sous-dérivation stricte avec* $T =_\beta \kappa$*.*

Lemme 4.1.9 (correction des types, voir lem. 1.4.9).

Si $\Gamma \vdash^\omega w : C$*, alors* $C \equiv \mathrm{Type}$ *ou bien il existe* κ *tel que* $\Gamma \vdash^\omega C : \kappa$*.*

Lemme 4.1.10 (voir lem. 1.4.14). *Si* $\Gamma \vdash^\omega \mathcal{O} : C$ *alors* $C \equiv \mathrm{Type}$*.*

Lemme 4.1.11 (voir lem. 1.4.13).

(i) Si $\Gamma \vdash^\omega C : \mathrm{Type}$ *alors* C *est un ordre et la dernière règle de la dérivation est (*ω*-ax) ou (*ω*-prod*$_\Box$*) ;*

(ii) Si $\Gamma \vDash^\omega C : \mathcal{O}$ alors la dernière règle de la dérivation est (ω-var), (ω-abs$_\Box$), (ω-app$_\Box$) ou (ω-prod$_\star$).

Preuve par récurrence structurelle sur la dérivation (utiliser lem. 4.1.10 pour éliminer des cas). □

Théorème 4.1.12 (λ^ω est un sous-système de CC).

(i) si $\Gamma\,\mathsf{wf}^\omega$, alors $\Gamma\,\mathsf{wf}^\mathsf{c}$;

(ii) si $\Gamma \vDash^\omega w : C$, alors $\Gamma \vdash^\mathsf{c} w : C$.

Preuve par récurrence structurelle sur la dérivation.

$$(\omega\textbf{-app}_\Box)\ \frac{\Gamma \vDash^\omega u : A \to B : \mathrm{Type} \quad \Gamma \vDash^\omega v : A}{\Gamma \vDash^\omega u\ v : B}$$

B étant un ordre (lem. 4.1.11), il n'a pas de variables libres, et donc $B[x \leftarrow v] \equiv B$ avec $A \to B \equiv \forall x^A.B$: (c-app) est donc plus générale que (ω-app$_\Box$). □

4.2 Avec motivations totales explicites – λ_e^ω

4.2.1 Définition du système

Termes bruts

◊ On ajoute aux termes bruts de λ^ω les constantes suivantes : o et $\top$.

Motivation d'un prédicat

◊ Le fait que *la substitution σ motive le prédicat P d'ordre $\mathcal{O}$*, noté $\sigma\,\mathrm{mot}^{\mathcal{O}}(P)$, est défini par :

$$\begin{aligned}\sigma\,\mathrm{mot}^{\mathrm{Prop}}(P) &:= \text{il existe un terme } t \text{ et un type } R \text{ tels que } \overset{\omega_e}{\vdash}_{[]} t : R \text{ et } \sigma(P) \overset{*}{\rightsquigarrow}_\beta R\\ \sigma\,\mathrm{mot}^{\mathcal{O}_1\to\mathcal{O}_2}(P). &:= \text{il existe un terme } u \text{ tel que } \overset{\omega_e}{\vdash}_{[]} u : \mathcal{O}_1 \text{ et } \sigma{::}(x \mapsto u)\,\mathrm{mot}^{\mathcal{O}_2}(P\ x).\end{aligned}$$

avec $x \in \mathcal{F}(P) \cap \mathcal{F}(\mathrm{dom}(\sigma))$ une variable fraîche pour P et σ.

Remarques

Les contraintes de λ_e^2 de la forme $\overset{2_e}{\vdash}_{[]} t : \sigma(A)$ sont un cas particulier de motivabilité lorsque le prédicat est d'ordre Prop (les types sont irréductibles dans λ_e^2).

$\sigma\,\mathrm{mot}^{\mathcal{O}_1\to\ldots\to\mathcal{O}_n\to\mathrm{Prop}}(P)$ dénote l'existence de termes $(u_i)_{1\le i\le n}$, d'un terme t et d'un type R tels que $\overset{\omega_e}{\vdash}_{[]} t : R$ avec $\sigma(P)\ \vec{u} \overset{*}{\rightsquigarrow}_\beta R$ et $\overset{\omega_e}{\vdash}_{[]} u_i : \mathcal{O}_i$.

Règles d'inférence de λ_e^ω

◊ On définit la relation $\overset{\omega_e}{\vdash}$ entre un environnement, deux termes bruts et une substitution, ainsi que la relation wf^{ω_e} entre un environnement et une substitution par les règles suivantes :

$$\frac{}{[\,]\,\mathsf{wf}^{\omega_e}_{[]}}\ (\omega e\text{-env}_1)$$

$$\frac{\Gamma \overset{\omega_e}{\vdash}_\sigma A : \kappa \quad \overset{\omega_e}{\vdash}_{[]} a : A' \quad \sigma(A) \overset{*}{\rightsquigarrow}_\beta A' \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^{\omega_e}_{\sigma::(x\,\mapsto\,a)}}\ (\omega e\text{-env}_2)$$

$$\frac{\Gamma\,\mathsf{wf}^{\omega_e}_\sigma}{\Gamma \overset{\omega_e}{\vdash}_\sigma o : \top : \mathrm{Prop} : \mathrm{Type}}\ (\omega e\text{-ax})$$

$$\frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^{\omega_e}_\sigma}{\Gamma, x : A, \Gamma' \overset{\omega_e}{\vdash}_\sigma x : A}\ (\omega e\text{-var})$$

$$\frac{\Gamma, x : A \overset{\omega_e}{\vdash}_{\sigma::(x\,\mapsto\,a)} u : B : \mathrm{Prop}}{\Gamma \overset{\omega_e}{\vdash}_\sigma \lambda x^A.u : \forall x^A.B}\ (\omega e\text{-abs}_\star)$$

$$\frac{\Gamma, x : A \overset{\omega_e}{\vdash}_{\sigma::(x\,\mapsto\,a)} u : B : \mathrm{Type} \quad \Gamma \overset{\omega_e}{\vdash}_\sigma A : \mathrm{Type}}{\Gamma \overset{\omega_e}{\vdash}_\sigma \lambda x^A.u : A \to B}\ (\omega e\text{-abs}_\Box)$$

$$\frac{\Gamma \overset{\omega_e}{\vdash}_\sigma u : \forall x^A.B : \mathrm{Prop} \quad \Gamma \overset{\omega_e}{\vdash}_\sigma v : A \quad \sigma\,\mathrm{mot}^{\mathrm{Prop}}(B[x \leftarrow v\,])}{\Gamma \overset{\omega_e}{\vdash}_\sigma u\ v : B[x \leftarrow v\,]}\ (\omega e\text{-app}_\star)$$

$$\frac{\Gamma \overset{\omega_e}{\vdash}_\sigma u : A \to B : \mathrm{Type} \quad \Gamma \overset{\omega_e}{\vdash}_\sigma v : A \quad \sigma\,\mathrm{mot}^{B}(u\ v)}{\Gamma \overset{\omega_e}{\vdash}_\sigma u\ v : B}\ (\omega e\text{-app}_\Box)$$

$$\frac{\Gamma, x : A \vDash^{\omega_e}_{\sigma::(x \mapsto a)} B : \mathrm{Prop} \quad \sigma\ \mathrm{mot}^{\mathrm{Prop}}(\forall x^A.B)}{\Gamma \vDash^{\omega_e}_{\sigma} \forall x^A.B : \mathrm{Prop}}\ (\omega e\text{-prod}_\star)$$

$$\frac{\Gamma, x : A \vDash^{\omega_e}_{\sigma::(x \mapsto a)} B : \mathrm{Type} \quad \Gamma \vDash^{\omega_e}_{\sigma} A : \mathrm{Type}}{\Gamma \vDash^{\omega_e}_{\sigma} A \to B : \mathrm{Type}}\ (\omega e\text{-prod}_\Box)$$

$$\frac{\Gamma \vDash^{\omega_e}_{\sigma} u : A \quad A =_\beta A' \quad \Gamma \vDash^{\omega_e}_{\sigma} A' : \mathrm{Prop}}{\Gamma \vDash^{\omega_e}_{\sigma} u : A'}\ (\omega e\text{-conv})$$

4.2.2 Résultats

Théorème 4.2.1 (λ_e^ω est un sous-système de λ^ω).

