Université de Savoie

U.F.R. Sciences Fondamentales et Appliquées

Laboratoire de Mathématiques LAMA

Thèse

pour obtenir le grade de

Docteur de l'Université de Savoie

Discipline : Mathématiques

Titre :

Étude d'un λ-calcul issu d'une logique classique

présentée et soutenue par

Khelifa SABER

Le 6 juillet 2007

dirigée par

Karim NOUR

Jury composé de:

Ph. De Groote	Rapporteur
R. Matthes	Rapporteur
R. David	Examinateur
T. Ehrhard	Examinateur
H. Herbelin	Examinateur
K. Nour	Directeur

”Nous élevons en rang qui Nous voulons et au-dessus de tout savant se trouve un savantissime.”

Coran 12, 76.

Remerciements

Se lancer dans l'aventure d'une thèse, c'était difficile mais il a été encore plus difficile de la mener à terme. On s'inquiète, on doute et surtout on souffre. Le stress, les incertitudes... on frôle la dépression. Mais quand on a un directeur qui s'appelle **Karim**, on s'accroche et on y croit. Je tiens à lui adresser mes profonds remerciements, sans son encadrement bienveillant, ses conseils et ses idées, ce travail n'aurait pu voir le jour. Par ses qualités humaines, il n'a jamais été avare de remonte-moral. Combien de fois a t-il supporté mes longues absences !? Mes retards et rendez-vous reportés pour ne pas dire annulés ! Mais lui, restait toujours patient et attentif à mes moindres soucis. Comment oublier son soutien inconditionnel, sans faille jusqu'au bout !? Pour lui témoigner ma gratitude et mon éternelle reconnaissance, les paroles ne peuvent suffire. Cela ne m'empêche pas de te dire et de te répeter : ***Karim***, *merci du fond du coeur.*

Je ne saurais manquer non plus de remercier **René David** pour son investissement tout au long de ce parcours. J'ai eu l'extrême honneur d'avoir bénéficié de son soutien lors de mon arrivée en France en octobre 2001 ; qu'il trouve ici l'expression de mes sincères remerciements.

Je tiens à remercier les rapporteurs, **Philippe De Groote** et **Ralph Matthes** pour leurs remarques pertinentes, qui ont contribué à l'amélioration de ce travail.

Je remercie également **Thomas Ehrhard** et **Hugo Herbelin** pour l'honneur qu'ils m'ont fait d'accepter d'être examinateurs de cette thèse.

Mes salutations à tous ceux que j'ai cotoyés durant ces années au sein du bâtiment "Le Chablais" en particulier les membres de l'équipe de logique.

Je ne saurais oublier tous les membres du laboratoire **LAIC** à Clermont-Ferrand pour leur chaleureux accueil et l'intérêt qu'ils ont porté à ce travail tout au long de cette année.

Enfin, une pensée à tous mes amis d'ici et de l'autre rive.

Contents

*" **Mon coeur m'a dit: "Je veux savoir, je veux connaître !***
Instruis-moi, Khayyâm, toi qui as tant travaillé !"
J'ai prononcé la première lettre de l'alphabet, et mon coeur m'a dit:
"Maintenant, je sais. Un est le premier chiffre du nombre qui ne finit pas..."

Omar el-Khayyâm

Chapter 1

Introduction

1.1 Historique de la déduction naturelle

Depuis les travaux de nombreux logiciens dont Georges Boole, Bertrand Russell, Alfred North Whitehead en passant par David Hilbert, Kurt Gödel, Gerhard Gentzen et jusqu'à aujourd'hui, la théorie de la démonstration a été au coeur de tout débat lié aux problèmes des fondements des mathématiques : des problèmes dûs à leur complexification et à l'apparition de divers paradoxes.

Cette discipline a vu le jour officiellement au début du siècle dernier. Au cours de son célèbre exposé, le 8 août 1900 à Paris, au second congrès international de mathématiques, D. Hilbert la présenta comme un excellent outil pour résoudre l'un de ses fameux 23 problèmes : en l'occurence, le deuxième dans lequel il s'interrogea sur la consistance des axiomes de l'arithmétique. Ceci revient donc à démontrer, par des moyens *finitistes*, la consistance de l'arithmétique. Cet objectif a suscité de nombreux travaux en logique dans les années qui suivirent, jusqu'en 1931 où il fût invalidé par K. Gödel et son célèbre *théorème d'incomplétude*, démontrant ainsi l'impossibilité de réaliser un tel programme. Cela n'a toutefois pas empêché cette discipline de se développer et de s'imposer parmi les sous-domaines majeurs au sein de la logique, grâce notamment aux travaux de Jacques Herbrand, Gerhard Gentzen et toute une génération de logiciens des années 30.

En effet, G. Gentzen fût le premier à avoir caractérisé la logique comme un cheminement naturel, et cela à travers le développement de ses deux célèbres formalismes : *la déduction naturelle* et *le calcul des séquents* dans leurs deux variantes classique et intuitionniste. Pour la déduction naturelle, la principale idée de départ était : *pas d'axiomes logiques, uniquement des règles de déduction et autant qu'il en faut pour reproduire toutes les formes élémentaires et naturelles du raisonnement.* Pour réaliser son idée, G. Gentzen développa un formalisme où les déductions ne sont pas des suites de phrases mais des arbres, faites de

colonnes de phrases qui se rejoignent jusqu'à la conclusion. Pour G. Gentzen, cette méthode était la mieux adaptée au style du raisonnement humain utilisé en pratique, d'où la terminologie *la déduction naturelle*.

Les règles de la déduction naturelle permettent d'enchaîner logiquement les phrases, c'est-à-dire introduire de nouvelles phrases comme conséquences logiques de celles qui précèdent. A chaque connecteur logique sont associées deux sortes de règles de déduction.

Une *règle d'introduction* qui permet d'avoir en conclusion une proposition ayant ce connecteur comme principal.

Une *règle d'élimination* qui explique comment manipuler une proposition ayant ce connecteur comme principal afin de poursuivre le raisonnement.

"Introduction" et "Elimination" sont nécessaires pour pouvoir assembler et désassembler des formules et analyser les démonstrations.

G. Gentzen proposa alors son formalisme pour prouver la cohérence de l'arithmétique. Mais les nombreuses difficultés qu'il rencontra le conduisirent à reformuler la déduction naturelle en une nouvelle version plus symétrique. Cela donna naissance au calcul des séquents qui révèle mieux la dualité hypothèse/conclusion exprimée par la symétrie parfaite entre la droite et la gauche dans les séquents, ce qui permet de rendre explicite un grand nombre de propriétés de la logique. C'est pourquoi le calcul des séquents doit se voir comme un formalisme pour *raisonner* sur les preuves formelles plutôt que pour *rédiger* des preuves formelles.