(i) si $\Gamma\, \mathsf{wf}^{\omega_e}_{\sigma}$, *alors* $\Gamma\, \mathsf{wf}^{\omega}$ *;*

(ii) si $\Gamma \vDash^{\omega_e}_{\sigma} w : C$, *alors* $\Gamma \vDash^{\omega} w : C$.

Preuve immédiate par récurrence structurelle sur la dérivation : il suffit d'« oublier » les motivations explicites et d'interpréter les constantes o par $\lambda A^{\mathrm{Prop}}.\lambda x^A.x$ et $\top$ par $\forall A^{\mathrm{Prop}}.A \to A$.$\Box$

Lemme 4.2.2 (voir lem. 4.1.1).

(i) Si $x_1 : A_1, \ldots, x_n : A_n\, \mathsf{wf}^{\omega_e}_{(y_1 \mapsto t_1)::\ldots::(y_m \mapsto t_m)}$, *alors :*
- *pour tout* i, $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$ *;*
- *pour tout* $i \neq j$ $x_i \not\equiv x_j$ *;*
- *pour tout* i $x_i \equiv y_i$ *et* t_i *est clos ;*
- $m = n$.

(ii) Si $x_1 : A_1, \ldots, x_n : A_n \vDash^{\omega_e}_{(y_1 \mapsto t_1)::\ldots::(y_m \mapsto t_m)} w : C$, *alors :*
- *pour tout* i, $\mathcal{V}(A_{i+1}) \subseteq \{x_1, \ldots, x_i\}$ *;*
- *pour tout* $i \neq j$ $x_i \not\equiv x_j$ *;*
- $\mathcal{V}(w, C) \subseteq \{x_1, \ldots, x_n\}$ *;*
- *pour tout* i $x_i \equiv y_i$ *et* t_i *est clos ;*
- $m = n$.

Lemme 4.2.3 (voir lem. 4.1.2). *Si* $\Gamma\, \mathsf{wf}^{\omega_e}_{\sigma}$ *ou* $\Gamma \vDash^{\omega_e}_{\sigma} w : C$, *alors* $\mathrm{Type} \notin \mathcal{S}(\Gamma)$ *et* $\mathrm{Type} \notin \mathcal{S}(w)$.

Lemme 4.2.4 (validité des environnements, voir lem. 4.1.3).

(i) si $\Gamma\, \mathsf{wf}^{\omega_e}_{\sigma}$, *alors pour tout environnement* $\Gamma' \preccurlyeq \Gamma$, $\Gamma'\, \mathsf{wf}^{\omega_e}_{\sigma'}$ *en est une sous-dérivation avec* $\sigma' \preccurlyeq \sigma$ *correspondant à* Γ' *;*

(ii) si $\Gamma \vDash^{\omega_e}_{\sigma} w : C$, *alors pour tout environnement* $\Gamma' \preccurlyeq \Gamma$, $\Gamma'\, \mathsf{wf}^{\omega_e}_{\sigma'}$ *en est une sous-dérivation stricte avec* $\sigma' \preccurlyeq \sigma$ *correspondant à* Γ'.

Lemme 4.2.5 (validité des types des environnements, voir lem. 4.1.4).

Si $x_1 : A_1, \ldots, x_n : A_n\, \mathsf{wf}^{\omega_e}_{\sigma}$, *alors pour tout* i *il existe* κ *tel que* $x_1 : A_1, \ldots, x_i : A_i \vDash^{\omega_e}_{\sigma_{\leq i}} A_{i+1} : \kappa$ *en est une sous-dérivation stricte.*

Lemme 4.2.6 (unicité du type, voir lem. 4.1.6).

Si $\Gamma \vDash^{\omega_e}_{\sigma} w : C$ *et* $\Gamma \vDash^{\omega_e}_{\sigma} w : C'$, *alors* $C =_\beta C'$.

Preuve est une conséquence immédiate que λ_e^ω est un sous-système de λ^ω (thm. 4.2.1). □

Lemme 4.2.7 (génération, voir lem. 4.1.8).

Soit $\Gamma \vdash_\sigma^{\omega_e} t : T$*, un de ces cas se présente :*

(i) si $t \equiv o$*, alors* $T =_\beta \top$ *;*

(ii) si $t \equiv \top$*, alors* $T =_\beta \mathrm{Prop}$ *;*

(iii) si $t \equiv \mathrm{Prop}$*, alors* $T =_\beta \mathrm{Type}$ *;*

(iv) si $t \equiv x$*, alors il existe* $(x : A) \in \Gamma$ *avec* $T =_\beta A$ *;*

(v) si $t \equiv \lambda x^A.u$*, alors il existe* a*,* B *et* κ *tels que* $\Gamma, x : A \vdash_{\sigma::(x \mapsto a)}^{\omega_e} u : B : \kappa$ *est une sous-dérivation stricte avec* $T =_\beta \forall x^A.B$ *;*

(vi) si $t \equiv u\ v$*, alors il existe* A *et* B *tels que* $\Gamma \vdash_\sigma^{\omega_e} u : \forall x^A.B$ *et* $\Gamma \vdash_\sigma^{\omega_e} v : A$ *sont des sous-dérivations strictes avec* $T =_\beta B[x \leftarrow v]$ *;*

(vii) si $t \equiv \forall x^A.B$*, alors il existe* a *et* κ *tel que* $\Gamma, x : A \vdash_{\sigma::(x \mapsto a)}^{\omega_e} B : \kappa$ *est une sous-dérivation stricte avec* $T =_\beta \kappa$*.*

Lemme 4.2.8 (voir lem. 4.1.10). *Si* $\Gamma \vdash_\sigma^{\omega_e} \mathcal{O} : C$ *alors* $C \equiv \mathrm{Type}$*.*

Preuve est une conséquence immédiate que λ_e^ω est un sous-système de λ^ω (thm. 4.2.1). □

Lemme 4.2.9 (voir lem. 4.1.11).

(i) Si $\Gamma \vdash_\sigma^{\omega_e} C : \mathrm{Type}$ *alors* C *est un ordre et la dernière règle de la dérivation est (*ω*e-ax) ou (*ω*e-prod*$_\Box$*) ;*

(ii) Si $\Gamma \vdash_\sigma^{\omega_e} C : \mathcal{O}$ *alors la dernière règle de la dérivation est (*ω*e-ax), (*ω*e-var), (*ω*e-abs*$_\Box$*), (*ω*e-app*$_\Box$*) ou (*ω*e-prod*$_\star$*).*

Lemme 4.2.10 (affaiblissement, voir lem. 3.2.13).

Si $\Gamma \vdash_\sigma^{\omega_e} w : C$*,* $\Gamma' \,\mathsf{wf}_{\sigma'}^{\omega_e}$*,* $\Gamma \subseteq \Gamma'$ *et* $\sigma \subseteq \sigma'$*, alors* $\Gamma' \vdash_{\sigma'}^{\omega_e} w : C$*.*

Preuve par récurrence structurelle sur la dérivation ; similaire à celle du lemme 3.2.13.
Il suffit de remarquer que si $\sigma \,\mathrm{mot}^{\mathcal{O}}(C)$ alors $\sigma' \,\mathrm{mot}^{\mathcal{O}}(C)$ avec $\sigma \subseteq \sigma'$. □

Lemme 4.2.11 (voir lem. 2.2.16). *Si* $\Gamma \vdash_\sigma^{\omega_e} C : \mathrm{Type}$ *alors il existe un terme* t *tel que* $\Gamma \vdash_\sigma^{\omega_e} t : C$*.*

Lemme 4.2.12.

$\sigma \,\mathrm{mot}^{\mathcal{O}}(C[z \leftarrow w])$ *si et seulement si* $(z \mapsto w) :: \sigma \,\mathrm{mot}^{\mathcal{O}}(C)$*, avec* w *clos et* $z \notin \mathrm{dom}(\sigma)$*.*

Preuve par récurrence structurelle sur $\mathcal{O}$ comme ordre (utilise lem. 1.2.6 et lem. 4.2.2). □

Lemme 4.2.13 (voir prop. 3.2.11).

Si $x_1 : A_1, \ldots, x_n : A_n \,\mathsf{wf}_\sigma^{\omega_e}$*, alors*

$$\forall i \quad \vdash_{[]}^{\omega_e} \sigma(x_i) : A'_i \quad \textit{avec} \quad \sigma(A_i) \overset{*}{\rightsquigarrow}_\beta A'_i$$

en ***sont des sous-dérivations strictes****.*

Preuve similaire à celle de la proposition 3.2.11. □

Lemme 4.2.14 (voir lem. 3.2.16).

(i) Si $z : C, \Gamma\, \mathsf{wf}^{\omega_e}_{(z \mapsto c)::\sigma}$ *alors* $\Gamma[z \leftarrow c]\, \mathsf{wf}^{\omega_e}_{\sigma}$*;*

(ii) Si $z : C, \Gamma \vdash^{\omega_e}_{(z \mapsto c)::\sigma} w : D$ *alors* $\Gamma[z \leftarrow c] \vdash^{\omega_e}_{\sigma} w[z \leftarrow c] : D[z \leftarrow c]$.

Preuve par récurrence structurelle sur la dérivation ; similaire à celle du lemme 3.2.16 (utilise lem. 4.2.12).

(ωe-var) Le traitement de ce cas est légèrement différent de celui effectué pour λ_e^2 :

$$\frac{z : C, \Gamma\, \mathsf{wf}^{\omega_e}_{(z \mapsto c)::\sigma}}{z : C, \Gamma \vdash^{\omega_e}_{(z \mapsto c)::\sigma} z : C}$$

Par hypothèse de récurrence, on a $\Gamma[z \leftarrow c]\, \mathsf{wf}^{\omega_e}_{\sigma}$.

Aussi $\vdash^{\omega_e}_{[]} c : C'$ avec $C \overset{*}{\rightsquigarrow}_\beta C'$ (lem. 4.2.13). Or $\vdash^{\omega_e}_{[]} C : \kappa$ (lem. 4.2.5) et on a deux cas :

- $\kappa \equiv \mathrm{Type}$: C est un ordre et est irréductible, donc $C \equiv C'$ et $\vdash^{\omega_e}_{[]} c : C$;
- $\kappa \equiv \mathrm{Prop}$: on peut appliquer la règle (ωe-conv) pour obtenir aussi $\vdash^{\omega_e}_{[]} c : C$.

Enfin par affaiblissement (lem. 4.2.10) on obtient bien $\Gamma[z \leftarrow c] \vdash^{\omega_e}_{\sigma} c : C$. □

Lemme 4.2.15 (voir lem. 3.2.17). *Si* $\Gamma \vdash^{\omega_e}_{\sigma} w : C$ *alors* $\vdash^{\omega_e}_{[]} \sigma(w) : \sigma(C)$.

Proposition 4.2.16 (λ_e^ω vérifie le critère de Poincaré, voir prop. 3.2.11).

Si $x_1 : A_1, \ldots, x_n : A_n\, \mathsf{wf}^{\omega_e}_{\sigma}$*, alors*

$$\forall i \quad \vdash^{\omega_e}_{[]} \sigma(x_i) : \sigma(A_i)$$

Preuve Pour tout i, on a (lem.4.2.13) :

$$\vdash^{\omega_e}_{[]} \sigma(x_i) : A'_i \quad \text{avec} \quad \sigma(A_i) \overset{*}{\rightsquigarrow}_\beta A'_i$$

Or $x_1 : A_1, \ldots, x_{i-1} : A_{i-1} \vdash^{\omega_e}_{\sigma_{\leq i-1}} A_i : \kappa$ (lem. 4.2.5) :

- $\kappa \equiv \mathrm{Type}$: A_i est un ordre donc est irréductible et $\sigma(A_i) \equiv A'_i$ d'où $\vdash^{\omega_e}_{[]} \sigma(x_i) : \sigma(A_i)$;
- $\kappa \equiv \mathrm{Prop}$: alors en transmettant la motivation en conclusion (lem. 4.2.14) on obtient $\vdash^{\omega_e}_{[]} \sigma(A_i) : \mathrm{Prop}$, et finalement par ($\omega e$-conv) on a bien $\vdash^{\omega_e}_{[]} \sigma(x_i) : \sigma(A_i)$. □

Lemme 4.2.17 (voir lem. 3.2.12). *Si* $\Gamma \vdash^{\omega_e}_{\sigma} C : \kappa$*, alors il existe un terme* t *et un type* C' *tels que* $\vdash^{\omega_e}_{[]} t : C'$ *avec* $\sigma(C) \overset{*}{\rightsquigarrow}_\beta C'$.

Preuve par récurrence structurelle sur la dérivation ; similaire à celle du lemme 3.2.12. Toutes les règles ne sont pas à considérer (lem. 4.2.9) :

$$(\omega e\text{-}\mathbf{app}_\square)\ \frac{\Gamma \vdash^{\omega_e}_{\sigma} u : A \to B : \mathrm{Type} \quad \Gamma \vdash^{\omega_e}_{\sigma} v : A \quad \sigma\, \mathrm{mot}^B(u\ v)}{\Gamma \vdash^{\omega_e}_{\sigma} u\ v : B}$$

Comme $B \equiv \mathrm{Prop}$ (lem. 4.2.3) alors par définition de $\sigma\, \mathrm{mot}^B(u\ v)$ on a le résultat.

$$(\omega e\text{-}\mathbf{prod}_\square)\ \frac{\Gamma, x : A \vdash^{\omega_e}_{\sigma} B : \mathrm{Type} \quad \Gamma \vdash^{\omega_e}_{\sigma} A : \mathrm{Type}}{\Gamma \vdash^{\omega_e}_{\sigma} A \to B : \mathrm{Type}}$$

On a t tel que $\Gamma \vdash^{\omega_e}_{\sigma} t : A \to B$ (lem. 4.2.11), et donc $\vdash^{\omega_e}_{[]} \sigma(t) : \sigma(A \to B)$ (lem. 4.2.15). □

Lemme 4.2.18. *Si* $\Gamma\, \mathsf{wf}^{\omega_e}_{\sigma}$*, alors pour tout ordre* $\mathcal{O}$ *on a* $\Gamma \vdash^{\omega_e}_{\sigma} \mathcal{O} : \mathrm{Type}$.

Preuve par récurrence structurelle sur $\mathcal{O}$ en tant qu'ordre.