Pour ce calcul, G. Gentzen chercha à démontrer la propriété suivante, dite propriété de *la sous-formule* et appelée aussi **Hauptsatz** (théorème principal) qui affirme : "*toute démonstration peut se ramener à une autre qui ne comporte pas de détours. On n'y introduit aucun concept qui ne soit pas contenu dans son résultat final et qui, par conséquent, ne doit pas nécessairement être utilisé pour obtenir ce résultat*" [10]. Cette propriété est très importante, car c'est elle qui lui permettra de démontrer la consistance du calcul des séquents. En effet, le séquent le plus simple est le séquent *vide* qui ne contient aucune formule, ni à gauche, ni à droite. Or ce séquent exprime une contradiction. Pour prouver que le calcul est cohérent, il suffit de démontrer que le séquent *vide* n'est pas dérivable. Néanmoins, une démonstration sans coupures du séquent *vide* ne devrait contenir que des séquents plus simples mais il n'y en a aucun. C'est ainsi que dans sa quête, G. Gentzen sera conduit à démontrer son célèbre théorème d'*élimination des coupures*. L'adaptation de cette démonstration au cas de l'arithmétique lui fournit une preuve de la cohérence relative de l'arithmétique. Une fois ce problème résolu, resta celui de la consistance de la déduction naturelle elle-même. G. Gentzen ne prouva cette consistance qu'en démontrant son équivalence au calcul des séquents.

Trente ans après, en 1965, D. Prawitz montra comment normaliser les preuves dans le cas de la présence de la règle d'absurdité classique. Dans le but d'obtenir la propriété de la sous-formule, D. Prawitz restreindra l'utilisation de cette règle au cas où sa conclusion serait atomique. Ensuite, il montra comment transformer toute déduction afin de réaliser cette propriété ([7] et [34]).

En 1980, W. Howard publia une correspondance entre les démonstrations en déduction naturelle intuitionniste et les termes du λ-calcul de A. Church, où la normalisation des preuves de D. Prawitz correspond à la β-réduction en λ-calcul. Bien avant, certaines correspondances entre la logique et le λ-calcul ont été constatées séparément par H. Curry et R. Feys en 1958 d'une part, et De Bruijn en 1968 d'autre part. Cela donna naissance à ce qu'on appelle aujourd'hui *la correspondance de Curry-Howard* (appelée aussi *isomorphisme de Curry-Howard*, ou *isomorphisme de Curry-Howard-De Bruijn*). C'est une relation entre logique intuitionniste, λ-calcul et informatique, ou encore entre démonstrations intuitionnistes et calculs par intermédiaire du λ-calcul. Cette correspondance s'est avérée être une puissante technique et un outil fiable par l'exposition du lien structurel et profond, qui permet aux λ-termes de coder les preuves de la logique intuitionniste, tandis que la β-réduction représente le calcul effectué par ordinateur.

Cependant, cette correspondance entre preuves et programmes resta partielle selon deux points de vue. D'un point de vue informatique, certains aspects de la programmation ne sont pas couverts comme la programmation parallèle ou la gestion des erreurs. Ensuite, d'un point de vue mathématique, en logique intuitionniste, le raisonnement par l'absurde utilisé dans la quasi-totalité des mathématiques est interdit.

Or, depuis la voie ouverte par T. Griffin [12] au début des années 1990, il est apparu naturel d'étendre la logique du λ-calcul, car la correspondance de Curry-Howard a été enrichie. T. Griffin a découvert que certaines instructions informatiques, telles que les commandes d'échappement, traduisent le processus du raisonnement par l'absurde. Au niveau de la théorie de la démonstration, le problème consista à définir un système de déduction naturelle classique doté d'une procédure de normalisation de preuve, tel que la preuve normale satisfait la propriété de la sous-formule. Cela permettra d'interpréter la procédure de normalisation des preuves comme un processus de calcul.

Sur la base de ce nouvel apport, M. Parigot a introduit son $\lambda\mu$-calcul en 1992 ([32] et [33]). C'est une extension du λ-calcul qui capture le contenu algorithmique des preuves de la logique classique propositionnelle du second ordre. Le nouvel opérateur μ est introduit pour coder la règle du raisonnement par absurde.

Signalons que les connecteurs $\wedge$ et $\vee$ peuvent être codés dans cette logique à l'aide du connecteur $\rightarrow$ et le quantificateur $\forall$. Le $\lambda\mu$-calcul possède toutes les bonnes propriétés : préservation de type, confluence et forte normalisation [2], [32], [33] et [36].

1.2 Le cadre de mes recherches

En 2001, Ph. De Groote [7] a introduit une extension du $\lambda\mu$-calcul : le $\lambda\mu^{\wedge\vee}$-calcul. L'objectif était de donner une preuve de la forte normalisation de l'élimination des coupures en déduction naturelle classique propositionnelle. Ainsi, les termes du $\lambda\mu^{\wedge\vee}$-calcul codent les démonstrations de cette logique, où sont considérées trois sortes de coupures : logiques, permutatives et classiques, nécessaires à l'obtention de la propriété de la sous-formule. D'un point de vue informatique, ce calcul peut être considéré comme un langage de programmation fonctionnelle incluant produit et opérateurs de contrôles dans lequel la normalisation des termes correspond à un processus de calcul. Ceci justifie encore plus l'intérêt porté au théorème de la forte normalisation car il garantit la terminaison de tout calcul, en d'autres termes, l'existence d'une forme normale indépendamment de la stratégie de réduction adoptée.

Dans [7], la preuve de ce résultat se décompose en deux grandes parties : la première partie consiste à démontrer la terminaison des réductions permutatives et classiques pour tout le calcul (typé ou non typé), ce qui prouve que ces réductions structurelles n'ont aucun contenu calculatoire. La deuxième partie consiste à traduire le $\lambda\mu^{\wedge\vee}$-calcul en λ-calcul par les transformations CPS, ce qui revient finalement à réduire le problème à celui de la forte normalisation du λ-calcul simplement typé.

L'idée principale des transformations CPS, c'est qu'elles préservent le caractère strict des réductions logiques, i.e. si un terme t se réduit à un terme u par au moins une réduction, alors la transformation CPS de t se réduit aussi à celle de u par au moins une réduction. Toutefois, cela ne fonctionne pas avec certains termes, en l'occurence les termes avec des μ-abstractions vacantes. Cependant, et même en isolant cette classe de termes, la preuve dans [7] n'est pas complète, car certains termes qui n'ont aucune μ-abstraction vacante peuvent se réduire à d'autres contenants des μ-abstractions vacantes. Récemment, quelques travaux qui traitent ce problème sont apparus, avec comme objectifs de corriger ou complèter les cas non couverts par les versions initiales de certaines preuves de forte normalisation via les CPS transformations, et cela à travers la notion d'augmentation [25] and [26].

Le théorème de la forte normalisation a suscité l'intérêt de nombreux auteurs: R. David et K. Nour [4] ont produit une autre démonstration. Il s'agit d'une preuve syntaxique basée essentiellement sur deux lemmes. Le premier lemme stipule que la démonstration obtenue en substituant une démonstration fortement normalisable dans une autre fortement normalisable l'est aussi. Le second lemme est très technique et relatif aux réductions commutatives. Une version détaillée de la preuve de ce dernier sera disponible sur la page web des auteurs.