Quand $\mathcal{O} \equiv \mathcal{O}_1 \to \mathcal{O}_2$, l'hypothèse de récurrence donne $\Gamma \vdash^{\omega_e}_{\sigma} \mathcal{O}_1 : \text{Type}$ et $\Gamma \vdash^{\omega_e}_{\sigma} \mathcal{O}_2 : \text{Type}$. On a alors un terme t tel que $\vdash^{\omega_e}_{[]} t : \mathcal{O}_1$ (lem. 4.2.11 et 4.2.15) et donc $\Gamma, z : \mathcal{O}_1 \, \mathsf{wf}^{\omega_e}_{\sigma::(z \mapsto t)}$ par (ωe-env$_2$). Par affaiblissement (lem. 4.2.10) on a alors $\Gamma, z : \mathcal{O}_1 \vdash^{\omega_e}_{\sigma::(z \mapsto t)} \mathcal{O}_2 : \text{Type}$ et finalement par (ωe-prod$_\Box$) on a bien $\Gamma \vdash^{\omega_e}_{\sigma} \mathcal{O}_1 \to \mathcal{O}_2 : \text{Type}$. □

Lemme 4.2.19 (voir lem. 3.2.14). *Si* $\vdash^{\omega_e}_{[]} w : C : \kappa$ *et* $z \notin \text{dom}(\Gamma)$ *:*
(i) si $\Gamma[z \leftarrow w] \, \mathsf{wf}^{\omega_e}_{\sigma}$ *alors* $z : C, \Gamma \, \mathsf{wf}^{\omega_e}_{(z \mapsto w)::\sigma}$ *;*
(ii) si $\Gamma[z \leftarrow w] \vdash^{\omega_e}_{\sigma} D[z \leftarrow w] : \mathcal{O}$ *alors* $z : C, \Gamma \vdash^{\omega_e}_{(z \mapsto w)::\sigma} D : \mathcal{O}$ *;*
(iii) si $\Gamma[z \leftarrow w] \vdash^{\omega_e}_{\sigma} D[z \leftarrow w] : \text{Type}$ *alors* $z : C, \Gamma \vdash^{\omega_e}_{(z \mapsto w)::\sigma} D : \text{Type}$.

Preuve par récurrence structurelle sur la dérivation ; similaire à celle du lemme 3.2.14.

Pour (ii) on traite le cas où $D \equiv z$ de la même façon que pour λ^2_e, mais en présence de la règle de conversion (ωe-conv), on aboutit à

$$z : C, \Gamma \vdash^{\omega_e}_{(z \mapsto w)::\sigma} z : C \quad \text{avec} \quad C =_\beta \mathcal{O}$$

Or comme $z : C, \Gamma \, \mathsf{wf}^{\omega_e}_{(z \mapsto w)::\sigma}$ (lem. 4.2.4) alors $z : C, \Gamma \vdash^{\omega_e}_{(z \mapsto w)::\sigma} \mathcal{O} : \text{Type}$ (lem. 4.2.18) et (ωe-conv) donne finalement

$$z : C, \Gamma \vdash^{\omega_e}_{(z \mapsto w)::\sigma} z : \mathcal{O}$$

Remarquons aussi qu'on n'a jamais le cas (iii) avec $D \equiv z$ car sinon w est un ordre (lem. 4.2.9) et alors $C \equiv \text{Type}$ (lem. 4.2.8), ce qui n'est pas possible (lem. 4.2.3).

(ωe-abs$_\Box$)

$$\frac{\Gamma[z \leftarrow w], x : A[z \leftarrow w] \vdash^{\omega_e}_{\sigma::(x \mapsto a)} u[z \leftarrow w] : B : \text{Type} \quad \Gamma[z \leftarrow w] \vdash^{\omega_e}_{\sigma} A[z \leftarrow w] : \text{Type}}{\Gamma[z \leftarrow w] \vdash^{\omega_e}_{\sigma} \lambda x^{A[z \leftarrow w]}.u[z \leftarrow w] : A[z \leftarrow w] \to B}$$

avec B un ordre par hypothèse.

L'hypothèse de récurrence sur les prémisses donne

$$z : C, \Gamma, x : A \vdash^{\omega_e}_{(z \mapsto w)::\sigma::(x \mapsto a)} u : B : \text{Type} \quad \text{et} \quad z : C, \Gamma \vdash^{\omega_e}_{(z \mapsto w)::\sigma} A : \text{Type}$$

et (ωe-abs$_\Box$) le résultat.

(ωe-app$_\Box$)

$$\frac{\Gamma[z \leftarrow w] \vdash^{\omega_e}_{\sigma} u[z \leftarrow w] : A \to B : \text{Type} \quad \Gamma[z \leftarrow w] \vdash^{\omega_e}_{\sigma} v[z \leftarrow w] : A \quad \sigma \, \text{mot}^B((u \; v)[z \leftarrow w])}{\Gamma[z \leftarrow w] \vdash^{\omega_e}_{\sigma} (u \; v)[z \leftarrow w] : B}$$

Comme $A \to B$ est un ordre (lem. 4.2.9), par hypothèse de récurrence

$$z : C, \Gamma \vdash^{\omega_e}_{(z \mapsto w)::\sigma} u : A \to B : \text{Type} \quad \text{et} \quad z : C, \Gamma \vdash^{\omega_e}_{(z \mapsto w)::\sigma} v : A$$

et comme de plus (lem. 4.2.12)

$$(z \mapsto w)::\sigma \, \text{mot}^B(u \; v)$$

alors (ωe-app$_\Box$) permet de conclure.

(ωe-prod$_\star$, ωe-prod$_\Box$) de même que précédemment. □

Proposition 4.2.20 (λ^ω_e vérifie la réciproque du critère de Poincaré).

Si

$$\vdash^{\omega_e}_{[]} t_1 : A_1 : \kappa_1 \quad \vdash^{\omega_e}_{[]} t_2 : A_2[x_1 \leftarrow t_1\,] : \kappa_2 \quad \ldots \quad \vdash^{\omega_e}_{[]} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}\,] : \kappa_n$$

(avec les x_i distinctes deux à deux), alors

$$x_1 : A_1, x_2 : A_2, \ldots, x_n : A_n \,\mathsf{wf}^{\omega_e}_{(x_1 \mapsto t_1)::(x_2 \mapsto t_2)::\ \ldots\ ::(x_n \mapsto t_n)}$$

Preuve similaire à celle du lemme 3.2.15 pour λ_e^2 ; utilise le lemme 4.2.19. □

Conjecture 4.2.21 (de substitution).

(i) si $\Gamma, y : C, \Gamma' \vdash^{\omega_e}_{\sigma::(y \mapsto c)::\sigma'} D : \mathcal{O}$, $\Gamma \vdash^{\omega_e}_{\sigma} w : C$, $\Gamma, \Gamma'[y \leftarrow w\,]\,\mathsf{wf}^{\omega_e}_{\sigma::\rho}$ *et* $\sigma::\rho\,\mathrm{mot}^{\mathcal{O}}(D[y \leftarrow w\,])$ *alors* $\Gamma, \Gamma'[y \leftarrow w\,] \vdash^{\omega_e}_{\sigma::\rho} D[y \leftarrow w\,] : \mathcal{O}$;

(ii) si $\Gamma, y : C, \Gamma' \vdash^{\omega_e}_{\sigma::(y \mapsto c)::\sigma'} d : D : \mathrm{Prop}$, $\Gamma \vdash^{\omega_e}_{\sigma} w : C$, $\Gamma, \Gamma'[y \leftarrow w\,]\,\mathsf{wf}^{\omega_e}_{\sigma::\rho}$ *et* $\sigma::\rho\,\mathrm{mot}^{\mathrm{Prop}}(D[y \leftarrow w\,])$ *alors* $\Gamma, \Gamma'[y \leftarrow w\,] \vdash^{\omega_e}_{\sigma::\rho} d[y \leftarrow w\,] : D[y \leftarrow w\,]$;

Proposition 4.2.22 (stabilité par réduction).