Une autre preuve de la forte normalisation est fournie par R. Matthes [24]. Elle s'inscrit dans le cadre des preuves sémantiques dont la base est la notion des ensembles saturés et points fixes.

1.2.1 La contribution de mon travail au $\lambda\mu^{\wedge\vee}$-calcul

1. Je présente ici une preuve du théorème de la forte normalisation. Ma preuve rejoint celle de R. Matthes et s'inscrit également dans le cadre des preuves sémantiques. C'est une adaptation de la preuve de la forte normalisation du $\lambda\mu$-calcul typé de M. Parigot [33], basée sur les candidats de réductibilité introduits par J.-Y. Girard [11]. Les types sont interprétés par des ensembles de termes qui satisfont certaines propriétés. L'interprétation donnée au type disjunction $A \vee B$ est très intuitive, elle traduit exactement la règle d'élimination du connecteur $\vee$. De ce fait, cela m'a permis de réduire de façon considérable la difficulté du problème de la forte normalisation. Néanmoins, cette preuve s'appuie sur le lemme technique présenté dans [4].

2. A partir de la sémantique précédente qui est à la base de la preuve de la forte normalisation, j'ai défini une nouvelle sémantique de réalisabilité pour le $\lambda\mu^{\wedge\vee}$-calcul. Au coeur de cette nouvelle sémantique, l'ensemble $\mathcal{N}$ des termes fortement normalisables est remplacé par un ensemble de termes saturé par expansion, noté $\mathcal{S}$. Cela permet plus de liberté dans l'application du nouveau lemme d'adéquation. Donc, en fonction des propriétés recherchées, l'ensemble $\mathcal{S}$ est redéfini afin de satisfaire les résultats voulus. Ainsi, comme application de ce nouveau lemme d'adéquation, on caractérise le comportement algorithmique de certains termes typés et clos uniquement à partir de leurs types. Des travaux similaires ont déjà été réalisés dans ce domaine (sémantique de réalisabilité et caractérisation de comportement [17] et [31]). Pouvoir capter ce type de comportement constitue une justification et montre l'intérêt d'une telle sémantique. Un autre outil, les méthodes syntaxiques, est à notre disposition pour capter ce type de comportement. A la différence des preuves syntaxiques, dans les preuves

sémantiques, il faut d'abord deviner ou avoir une idée de ce que fait ce terme, ensuite vérifier cette preuve avec le bon ensemble $\mathcal{S}$. Tandis que dans les preuves syntaxiques, ce comportement se construit tout au long de la démonstration.

3. La plupart des preuves sémantiques de la forte normalisation sont basées sur un résultat appelé *lemme d'adéquation*, stipulant que chaque terme est dans l'interprétation de son type. La question qu'on peut se poser est la suivante : "a-t-on un résultat réciproque de ce lemme ?"; autrement dit : "existe-t-il une classe de type pour laquelle la réciproque de ce lemme est vraie ?". J. R. Hindley fût le premier à s'être intéressé à cette question dans le cadre des systèmes de types simples [13], [14] et [15]. Dans [21], R. Labib-Sami a établi un résultat de complétude pour une classe de types appelés types $\forall$-positifs du système $\mathcal{F}$ de J.-Y. Girard. Sa sémantique est basée sur des ensembles stables par $\beta\eta$-équivalence. Plus tard, S. Farkh et K. Nour ont démontré un résultat plus affiné que celui de [21] et cela toujours pour la même classe de types mais avec une sémantique à base d'ensembles saturés par réduction de tête faible [8], ce qui est plus léger comme condition. Les mêmes auteurs ont prouvé dans [9] un autre résultat de complétude pour une classe de types du système $\mathcal{AF}2$ de J.-L. Krivine. Dans le cadre du $\lambda\mu^{\wedge\vee}$-calcul, l'étude de cette question a donné lieu à une réponse négative dûe à la présence des connecteurs $\wedge$ et $\vee$. C'est pour cela que je me suis restreint au $\lambda\mu$-calcul simplement typé, pour lequel un résultat de complétude est démontré à travers un long travail et quelques légères modifications apportées à notre sémantique.

4. La confluence reste parmi les plus importantes propriétés de tout système de réduction abstrait. Elle garantit l'unicité de la forme normale au cas où celle-ci existe. De ce fait, pour le $\lambda\mu^{\wedge\vee}$-calcul, diverses preuves de la confluence sont apparues. Dans [7], Ph. De Groote propose de démontrer la confluence par le lemme de Newman et la vérification de la confluence locale. Dans [1], Y. Andou démontre ce résultat en utilisant des réductions parallèles définies par une extension de la notion des *segments* de D. Prawitz ([34] et [35]) et de développement complet. Dans [5], R. David démontre la confluence de la β-réduction en s'appuyant sur un théorème de standardisation et de développements finis. Ma démonstration s'inspire fortement de celle de R. David. Je démontre tout d'abord un théorème de standardisation et un théorème de développements finis. La confluence découle par arguments de forte normalisation, lemme de Newman et confluence locale.

5. Le théorème de standardisation est très utile et intéressant en lui même. Il stipule que chaque suite de réductions peut être transformée en une autre qui sera *standard* dans le sens où les réductions sont faites de *gauche* à *droite*, éventuellement avec des *sauts*. Mais sans la possibilité de *revenir*

sur les redexes ou leurs résidus après les avoir *sautés*. Dans la littérature, plusieurs définitions équivalentes sont données à cette notion. La définition de R. David [5] me semble la mieux adaptée, d'autant plus qu'elle ne fait pas intervenir explicitement la notion de résidus de redexes. La difficulté pour le $\lambda\mu^{\wedge\vee}$-calcul, c'est la présence des réductions permutative et classique. Pour cela, le travail effectué par F. Joachimski et R. Matthes [16] m'a été d'une grande utilité.

En effet, dans [16], les auteurs présentent le ΛJ-calcul, une extension du λ-calcul avec des applications généralisées, où ils traitent justement la question de la standardisation en présence de réduction parmutative. Ainsi le ΛJ-calcul sert comme un modèle minimal pour l'étude des systèmes de réecriture avec réduction commutative. Je définis alors une réduction standard "à la David" qui capture la notion intuitive de réduire de gauche à droite et de *l'exterieur* vers *l'interieur* d'une manière similaire à celle présentée dans [16]. L'une des conséquences de cette définition est le fait que *la réduction gauche* est une stratégie gagnante. Pour définir *la* réduction gauche dans le $\lambda\mu^{\wedge\vee}$-calcul, il faudra d'abord définir la notion de réduction de tête. La difficulté provient du fait que les $\lambda\mu^{\wedge\vee}$-termes n'ont pas qu'une seule tête, d'où la formulation de la notion *d'une* réduction gauche comme itérations de réductions de tête. Cette légère difficulté est contournée en considérant *la* réduction gauche comme une réduction standard particulière *déterministe*, cela permet de dériver une définition *équivalente* à la précédente.

6. Le théorème des développements finis est aussi important et joue le rôle du théorème de forte normalisation quand on le combine avec le lemme de Newman. Il stipule qu'en partant d'un terme et d'un ensemble fixé de ses redexes, si on ne réduit que les résidus de ces redexes, alors toute réduction termine. J'introduis dans ce travail une version colorée du $\lambda\mu^{\wedge\vee}$-calcul afin de suivre les traces des redexes et leurs résidus. En n'autorisant que la réduction des redexes colorés, je démontre un résultat de forte normalisation (ce qui correspond intuitivement à un théorème des développements finis).

7. La machine de Krivine est une simple implémentation de la stratégie de la réduction de tête faible des λ-termes. Cette machine a servi comme base pour de nombreuses études théoriques et pour l'implémentation d'autres stratégies de réductions ([3], [6] et [22]). Une extension de cette machine au cadre du $\lambda\mu^{\wedge\vee}$-calcul sera présentée à la fin du second chapitre.

8. Plusieurs langages de programmation ont été développés à travers l'étude de différents calculs par valeur comme `ML` et `Lisp` pour le λ-calcul, et μPCF_V pour le $\lambda\mu$-calcul. C'est pour cela que l'étude d'un $\lambda\mu^{\wedge\vee}$-calcul par valeur semble intéressante. Je présente dans mes travaux un $\lambda\mu^{\wedge\vee}$-calcul par valeur. La preuve de la confluence de ce calcul est une adaptation de celle de

Y. Andou [1]. Cette méthode est la mieux adaptée pour fournir le résultat voulu.
En effet, vu la présence des réductions permutatives ainsi que la μ'-réduction, on ne peut démontrer la confluence à travers des résultats de retardement ou de commutations. La forte normalisation de ce système n'étant pas prouvée, on ne pourra pas non plus utiliser le lemme de Newman et la confluence locale. Il est clair donc que la méthode la plus efficace reste la méthode des réductions parallèles.

1.2.2 Les principaux résultats de ce travail

1. La standardisation, la confluence et une extension de la machine de J.-L. Krivine.

2. Une preuve sémantique de la forte normalisation.

3. Une sémantique de réalisabilité qui permet de caractériser le comportement calculatoire de certains termes typés.

4. Un théorème de complétude pour le $\lambda\mu$-calcul simplement typé.

5. Une introduction à un calcul par valeur confluent.

1.3 Le plan de la thèse

Cette thèse est divisée en cinq chapitres qui, hormis quelques points de détail, sont indépendants les uns des autres et "autosuffisants". Le lecteur peut donc les aborder au gré de son intérêt et de ses besoins.

- **Le chapitre 2** présente une collection de résultats communs à tout système de réduction abstrait. Tout d'abord, je démontre des résultats de standardisation, de développements finis et de confluence, puis je termine par une extension de la machine de Krivine.

- **Le chapitre 3** contient une preuve sémantique de la forte normalisation du $\lambda\mu^{\wedge\vee}$-calcul typé. Cette preuve est basée sur la méthode des candidats de réductibilités de G.-Y. Girard [11], adaptée par M. Parigot au cas classique [32].

- **Le chapitre 4** définit une sémantique de réalisabilité, et par conséquent, un lemme de correction qui aura pour application la caractérisation de quelques termes clos de certains types.

– **Le chapitre 5** présente le $\lambda\mu$-calcul simplement typé. Je démontre que les types de ce système sont complets pour la sémantique définie dans les deux précédents chapitres.

– **Le chapitre 6** introduit un $\lambda\mu^{\wedge\vee}$-calcul par valeur. La confluence est prouvée par la méthode des réductions parallèles et du développement complet, inspirée et basée sur le travail de Y. Andou [1].

Bibliography

[1] Y. Andou. *Church-Rosser property of simple reduction for full first-order classical natural deduction.* Annals of Pure and Applied Logic, vol 119, pp. 225-237, 2003.

[2] K. Baba, S. Hirokawa & K. Fujita. *Parallel Reduction in Type Free $\lambda\mu$-Calculus.* Electronic Notes in Theoritical Computer Science, vol 42, pp. 1-15, 2001.

[3] P. Crégut. *Machines à environnement pour la réduction symbolique et l'évaluation partielle.* PhD Thesis, Paris 7 University, 1991.

[4] R. David and K. Nour. *A short proof of the Strong Normalization of Classical Natural Deduction with Disjunction.* Journal of Symbolic Logic, vol 68, num 4, pp. 1277-1288, 2003.

[5] R. David. *A simple proof of basic results in λ-calculus.* Compte Rendu de l'Académie des Sciences, Paris, Tome 320, Série 1, 1995.

[6] P. De Groote. *An environment machine for the lambda-mu-calculus.* Mathematical Structures in Computer Science, 8(6), pp. 637-669, 1998.

[7] P. De Groote. *Strong Normalization of Classical Natural Deduction with Disjunction.* In 5th International Conference on typed lambda calculi and applications, TLCA'01. LNCS (2044), pp. 182-196. Springer Verlag, 2001.

[8] S. Farkh and K. Nour. *Un résultat de complétude pour les types $\forall^+$ du système $\mathcal{F}$.* CRAS. Paris 326, Série I, pp. 275-279, 1998.

[9] S. Farkh and K. Nour. *types complets dans une extension du système $\mathcal{AF}2$.* Informatique théorique et application 31-6, pp. 513-537, 1998.

[10] G. Gentzen. *Recherches sur la déduction logique.* Press Universitaires de France, 1955. Traduction et commentaires par R. Feys et J. Ladrière.

[11] J.-Y. Girard, Y. Lafont, P. Taylor. Proofs and types. *Cambridge University Press*, 1986.

[12] T. Griffin. *A formulae-as-types notion of control.* Proc. POLP, 1990.

[13] J. R. Hindley. *The simple semantics for Coppe-Dezani-Sallé types.* Proceeding of the 5th Colloquium on International Symposium on Programming, pp. 212-226, April 06-08, 1982.

[14] J. R. Hindley. *The completeness theorem for the typing λ-terms.* Theoretical Computer Science, 22(1), pp. 1-17, 1983.

[15] J. R. Hindley. *Curry's type-rules are complete with respect to the F-semantics too.*Theoretical Computer Science, 22, pp. 127-133, 1983.

[16] F. Joachimski and R. Matthes. *Standardization and Confluence for a Lambda calclus with Generalized Applications.* 11th International Confrence, RTA 2000, Norwich, UK, July 10-12, pp. 141-155, 2000.

[17] J.-L. Krivine. *Lambda calcul, types et modèle.* Masson, Paris, 1990.