(i) Si $\Gamma\,\mathsf{wf}^{\omega_e}_{\sigma}$ *et* $\Gamma \rightsquigarrow_\beta \Gamma'$, *alors* $\Gamma'\,\mathsf{wf}^{\omega_e}_{\sigma}$;

(ii) Si $\Gamma \vdash^{\omega_e}_{\sigma} w : C$ *et* $\Gamma \rightsquigarrow_\beta \Gamma'$, *alors* $\Gamma' \vdash^{\omega_e}_{\sigma} w : C$;

(iii) Si $\Gamma \vdash^{\omega_e}_{\sigma} w : C$ *et* $w \rightsquigarrow_\beta w'$, *alors* $\Gamma \vdash^{\omega_e}_{\sigma} w' : C$;

(iv) Si $\Gamma \vdash^{\omega_e}_{\sigma} w : C$ *et* $C \rightsquigarrow_\beta C'$, *alors* $\Gamma \vdash^{\omega_e}_{\sigma} w : C'$;

(v) Si $\Gamma \vdash^{\omega_e}_{\sigma} P : \mathcal{O}$, $P \rightsquigarrow_\beta P'$ *et* $\sigma\,\mathrm{mot}^{\mathcal{O}}(P)$, *alors* $\sigma\,\mathrm{mot}^{\mathcal{O}}(P')$.

Preuve par récurrence structurelle sur la dérivation ; puis pour (iii) par cas sur la définition de $\rightsquigarrow_\beta$ (similaire à celle du lem. 1.4.10) ; et pour (v) par récurrence structurelle sur $\mathcal{O}$ en tant qu'ordre (non traité ici).

$$(\omega e\textbf{-var})\ \frac{\Gamma, x : A, \Gamma'\,\mathsf{wf}^{\omega_e}_{\sigma}}{\Gamma, x : A, \Gamma' \vdash^{\omega_e}_{\sigma} x : A} \quad \text{avec } A \rightsquigarrow_\beta A'.$$

$\Gamma \vdash^{\omega_e}_{\sigma_1} A : \kappa$ est une sous-dérivation stricte (lem. 4.2.5) avec $\sigma_1 \preccurlyeq \sigma$ correspondant à Γ. De plus, nécessairement $\kappa \equiv \mathrm{Prop}$ sinon A serait un ordre et donc irréductible.

- Par hypothèse de récurrence on a $\Gamma, x : A', \Gamma'\,\mathsf{wf}^{\omega_e}_{\sigma}$, et donc par ($\omega e$-var) $\Gamma, x : A', \Gamma' \vdash^{\omega_e}_{\sigma} x : A'$. Par affaiblissement (lem. 4.2.10) $\Gamma, x : A', \Gamma' \vdash^{\omega_e}_{\sigma} A : \mathrm{Prop}$ et par (ωe-conv) $\Gamma, x : A', \Gamma' \vdash^{\omega_e}_{\sigma} x : A$.
- Par hypothèse de récurrence $\Gamma \vdash^{\omega_e}_{\sigma_1} A' : \kappa$ puis par affaiblissement (lem. 4.2.10) on obtient $\Gamma, x : A, \Gamma' \vdash^{\omega_e}_{\sigma} A' : \mathrm{Prop}$ d'où (ωe-conv) nous donne $\Gamma, x : A, \Gamma' \vdash^{\omega_e}_{\sigma} x : A'$.

$$(\omega e\textbf{-app}_\star)\ \frac{\Gamma \vdash^{\omega_e}_{\sigma} u : \forall x^A.B : \mathrm{Prop} \quad \Gamma \vdash^{\omega_e}_{\sigma} v : A \quad \sigma\,\mathrm{mot}^{\mathrm{Prop}}(B[x \leftarrow v\,])}{\Gamma \vdash^{\omega_e}_{\sigma} u\ v : B[x \leftarrow v\,]}$$

On a trois cas :

- $u \rightsquigarrow_\beta u'$: immédiat par hypothèse de récurrence et (ωe-app$_\star$).
- $v \rightsquigarrow_\beta v'$. Par hypothèse de récurrence on a $\Gamma \vdash^{\omega_e}_{\sigma} v' : A$, puis par génération (lem. 4.2.7) et substitution (lem. 4.2.21), on a

$$\Gamma \vdash^{\omega_e}_{\sigma} B[x \leftarrow v\,] : \kappa \qquad \Gamma \vdash^{\omega_e}_{\sigma} B[x \leftarrow v'\,] : \kappa$$

et donc aussi $\vdash^{\omega_e}_{[]} \sigma(B[x \leftarrow v'\,]) : \kappa$ (lem. 4.2.15).

De plus, comme $v =_\beta v'$ alors $B[x \leftarrow v] =_\beta B[x \leftarrow v']$ et $\sigma(B[x \leftarrow v]) =_\beta \sigma(B[x \leftarrow v'])$ (lem. 1.2.8). D'où de $\Vdash^{\omega_e}_{[]} t : Q$ avec $\sigma(B[x \leftarrow v]) \overset{*}{\rightsquigarrow}_\beta Q$ par (ωe-conv) on a $\Vdash^{\omega_e}_{[]} t : \sigma(B[x \leftarrow v'])$, i.e. $\sigma\,\mathrm{mot}^{\mathrm{Prop}}(B[x \leftarrow v'])$. On peut alors appliquer (ωe-app$_\star$) pour obtenir

$$\Gamma \Vdash^{\omega_e}_{\sigma} u\ v' : B[x \leftarrow v']$$

puis par (ωe-conv)

$$\Gamma \Vdash^{\omega_e}_{\sigma} u\ v' : B[x \leftarrow v]$$

- $u \equiv \lambda x^C.w$ et $u\ v \rightsquigarrow_\beta w[x \leftarrow v]$: se traite de façon usuelle (voir thm. 1.4.11) en utilisant les lemmes de génération (lem. 4.2.7) et de substitution (lem. 4.2.21).

$$(\omega e\textbf{-app}_\square)\ \frac{\Gamma \Vdash^{\omega_e}_{\sigma} u : A \to B : \mathrm{Type} \quad \Gamma \Vdash^{\omega_e}_{\sigma} v : A \quad \sigma\,\mathrm{mot}^B(u\ v)}{\Gamma \Vdash^{\omega_e}_{\sigma} u\ v : B} \quad \text{avec } B \text{ un ordre (lem. 4.2.9)}$$

donc irréductible.

Posons $B \equiv \mathcal{O}_1 \to \ldots \to \mathcal{O}_n \to \mathrm{Prop}$ avec les $\mathcal{O}_i$ des ordres.

On a trois cas :

- $u \rightsquigarrow_\beta u'$: comme alors $u\ v \rightsquigarrow_\beta u'\ v$ il suffit d'appliquer l'hypothèse de récurrence (cas (v)).
- $v \rightsquigarrow_\beta v'$: similaire au cas précédent.
- $u \equiv \lambda x^C.w$ et $u\ v \rightsquigarrow_\beta w[x \leftarrow v]$: se traite de façon usuelle (voir thm. 1.4.11).

$$(\omega e\textbf{-prod}_\star)\ \frac{\Gamma, x : A \Vdash^{\omega_e}_{\sigma::(x \mapsto a)} B : \mathrm{Prop} \quad \sigma\,\mathrm{mot}^{\mathrm{Prop}}(\forall x^A.B)}{\Gamma \Vdash^{\omega_e}_{\sigma} \forall x^A.B : \mathrm{Prop}} \quad \text{avec } A \rightsquigarrow_\beta A'.$$

Par hypothèse de récurrence $\Gamma, x : A' \Vdash^{\omega_e}_{\sigma::(x \mapsto a)} B : \mathrm{Prop}$, et puisque $\forall x^A.B \rightsquigarrow_\beta \forall x^{A'}.B$ aussi $\sigma\,\mathrm{mot}^{\mathrm{Prop}}(\forall x^{A'}.B)$ d'où (ωe-prod$_\star$) permet de conclure.