[18] J.-L. Krivine. *Un interpréteur du λ-calcul.* Unpublished draft. Available at `http://www.pps.jussieu.fr/krivine/`.

[19] J.-L. Krivine. *A call by-name lambda-calculus machine.* to apear in Higher Order and Symbolic Computation.

[20] J.-L. Krivine. *Opérateurs de mise en mémoire et traduction de Gödel.* Archive for Mathematical Logic, vol 30, pp. 241-267, 1990.

[21] R. Labib-Sami. *Typer avec (ou sans) types auxiliaires.* Manuscrit, 1986.

[22] F. Lang, Z. Benaissa and P. Lescane. *Super-Closures.* In Proc, of WPAM'98, as Technical Report of the University of SaarBruck, number A 02/98, 1998.

[23] O. Laurent. *Interprétation calculatoire de la logique classique: λμ-calcul et machine de Krivine.* Available at `http://www.pps.jussieu.fr/laurent/`.

[24] R. Matthes. *Non-Strictly Positive Fixed Points for Classical Natural Deduction.* APAL, vol 133, pp. 205-230, 2005.

[25] K. Nakazawa. and M. Tatsuka *Strong normalization proof with CPS-translation for second oder classical natural deductionλμ-calculus.* The Journal of Symbolic Logic, vol 68, number 3, pp. 851-859, Sept. 2003.

[26] K. Nakazawa. *Confluency and Strong normalizability of call-by-value λμ-calculus.* Theoretical Computer Science, vol 290, pp. 429-463, 2003.

[27] K. Nour and K. Saber. *A Semantics of Realizability for the Classical Propositional Natural Deduction.* Electronic Notes in Theoretical Computer Science, vol 140, pp. 31-39, 2005.

[28] K. Nour and K. Saber. *A semantical proof of strong normalization theorem for full propositional classical natural deduction.* Archive for Mathematical Logic, vol 45, pp. 357-364, 2005.

[29] K. Nour and K. Saber. *Confluency property of the call-by-value $\lambda\mu^{\wedge\vee}$-calculus.* Computational Logic and Applications CLA'05. Discrete Mathematics and Theoretical Computer Science proc, pp. 97-108, 2006.

[30] K. Nour. *Opérateurs de mise en mémoire et types $\forall$-positifs.* Theoretical Informatics and Applications, vol 30, n° 3, pp. 261-293, 1996.

[31] K. Nour. *Mixed Logic and Storage Operators.* Archive for Mathematical Logic, vol 39, pp. 261-280, 2000.

[32] M. Parigot. *$\lambda\mu$-calculus: An algorithm interpretation of classical natural deduction.* Lecture Notes in Artificial Intelligence, vol 624, pp. 190-201. Springer Verlag, 1992.

[33] M. Parigot. *Proofs of strong normalization for second order classical natural deduction.* Journal of Symbolic Logic, vol 62 (4), pp. 1461-1479, 1997.

[34] D. Prawitz. *Natural Deduction, A proof-Theoretical Study.* Almqvist & Wiksell, Stockholm, 1965.

[35] D. Prawitz. *Idea and result in proof theory.* In 2nd Scandinavian Logic Symp, pp. 235-307, 1971.

[36] W. Py. *Confluence en $\lambda\mu$-calcul.* PhD thesis, University of Chambéry, 1998.

Chapter 2

Some properties of the $\lambda\mu^{\wedge\vee}$-calculus

2.1 Introduction

The $\lambda\mu^{\wedge\vee}$-calculus is an extension of the $\lambda\mu$-calculus associated by the Curry-Howard correspondence to the full classical natural deduction system, it was introduced by P. De Groote [11].

In the $\lambda\mu^{\wedge\vee}$-calculus as in any other abstract reduction system, termination, confluence and standardization appear among the principal properties. The question of termination, of course for the typed $\lambda\mu^{\wedge\vee}$-calculus, was studied by many authors [7], [11], [18] and [19].

The confluence is a very important property, it guaranteed the uniqueness of the normal form (if it exists) independently of the strategy of reduction, i.e. if we are allowed to write terms which necessarily do not finish under reduction, one expects at least that the possible result is independent of the strategy of reduction. There are different methods to prove the confluence property:
parallel reduction, complete development, finiteness developments and standardization... For a strongly normalizable term rewriting system, one needs only to check the local confluence which suffices when combinated with Newman lemma.

Standardization is classical and a very convenient tool, the issue is the order in which the reduction steps are performed. In a standard reduction, this is done from *left* to *right*. According to the standardization theorem, any sequence of reductions can be transformed into a standard one. We find in the current literature various equivalent definitions of this notion. In our work, we adopt the one given by R. David and W. Py (see [8] and [22]), it is very convenient and has the advantage that we do not need to explicit the notion of residus "descendant" of a redex.

The presence of the permutative reduction of the form $((t\ [x.u, y.v])\ \varepsilon) \rightarrow_\delta (t\ [x.(u\ \varepsilon), y.(v\ \varepsilon)])$ has certain consequences not only on the termination of

the system, but also on the standardization and the confluence, since the resulting rewriting system is not orthogonal. Therefore the treatment of these two notions is not trivial at all. Intuitively standard reduction contract redexes from *external* to *internal* and from *left* to *right*. However There is more freedom in the presence of permutative reductions, a permutative redex of the form $((\mu a.t\ [x.u, y.v])\ [r.p, s.q])\ \varepsilon$ may permute to $(\mu a.t\ [x.(u\ [r.p, s.q]), y.(v\ [r.p, s.q])])\ \varepsilon$ and to $(\mu a.t\ [x.u, y.v])\ [r.(p\ \varepsilon), s.(q\ \varepsilon)]$ and both possiblities as well as the embedded μ-redex should be treated equivalently. We can not privilege one between them and consider for example the classical as *external* or *the leftmost*. That would be also the same thing for the two permutatives. In this work we use a definition "à la David" which captures this intuitive notion of standardization, when restricted to the λ-calculus (resp the $\lambda\mu$-calculus) corresponds exactly to the one given in [8] (resp [22]). As an application of this definition, we prove that leftmost reduction in a sense precise later is a gaining strategy.

The finiteness developments theorem says that: if we mark a set R of redex occurences in a given term t and reduce only the marked redex occurences and redex occurences which descend from marked redex occurences, the reduction process always terminates. If we reduce every marked redex occurence, then the order in which such reductions are performed does not matter, R uniquely determines a term u to which t is reduced under any complete reduction of marked redex occurences. In addition, if we mark another set R' of redex occurences in t and follow this set through a complete R-reduction, the redex occurences from R' may be transformed by substitution or copied. However, it does not matter in what way we perform a complete R-reduction, the set of redex occurences in u which descend from R' is again uniquely determined. This theorem has important consequences like the confluence property, what guaranteed the uniqueness of normal form if it exists. The proof of such theorem is difficult and required a standardization theorem, this is what we do in the major part of this work.