De même si $B \rightsquigarrow_\beta B'$.

$$(\omega e\textbf{-conv})\ \frac{\Gamma \Vdash^{\omega_e}_{\sigma} u : A \quad A =_\beta A' \quad \Gamma \Vdash^{\omega_e}_{\sigma} A' : \mathrm{Prop}}{\Gamma \Vdash^{\omega_e}_{\sigma} u : A'} \quad \text{avec } A' \rightsquigarrow_\beta A''.$$

Par hypothèse de récurrence $\Gamma \Vdash^{\omega_e}_{\sigma} A'' : \mathrm{Prop}$ et comme $A =_\beta A''$, alors par (ωe-conv) sur la première prémisse $\Gamma \Vdash^{\omega_e}_{\sigma} u : A''$. □

Lemme 4.2.23 (correction des types, voir lem. 3.2.27).

Si $\Gamma \Vdash^{\omega_e}_{\sigma} w : C$ alors $C \equiv \mathrm{Type}$ ou bien il existe κ tel que $\Gamma \Vdash^{\omega_e}_{\sigma} C : \kappa$.

Preuve par récurrence structurelle sur la dérivation (similaire à lem. 4.2.23). □

Proposition 4.2.24 (λ^ω_e est un pseudo sous-système pédagogique de CC).

λ^ω_e vérifie les propriétés suivantes :

(i) λ^ω_e est un sous-système de CC ;

(ii) Si $\Gamma \Vdash^{\omega_e}_{\sigma} t : C$ et $t \rightsquigarrow_\beta t'$, alors $\Gamma \Vdash^{\omega_e}_{\sigma} t' : C$.

*(iii) $x_1 : A_1, \ldots, x_n : A_n\ \mathsf{wf}^{\omega_e}_{(x_1 \mapsto t_1)::\ldots::(x_n \mapsto t_n)}$ **si et seulement si** $x_1 : A_1, \ldots, x_n : A_n\ \mathsf{wf}^{\mathsf{c}}$ et il existe des termes $t_1, \ldots, t_n$ tels que*

$$\Vdash^{\omega_e}_{[]} t_1 : A_1 \qquad \Vdash^{\omega_e}_{[]} t_2 : A_2[x_1 \leftarrow t_1] \quad \ldots \quad \Vdash^{\omega_e}_{[]} t_n : A_n[x_1, \ldots, x_{n-1} \leftarrow t_1, \ldots, t_{n-1}]$$

Preuve

(i) λ_e^ω est un sous-système de λ^ω (thm. 4.2.1) lui-même sous-système de CC (thm. 4.1.12).

(ii) C'est exactement la proposition 4.2.22.

(iii) Ce cas est similaire à celui pour λ_e^2 (prop. 3.2.28) :

$\boxed{\Rightarrow}$ C'est exactement la proposition 4.2.16.

$\boxed{\Leftarrow}$ Similaire à λ_e^2 en utilisant la proposition 4.2.20 et le lemme 4.2.23. □

Troisième partie

Annexes

Annexe A

Quelques résultats d'intérêt

A.1 Avec motivations totales explicites – λ_e^2

Motivation triviale

◊ Une substitution τ est une *motivation triviale* pour l'environnement Γ si $\Gamma\,\mathsf{wf}^{2_e}_{\tau}$ et pour tout $(x : \mathrm{Prop}) \in \Gamma$ on a $\tau(x) \equiv \top$.

Lemme A.1.1. *Si $\Gamma\,\mathsf{wf}^{2_e}_{\sigma}$ ou $\Gamma \vdash^{2_e}_{\sigma} w : C$, alors il existe une motivation triviale τ pour Γ.*

Preuve par récurrence structurelle sur la dérivation.

$$\textbf{(e-env}_{\mathbf{2}}\textbf{)}\ \frac{\Gamma \vdash^{2_e}_{\sigma} A : \kappa \quad \vdash^{2_e}_{[]} a : \sigma(A) \quad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A\,\mathsf{wf}^{2_e}_{\sigma::(x \mapsto a)}}$$

Déjà, l'hypothèse de récurrence nous donne une motivation triviale τ' de Γ.

On doit traiter deux cas, selon la valeur de κ.

- $\kappa \equiv \mathrm{Type}$: alors $A \equiv \mathrm{Prop}$ (lem. 3.2.8) et $\tau := \tau'::(x \mapsto \top)$ convient.
- $\kappa \equiv \mathrm{Prop}$: alors par échange de motivations $\Gamma \vdash^{2_e}_{\tau'} A : \kappa$ (lem. 3.2.24) d'où on a a' tel que $\vdash^{2_e}_{[]} a' : \tau'(A)$ et donc $\tau := \tau'::(x \mapsto a')$ convient.

(*) Pour toutes les autres règles, on a que $\Gamma\,\mathsf{wf}^{2_e}_{\sigma}$ est une sous-dérivation stricte (lem. 3.2.4), sur laquelle il suffit d'appliquer l'hypothèse de récurrence. □

Lemme A.1.2.

(i) Si $\Gamma, z : C, \Gamma'\,\mathsf{wf}^{2_e}_{\sigma::(z \mapsto c)::\sigma'}$ et $\vdash^{2_e}_{[]} c' : \sigma(C)$, alors il existe une substitution ρ telle qu'on ait $\Gamma, z : C, \Gamma'\,\mathsf{wf}^{2_e}_{\sigma::(z \mapsto c')::\rho}$;

(ii) Si $\Gamma, z : C, \Gamma' \vdash^{2_e}_{\sigma::(z \mapsto c)::\sigma'} d : D$ et $\vdash^{2_e}_{[]} c' : \sigma(C)$, alors il existe une substitution ρ telle que $\Gamma, z : C, \Gamma' \vdash^{2_e}_{\sigma::(z \mapsto c')::\rho} d : D$.

Preuve par récurrence structurelle sur la dérivation (pour (i)).

Le (ii) s'obtient à partir de (i) : on a $\Gamma, z : C, \Gamma'\,\mathsf{wf}^{2_e}_{\sigma::(z \mapsto c)::\sigma'}$ (lem. 3.2.4) sur laquelle on applique (i) et par échange des motivations (lem. 3.2.24) on obtient le résultat.

(e-env$_2$) Il faut traiter deux cas selon que Γ' est vide ou non :

- $$\frac{\Gamma \vdash^{2_e}_{\sigma} C : \kappa \quad \vdash^{2_e}_{[]} c : \sigma(C) \quad z \notin \mathrm{dom}(\Gamma)}{\Gamma, z : C\, \mathsf{wf}^{2_e}_{\sigma::(z \mapsto c)}}$$

  Ce cas est immédiat par (e-env$_2$) et l'hypothèse.