The Krivine machine (see [15]) is a simple and natural implementation of the weak-head call-by-name reduction strategy for the pure λ-terms. It can be described just with three or four rules in minimal machinery. The Krivine machine has served as a basis for many variants, extensions and theoretical studies.
P. Crégut [4] used it as a basis for the implementation of other reduction strategies (call-by-need, head and strong reductions). Many others used it for their works either practical or theoretical (see [10, 16, 23]). All these works demonstrated that the simplicity of the Krivine machine makes it a valuable tool to study new implementation techniques and various λ-calculus extensions. We define here an extension of this machine to the $\lambda\mu^{\wedge\vee}$-calculus.

This chapter is presented as follows. Section 2 is an introduction to the typed $\lambda\mu^{\wedge\vee}$-calculus. Section 3, contains some useful technical results, in order to well defining head and leftmost reduction. In Section 4, we define the standard reductions and prove the standardization theorem. In Section 5, we introduce a coloured version of $\lambda\mu^{\wedge\vee}$-calculus, to keep the trace of the residus of redexes and

prove the finiteness developments theorem. We close this section by the main theorem of this chapter, i.e. the confluence property. In Section 6, we extend the Krivine machine to the $\lambda\mu^{\wedge\vee}$-calculus, in order to perform terms by a particular weak-head reduction.

2.2 Notations and definitions

Definition 2.2.1 *We use notations inspired by the paper [2].*

1. *Types are formulas of propositional logic built from the set of propositional variables and the constant type $\bot$, using the connectors $\to$, $\wedge$ and $\vee$.*

2. *Let $\mathcal{X}$ and $\mathcal{A}$ be two disjoint infinite alphabets for distinguishing the λ-variables and μ-variables respectively. We code deductions by using a set of terms $\mathcal{T}$ which extends the λ-terms and is given by the following grammars:*

$$\begin{array}{rcl} \mathcal{T} & := & \mathcal{X} \mid \lambda\mathcal{X}.\mathcal{T} \mid (\mathcal{T}\ \ \mathcal{E}) \mid \langle \mathcal{T}, \mathcal{T} \rangle \mid \omega_1\mathcal{T} \mid \omega_2\mathcal{T} \mid \mu\mathcal{A}.\mathcal{T} \mid (\mathcal{A}\ \ \mathcal{T}) \\ \mathcal{E} & := & \mathcal{T} \mid \pi_1 \mid \pi_2 \mid [\mathcal{X}.\mathcal{T}, \mathcal{X}.\mathcal{T}] \end{array}$$

   An element of the set $\mathcal{E}$ is said to be an $\mathcal{E}$-term.

3. *The meaning of the new constructors is given by the typing rules below where Γ (resp Δ) is a context, i.e. a set of declarations of the form $x : A$ (resp $a : A$) where x is a λ-variable (resp a is a μ-variable) and A is a formula.*

$$\frac{}{\Gamma, x : A\ \vdash x : A\ ;\ \Delta}\, ax$$

$$\frac{\Gamma, x : A \vdash t : B; \Delta}{\Gamma \vdash \lambda x.t : A \to B; \Delta}\to_i \qquad \frac{\Gamma \vdash u : A \to B; \Delta \quad \Gamma \vdash v : A; \Delta}{\Gamma \vdash (u\ \ v) : B; \Delta}\to_e$$

$$\frac{\Gamma \vdash u : A; \Delta \quad \Gamma \vdash v : B; \Delta}{\Gamma \vdash \langle u, v \rangle : A \wedge B; \Delta}\wedge_i$$

$$\frac{\Gamma \vdash t : A \wedge B; \Delta}{\Gamma \vdash (t\ \ \pi_1) : A; \Delta}\wedge_e^1 \quad \frac{\Gamma \vdash t : A \wedge B; \Delta}{\Gamma \vdash (t\ \ \pi_2) : B; \Delta}\wedge_e^2$$

$$\frac{\Gamma \vdash t : A; \Delta}{\Gamma \vdash \omega_1 t : A \vee B; \Delta}\vee_i^1 \quad \frac{\Gamma \vdash t : B; \Delta}{\Gamma \vdash \omega_2 t : A \vee B; \Delta}\vee_i^2$$

$$\frac{\Gamma \vdash t : A \vee B; \Delta \quad \Gamma, x : A \vdash u : C; \Delta \quad \Gamma, y : B \vdash v : C; \Delta}{\Gamma \vdash (t\ \ [x.u, y.v]) : C; \Delta}\vee_e$$

$$\frac{\Gamma \vdash t : A; \Delta, a : A}{\Gamma \vdash (a\ \ t) : \bot; \Delta, a : A}\bot_i \qquad \frac{\Gamma \vdash t : \bot; \Delta, a : A}{\Gamma \vdash \mu a.t : A; \Delta}\mu$$

4. *The cut-elimination procedure corresponds to the reduction rules given below. They are those we need to the subformula property.*

   - $(\lambda x.u\ \ v) \rhd_\beta u[x := v]$
   - $(\langle t_1, t_2\rangle\ \ \pi_i) \rhd_{\pi_i} t_i$
   - $(\omega_i t\ \ [x_1.u_1, x_2.u_2]) \rhd_D u_i[x_i := t]$
   - $((t\ \ [x_1.u_1, x_2.u_2])\ \ \varepsilon) \rhd_\delta (t\ \ [x_1.(u_1\ \varepsilon), x_2.(u_2\ \varepsilon)])$
   - $(\mu a.t\ \ \varepsilon) \rhd_\mu \mu a.t[a :=^* \varepsilon]$.

   where $t[a :=^* \varepsilon]$ is obtained from t by replacing inductively each subterm in the form $(a\ v)$ by $(a\ (v\ \varepsilon))$.

5. *Let ε and ε' be $\mathcal{E}$-terms. The notation $\varepsilon \rhd \varepsilon'$ means that ε reduces to ε' by using one step of the reduction rules given above. Similarly, $\varepsilon \rhd^* \varepsilon'$ means that ε reduces to ε' by using some steps of the reduction rules given above.*

6. *Let ε and ε' be $\mathcal{E}$-terms and r a redex of ε. The notation $\varepsilon \rhd^r \varepsilon'$ means that ε reduces to ε' by reducing the redex r.*

2.3 Characterization of the $\lambda\mu^{\wedge\vee}$-terms

In this section we develop some results present in [7]. The presence of the critical pairs for the standardization, and the fact that a $\lambda\mu^{\wedge\vee}$-term can have more than one head, contrary to the λ-calculus or the $\lambda\mu$-calculus need such results before defining the notion of standardization, head and leftmost reductions. For simplicity of proofs, in the rest of this work we will consider only typed terms and thus, for example, that terms such as $(\langle u, v\rangle\, [x, p.y, q])$ are not allowed since they, obviously, can not be typed.

Definition 2.3.1 1. *A term t is said to be simple if it is a variable or an application.*

2. *Let $*_1, ..., *_n, ...$ be an infinite number of holes. A general context $\mathbf{C}$ is a term with holes. The set of general contexts is given by the following grammar:*

$$\mathbf{C} := *_{i_j}\ \mid \lambda x.\mathbf{C}\ \mid \langle \mathbf{C_1}, \mathbf{C_2}\rangle\ \mid \omega_1\mathbf{C}\ \mid \omega_2\mathbf{C}\ \mid \mu a.\mathbf{C}$$

We consider only general contexts with different holes, i.e. if $*_{i_1}, ..., *_{i_n}$ are the holes of a general context $\mathbf{C}$, then, $*_{i_p} \neq *_{i_q}$ for each $p \neq q$.

3. *Let* $\mathbf{C}$ *be a general context with holes* $*_{i_1}, ..., *_{i_n}$ *and* f *a bijection from* $\mathbb{N}$ *to* $\mathbb{N}$*, then, the general context* $f(\mathbf{C})$ *is the context* $\mathbf{C}$ *with the holes* $*_{f(i_1)}, ..., *_{f(i_n)}$*, i.e.* $f(\mathbf{C})$ *is the general context* $\mathbf{C}$ *just with a different enumeration of its holes.*

4. *Let* $\mathbf{C}$ *and* $\mathbf{C}'$ *be two general contexts, we said that* $\mathbf{C}$ *is equivalent to* $\mathbf{C}'$ *and denote this by* $\mathbf{C} \simeq \mathbf{C}'$*, if there exists a bijection* f *from* $\mathbb{N}$ *to* $\mathbb{N}$ *such that* $\mathbf{C}' = f(\mathbf{C})$*. Thus, if* $\mathbf{C}' \simeq \mathbf{C}$*, then,* $\mathbf{C}$ *and* $\mathbf{C}'$ *have the same number of holes.*

5. *A context is an equivalent class for the previous equivalent relation. Then we can always suppose that the* n *holes of a context are* $*_1, ..., *_n$ *in this given order.*

6. *If* $\mathbf{C}$ *is a context with holes* $*_1, ..., *_n$ *and* $t_1, ..., t_n$ *are terms, then* $\mathbf{C}[t_1, ..., t_n]$ *is the term obtained by replacing each* $*_i$ *by* t_i*. The free variables of* t_i *can be captured in the term* $\mathbf{C}[t_1, ..., t_n]$*.*

Lemma 2.3.1 *Let* $t_1, ..., t_n, t'_1, ..., t'_m$ *be simple terms and* $\mathbf{C}, \mathbf{C}'$ *two contexts. If* $\mathbf{C}[t_1, ..., t_n] = \mathbf{C}'[t'_1, ..., t'_m]$*, then,* $\mathbf{C} = \mathbf{C}'$*.*

Proof. By induction on $\mathbf{C}$. ■

Lemma 2.3.2 *Each term* t *can be uniquely written as* $\mathbf{C}[t_1, ..., t_n]$*, where* $\mathbf{C}$ *is a context and* $t_1, ..., t_n$ *are simple terms.*

Proof. By induction on t.

- If $t = x$, $t = (u\ \varepsilon)$ or $t = (a\ u)$, then $\mathbf{C} = *_1$, $t_1 = t$ and $t = \mathbf{C}[t_1]$.

- If $t = \lambda y.u$, then, by the induction hypothesis, $u = \mathbf{C}'[u_1, ..., u_n]$. Hence $\mathbf{C} = \lambda y.\mathbf{C}'$ and $t = \mathbf{C}[u_1, ..., u_n]$.

- The cases $t = \omega_i u$ and $t = \mu a.u$ are similar to the previous case.

- If $t = \langle u, v\rangle$, then, by the induction hypothesis, $u = \mathbf{C}_1[u_1, ..., u_n]$ and $v = \mathbf{C}_2[v_1, ..., v_m]$. We can suppose that the holes of $\mathbf{C}_1$ and $\mathbf{C}_2$ are differents. Hence $\mathbf{C} = \langle \mathbf{C}_1, \mathbf{C}_2\rangle$ and $t = \mathbf{C}[u_1, ..., u_n, v_1, ..., v_m]$.

For uniqueness, it suffices to use the lemma 2.3.1. ■

Definition 2.3.2 *1. A sequence $\bar{w} = w_1...w_n$ of $\mathcal{E}$-terms is said to be nice iff w_n is the only $\mathcal{E}$-term which can be in the form $[x.u, y.v]$.*

2. A sequence $\bar{w} = w_1...w_n$ of $\mathcal{E}$-terms is said to be good iff w_i is not in the form $[x.u, y.v]$ for each $1 \leq i \leq n$. Observe that any good sequence is a nice one but not vice versa.

3. A sequence $\bar{w} = w_1...w_n$ is said to be normal iff each w_i is normal. An $\mathcal{E}$-term of the form $[x.u, y.v]$ is normal iff u and v are normal.

4. Let $\bar{w} = w_1...w_i...w_n$, then $\bar{w} \triangleright \bar{w}'$ iff $\bar{w}' = w_1...w_i'...w_n$ where $w_i \triangleright w_i'$.

Lemma 2.3.3 (*and definition*) *Let t be a simple term.*

1. The term t can be uniquely written as one of the figures below where $\bar{w} = w_1...w_n$ is a finite sequence of $\mathcal{E}$-terms.

2. A head of t (denoted as $hd(t)$) defined by the figures below, is either a redex or a variable. In the cases (5) *and* (6) *if $\varepsilon\bar{w}$ is not nice, then $\varepsilon\bar{w} = \bar{r}\,[y.u, z.v]w_i\bar{s}$.*

	t	*hd (t)*
1	$(a\ v)$	a
2	$((\lambda x.u\ v)\ \bar{w})$	$(\lambda x.u\ v)$
3	$((\langle u_1, u_2\rangle\ \pi_i)\ \bar{w})$	$(\langle u_1, u_2\rangle\ \pi_i)$
4	$((\omega_i v\ [x_1.u_1, x_2.u_2])\ \bar{w})$	$(\omega_i v\ [x_1.u_1, x_2.u_2])$
5	$((\mu a.u\ \varepsilon)\ \bar{w})$	$(\mu a.u\ \varepsilon)$ *or* *any permutative redex in the form:* $((\mu a.u\ \bar{r}[y.u, z.v])\ w_i\bar{s})$
6	$((x\,\varepsilon)\ \bar{w})$	x *if the sequence $\varepsilon\bar{w}$ is nice,* *else, any permutative redex in the form:* $((x\ \bar{r}[y.u, z.v])\ w_i\bar{s})$

Proof. By induction on the simple term t. We have either t is a variable which gives the case (1) or (6) (with empty sequences v and $\varepsilon\bar{w}$), either $t = (u\ \varepsilon)$. Therefore we will examine the form of u.

- If u is not a simple term, then $u = \lambda x.v$, $u = \langle u_1, u_2\rangle$, $u = \omega_i v$ or $u = \mu a.v$. All these forms give us that t is respectively in the case (2), (3), (4) or (5).