- $$\frac{\Gamma, z : C, \Gamma'' \vdash^{2_e}_{\sigma::(z \mapsto c)::\sigma''} A : \kappa \quad \vdash^{2_e}_{[]} a : \sigma::(z \mapsto c)::\sigma''(A) \quad x \notin \mathrm{dom}(\Gamma, z, \Gamma'')}{\Gamma, z : C, \Gamma'', x : A\, \mathsf{wf}^{2_e}_{\sigma::(z \mapsto c)::\sigma''::(x \mapsto a)}}$$

  Comme $\Gamma, z : C, \Gamma''\, \mathsf{wf}^{2_e}_{\sigma::(z \mapsto c)::\sigma''}$ est uns sous-dérivation stricte (lem. 3.2.4), alors par hypothèse de récurrence on a $\Gamma, z : C, \Gamma''\, \mathsf{wf}^{2_e}_{\sigma::(z \mapsto c')::\rho}$ puis par échange de motivation (lem. 3.2.24) $\Gamma, z : C, \Gamma'' \vdash^{2_e}_{\sigma::(z \mapsto c')::\rho} A : \kappa$, d'où $\vdash^{2_e}_{[]} a' : \sigma::(z \mapsto c')::\rho(A)$ (lem. 3.2.12) et finalement le résultat par (e-env$_2$). □

A.2 Avec motivations totales – λ^2_t

Hauteur d'une dérivation

◊ On note par $\mathcal{H}(\Gamma \vdash^{2_t} \Gamma u : A)$ la *hauteur* de l'arbre de dérivation considéré de racine $\Gamma \vdash^{2_t} \Gamma u : A$.

◊ On étend cette notation à un ensemble fini de dérivations : la hauteur de l'ensemble vide est fixé à 0 par convention.

Proposition A.2.1 (λ^2_t vérifie le critère de Poincaré).

(i) si $\Gamma\, \mathsf{wf}^{2_t}$ alors il existe une substitution σ qui motive Γ avec $\mathcal{H}(\sigma \,\mathrm{mot}\, \Gamma) \leq \mathcal{H}(\Gamma\, \mathsf{wf}^{2_t})$.

(ii) si $\Gamma \vdash^{2_t} C : \kappa$, alors il existe une substitution σ qui motive C sous Γ avec $\mathcal{H}(\sigma \,\mathrm{mot}_\Gamma\, C) \leq \mathcal{H}(\Gamma \vdash^{2_t} C : \kappa)$.

Preuve par récurrence forte sur la hauteur de la dérivation puis analyse par cas sur la dernière règle utilisée.

(t-env$_1$) $\sigma := [\,]$ convient.

(t-env$_2$) $$\frac{\Gamma \vdash^{2_t} A : \kappa \quad x \notin \mathcal{V}(\Gamma)}{\Gamma, x : A\, \mathsf{wf}^{2_t}}$$

Par hypothèse de récurrence on a σ telle que $\sigma \,\mathrm{mot}_\Gamma\, A$ avec

$$\mathcal{H}(\sigma \,\mathrm{mot}_\Gamma\, A) \leq \mathcal{H}(\Gamma \vdash^{2_t} A : \kappa)$$

On construit alors $\sigma' := \sigma::(x \mapsto t)$ (le t de $\sigma \,\mathrm{mot}_\Gamma\, A$) et on a bien $\sigma' \,\mathrm{mot}\, \Gamma, x : A$ avec

$$\mathcal{H}(\sigma' \,\mathrm{mot}\, \Gamma, x : A) = \mathcal{H}(\sigma \,\mathrm{mot}_\Gamma\, A) \leq \mathcal{H}(\Gamma \vdash^{2_t} A : \kappa) < \mathcal{H}(\Gamma, x : A\, \mathsf{wf}^{2_t})$$

(t-ax) $$\frac{\Gamma\, \mathsf{wf}^{2_t}}{\Gamma \vdash^{2_t} \top : \mathrm{Prop}}$$

Par hypothèse de récurrence on a σ telle que $\sigma \,\mathrm{mot}\, \Gamma$ avec

$$\mathcal{H}(\sigma \,\mathrm{mot}\, \Gamma) \leq \mathcal{H}(\Gamma\, \mathsf{wf}^{2_t}) < \mathcal{H}(\Gamma \vdash^{2_t} \top : \mathrm{Prop})$$

On construit la dérivation $\dfrac{[\,]\,\mathsf{wf}^{2_t}}{\vdash^{2_t} o : \top}$ (t-ax) avec

$$\mathcal{H}(\vdash^{2_t} o : \top) \leq \mathcal{H}(\Gamma \vdash^{2_t} \top : \mathrm{Prop})$$

donc σ et o conviennent pour avoir $\sigma\,\mathrm{mot}_\Gamma\,\top$.

De même pour $\dfrac{\Gamma\,\mathsf{wf}^{2_t}}{\Gamma \vdash^{2_t} \mathrm{Prop} : \mathrm{Type}}$

(t-var) $\dfrac{\Gamma, x : \mathrm{Prop}, \Gamma'\,\mathsf{wf}^{2_t}}{\Gamma, x : \mathrm{Prop}, \Gamma' \vdash^{2_t} x : \mathrm{Prop}}$

Par hypothèse de récurrence on a σ telle que $\sigma\,\mathrm{mot}(\Gamma, x : \mathrm{Prop}, \Gamma')$ avec

$$\mathcal{H}(\sigma\,\mathrm{mot}(\Gamma, x : \mathrm{Prop}, \Gamma')) \leq \mathcal{H}(\Gamma, x : \mathrm{Prop}, \Gamma'\,\mathsf{wf}^{2_t}) < \mathcal{H}(\Gamma, x : \mathrm{Prop}, \Gamma' \vdash^{2_t} x : \mathrm{Prop})$$

Par définition, on a alors $\vdash^{2_t} \sigma(x) : \mathrm{Prop}$ avec

$$\mathcal{H}(\vdash^{2_t} \sigma(x) : \mathrm{Prop}) \leq \mathcal{H}(\sigma\,\mathrm{mot}(\Gamma, x : \mathrm{Prop}, \Gamma')) < \mathcal{H}(\Gamma, x : \mathrm{Prop}, \Gamma' \vdash^{2_t} x : \mathrm{Prop})$$

et on peut appliquer l'hypothèse de récurrence pour obtenir un terme t vérifiant $\vdash^{2_t} t : \sigma(x)$ et

$$\mathcal{H}(\vdash^{2_t} t : \sigma(x)) \leq \mathcal{H}(\vdash^{2_t} \sigma(x) : \mathrm{Prop}) < \mathcal{H}(\Gamma, x : \mathrm{Prop}, \Gamma' \vdash^{2_t} x : \mathrm{Prop})$$

Finalement on a bien $\sigma\,\mathrm{mot}_{\Gamma, x:\mathrm{Prop}, \Gamma'}\,x$ avec

$$\mathcal{H}(\sigma\,\mathrm{mot}_{\Gamma, x:\mathrm{Prop}, \Gamma'}\,x) \leq \mathcal{H}(\Gamma, x : \mathrm{Prop}, \Gamma' \vdash^{2_t} x : \mathrm{Prop})$$

(t-app) N'arrive jamais (3.3.5).

(t-prod) Cas immédiat par la prémisse. □

Remarque (Note sur la formulation du théorème)

Dans le cas (t-ax), on construit une motivation $\sigma\,\mathrm{mot}_\Gamma\,\top$ qui n'est pas en général une sous-dérivation de $\Gamma \vdash^{2_t} \top : \mathrm{Prop}$, d'où une récurrence sur la hauteur plutôt qu'une récurrence structurelle sur la dérivation.

Remarque

À partir de là on peut démontrer de façon directe que λ_t^2 vérifie la réciproque du critère de Poincaré (similaire à la preuve pour λ_e^2 : voir lem. 3.2.14 et prop. 3.2.15).

Bibliographie

Andler, D. et Martin, R. (1984). Logique mathématique. *Encyclopædia Universalis*, 14.

Avigad, J. et Feferman, S. (1998). *The Handbook of Proof Theory*, chapitre Gödel's functional ("Dialectica") interpretation, pages 337–405. North-Holland.

Barendregt, H. (1991). Introduction to generalized type systems. *Journal of Functional Programming*, 1(2):125–154.

Barendregt, H. (1992). *Lambda calculi with types*, volume 2 de *Handbook of Logic in Computer Science*, pages 117–309. Oxford University Press.

Barras, B. (1996). Coq en coq. Rapport de Recherche 3026, INRIA.

Böhm, C. et Berarducci, A. (1985). Automatic synthesis of typed λ-programs on term algebras. *Theoretical Computer Science*, 39(0):135 – 154.

Church, A. (1933). A set of postulates for the foundation of logic. *The Annals of Mathematics*, 34(4):pp. 839–864.

Colson, L. (1986). Quelques remarques sur l'environnement dans la Théorie Intuitionniste des Types. Mémoire de D.E.A., Université Paris Sud – Orsay.

Colson, L. et Michel, D. (2007). Pedagogical natural deduction systems : the propositional case. *Journal of Universal Computer Science*, 13(10):1396–1410.

Colson, L. et Michel, D. (2008). Pedagogical Second-order Propositional Calculi. *Journal of Logic and Computation*, 18(4):669–695.

Colson, L. et Michel, D. (2009). Pedagogical second-order λ-calculus. *Theoretical Computer Science*, 410:4190–4203.

Coquand, T. (1985). *Une théorie des constructions*. Thèse de doctorat, Université Paris VII.

Coquand, T. et Huet, G. P. (1985). Concepts mathématiques et informatiques formalisés dans le calcul des constructions. *Dans Logic Colloquium*, pages 123–146.

de Bruijn, N. (1972). Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem. *Dans Indagationes Mathematicæ*, volume 34.

de Bruijn, N. G. (1970). The mathematical language automath, its usage, and some of its extensions. *Dans* Laudet, M., Lacombe, D., Nolin, L. et Schützenberger, M., éditeurs : *Symposium on Automatic Demonstration*, volume 125 de *Lecture Notes in Mathematics*, pages 29–61. Springer Berlin / Heidelberg.

de Bruijn, N. G. (1980). A survey of the projet Automath. *Dans* Seldin, J. P. et Hindley, J. R., éditeurs : *To H. B. Curry : Essays on Combinatory Logic, Lambda Calculus, and Formalism*, pages 579–606. Academic Press.

Dowek, G. (1991). A second-order pattern matching algorithm for the cube of typed lambda-calculi. *Dans MFCS*, pages 151–160.

Feferman, S., Dawson, Jr., J. W., Kleene, S. C., Moore, G. H., Solovay, R. M. et van Heijenoort, J., éditeurs (1990). *Kurt Gödel : collected works. Vol. II : Publications 1938–1974*. Oxford University Press, Inc., New York, NY, USA.

Franchella, M. (1994). Brouwer and griss on intuitionistic negation. *Modern Logic 4*, 3:256–265.

Friedman, H. (1978). Classically and intuitionistically provably recursive functions. *Dans* Springer, éditeur : *Higher Set Theory*, volume 669, pages 21–27.

Gilmore, P. (1953). The effect of Griss' criticism of the intuitionistic logic on deductive theories formalized within the intuitionistic logic. *Indagationes Mathematicæ*, 15:162–174, 175–186.

Girard, J.-Y. (1972). *Interprétation fonctionnelle et élimination des coupures dans l'arithmétique d'ordre supérieur*. Thèse de doctorat d'état, Université Paris VII.

Girard, J.-Y., Taylor, P. et Lafont, Y. (1990). *Proofs and types*. Cambridge University Press.

Griss, G. (1946). Negationless intuitionistic mathematics. *Indagationes Mathematicæ*, 8:675–681.

Griss, G. (1950). Negationless intuitionistic mathematics II. *Indagationes Mathematicæ*, 12:108–115.

Griss, G. (1951a). Negationless intuitionistic mathematics III. *Indagationes Mathematicæ*, 13:193–199.

Griss, G. (1951b). Negationless intuitionistic mathematics IVa, IVb. *Indagationes Mathematicæ*, 13:452–462,463–471.

Gödel, V. K. (1958). Über eine bisher noch nicht benÜtzte erweiterung des finiten standpunktes. *Dialectica*, 12(3-4):280–287. Traduction en anglais dans Feferman *et al.* (1990).

Heyting, A. (1955). G. f. c. griss and his negationless intuitionistic mathematics. *Synthese*, 9:91–96. 10.1007/BF00567395.

Howard, W. A. (1980). The formulas-as-types notion of construction. *Dans* Seldin, J. P. et Hindley, J. R., éditeurs : *To H. B. Curry : Essays on Combinatory Logic, Lambda Calculus, and Formalism*, pages 479–490. Academic Press.

Huet, G. (1989). The Constructive Engine. *Dans* Publishing, W. S., éditeur : *A Perspective in Theoretical Computer Science – Commemorative Volume for Gift Siromoney*, pages 38–69.

Kleene, S. C. (1935). A theory of positive integers in formal logic. part I. *American Journal of Mathematics*, 57(1):pp. 153–173.

Krivtsov, V. N. (2000a). A Negationless Interpretation of Intuitionistic Theories. I. *Studia Logica*, 64(3):323–344.

Krivtsov, V. N. (2000b). A Negationless Interpretation of Intuitionistic Theories. II. *Studia Logica*, 65(2):155–179.

López-Escobar, E. G. K. (1972). Constructions and negationless logic. *Studia Logica*, 30(1):7–22.

López-Escobar, E. G. K. (1974). Elementary interpretations of negationless arithmetic. *Fundamenta Mathematicae*, 82(1):25–38.

Luo, Z. (1990). *An Extended Calculus of Constructions.* Thèse de doctorat, University of Edinburgh.

Martin-Löf, P. (1971). A theory of types.

Mezhlumbekova, V. (1975). Deductive capabilities of negationless intuitionistic arithmetic. *Moscow University Mathematical Bulletin*, 30(2).

Michel, D. (2008). *Systèmes formels et systèmes fonctionnels pédagogiques.* Thèse de doctorat, Université Paul-Verlaine – Metz.

Minichiello, J. K. (1969). An extension of negationless logic. *Notre Dame J. Formal Logic*, 10:298–302.

Mints, G. (2006). Notes on Constructive Negation. *Synthese*, 148(3):701–717.

Nelson, D. (1966). Non-Null Implication. *The Journal of Symbolic Logic*, 31(4):562–572.

Nelson, D. (1973). A complete negationless system. *Studia Logica*, 32:41–49.

Poincaré, H. (1913). *Dernières pensées.* Flammarion.

Reynolds, J. (1974). Towards a theory of type structure. *Dans* Robinet, B., éditeur : *Programming Symposium*, volume 19 de *Lecture Notes in Computer Science*, pages 408–425. Springer Berlin / Heidelberg.

Sørensen, M. H. et Urzyczyn, P. (2006). *Lectures on the Curry-Howard Isomorphism*, volume 149 de *Studies in Logic and the Foundations of Mathematics.* Elsevier.

Urzyczyn, P. (1997). Inhabitation in typed lambda-calculi (a syntactic approach). *Dans* de Groote, P. et Roger Hindley, J., éditeurs : *Typed Lambda Calculi and Applications*, volume 1210 de *Lecture Notes in Computer Science*, pages 373–389. Springer Berlin / Heidelberg.

Valpola, V. (1955). Ein system der negationlosen Logik mit ausschliesslich realisierbaren Prädicaten. *Acta Philosophica Fennica*, 9:1–247.

van Daalen, D. T. (1994). Selected papers on Automath. *Dans* Nederpelt, R. P., Geuvers, J. H. et de Vrijer, R. C., éditeurs : *Selected Papers on Automath*, volume 133 de *Studies in Logic and the Foundations of Mathematics*, pages 101–126. Elsevier.

Vredenduin, P. (1953). The logic of negationless mathematics. *Compositio Mathematica*, 11:204–277.