- If u is a simple term, then the induction hypothesis concludes.

The uniqueness is clear. ■

Remark 2.3.1 *Observe that simple terms of the form* $t = ((x\,\varepsilon)\,\bar{w})$ *and* $t = ((\mu a.u\,\varepsilon)\,\bar{w})$ *can have more than one head. For the first one, it suffices that* $\varepsilon\bar{w}$ *contains more than one term in the form* $[y.p, z.q]$*, for the second, it is enough that* $\varepsilon\bar{w}$ *is not nice.*

The precedent lemma allows the $\lambda\mu^{\wedge\vee}$-term to be characterized in the following corollary, this characterization will be useful for the standardization theorem section.

Corollary 2.3.1 *Any term* t *can be written in the following form:* $t = (T\ \bar{w})$*, where* $T = x$*,* a*,* $\lambda x.u$*,* $\langle u_1, u_2\rangle$*,* $\omega_i u$ *or* $\mu a.u$ *and* $\bar{w}$ *is a finite sequence of* $\mathcal{E}$*-terms possibly empty.*

Proof. A direct consequence of the precedent lemma. ■

Lemma 2.3.4 *Let* $t = \mathbf{C}[t_1, ..., t_i, ..., t_n]$ *be a term and* r *a redex of* t_i*. If* $t \rhd^r t'$*, then* $t_i \rhd^r \mathbf{C}'_i[t_1^i, ..., t_m^i]$ *and* $t' = \mathbf{C}'[t_1, ..., t_{i-1}, t_1^i, ..., t_m^i, t_{i+1}, ..., t_n]$ *where* $\mathbf{C}' = \mathbf{C}[*_i := \mathbf{C}_i]$*.*

Proof. By induction on $\mathbf{C}$. ■

2.4 Standardization theorem

The standardization theorem is a useful result stating that if $t \rhd^* t'$, then there is a reduction sequence from t to t' *"standard"* in the sense that contractions are made from left to right, possibly with some jumps in between. One of its consequences is that normal forms, if existing, can be reached by leftmost reduction sequences.

Definition 2.4.1 *1. Let* $\bar{w} = \bar{r}[x.p, y.q]\varepsilon_i\bar{s}$ *be a finite sequence of* $\mathcal{E}$*-terms, we define a new reduction relation* $\succ$ *by:* $\bar{w} \succ \bar{r}[x.(p\,\varepsilon_i), y.(q\,\varepsilon_i)]\bar{s}$*, where* $\bar{r}$ *and* $\bar{s}$ *are possibly empty. As usual* $\succ^*$ *denotes the reflexive and transitive closure of* $\succ$*.*

2. Let R *be a simple term in the form* $(\lambda x.u\,v)$ *(resp* $(\langle u_1, u_2\rangle\ \pi_i)$*,*$(\mu a.u\ \theta)$*,* $(\omega_i u\ [x_1.v_1, x_2.v_2])$*,* x*,* a *), we denote by* r *its possible reductom* $u[x := v]$ *(resp* u_i*,* $\mu a.u[a :=^* \theta]$*,* $v_i[x_i := u]$*).*

Let also ε*,* ε' *be two* $\mathcal{E}$*-terms, we define* $\varepsilon \rhd^*_{st} \varepsilon'$ *a standard sequence reductions from* ε *to* ε' *by induction on the ordered lexicographic pair* $\kappa(\varepsilon \rhd^*_{st} \varepsilon') = (l, c)$ *where* l *stands for the length of the reduction* $\varepsilon \rhd^*_{st} \varepsilon'$ *and* c *the complexity of* ε*. We use simultaneously the following abbreviations:*

$\bar{w} = \varepsilon_1...\varepsilon_n \rhd^*_{st} \bar{w}' = \varepsilon'_1...\varepsilon'_n$ *means that* $\varepsilon_i \rhd^*_{st} \varepsilon'_i$ *for each* $1 \leq i \leq n$*.*

(C_λ) *If* $\varepsilon = (\lambda x.u\ \bar{w})$, *then* $\varepsilon' = (\lambda x.u'\ \bar{w}')$ *with* $u \rhd_{st}^* u'$ *and* $\bar{w} \rhd_{st}^* \bar{w}'$.

(C_μ) *If* $\varepsilon = (\mu a.u\ \bar{w})$, *then* $\varepsilon' = (\mu a.u'\ \bar{w}')$ *with* $u \rhd_{st}^* u'$ *and* $\bar{w} \rhd_{st}^* \bar{w}'$.

(C_π) *If* $\varepsilon = (\langle u_1, u_2\rangle\ \bar{w})$, *then* $(\varepsilon' = \langle u_1', u_2'\rangle\ \bar{w}')$ *with* $u_i \rhd_{st}^* u_i'$ *and* $\bar{w} \rhd_{st}^* \bar{w}'$.

(C_ω) *If* $\varepsilon = (\omega_i u\ \bar{w})$, *then* $\varepsilon' = (\omega_i\ u'\bar{w}')$ *with* $u \rhd_{st}^* u'$ *and* $\bar{w} \rhd_{st}^* \bar{w}'$.

(V_λ) *If* $\varepsilon = (x\ \bar{w})$, *then* $\varepsilon' = (x\ \bar{w}')$ *with* $\bar{w} \rhd_{st}^* \bar{w}'$.

(V_μ) *If* $\varepsilon = (a\,u)$, *then* $\varepsilon' = (a\,u')$ *with* $u \rhd_{st}^* u'$.

- *If* $\varepsilon = (R\,\bar{w})$, *then:*

  ($\Rightarrow$) *Either* $\varepsilon \rhd (r\,\bar{w}) \rhd_{st}^* \varepsilon'$,

  ($\succ_\delta$) *Either* $\varepsilon \rhd (R\,\bar{w}') \rhd_{st}^* \varepsilon'$ *with* $\bar{w} \succ \bar{w}'$.

  ($\succ_\mu$) *Either* $\varepsilon \rhd (\mu a.u\,\bar{w}') \rhd_{st}^* \varepsilon'$ *with* $\theta\bar{w} \succ \bar{w}'$, *and this only if* $R = (\mu a.u\ \theta)$.

  ($\succ_\omega$) *Or* $\varepsilon \rhd (\omega_i u\,\bar{w}') \rhd_{st}^* \varepsilon'$ *with* $[x_1.v_1, x_2.v_2]\bar{w} \succ \bar{w}'$, *and this only if* $R = (\omega_i u\ [x_1.v_1, x_2.v_2])$.

- *If* $\varepsilon = \pi_i$, *then* $\varepsilon' = \pi_i$.

- *If* $\varepsilon = [x_1.u_1, x_2.u_2]$, *then* $\varepsilon' = [x_1.u_1', x_2.u_2']$ *with* $u_i \rhd_{st}^* u_i'$.

Remark 2.4.1 *1. In the rules* $C_\lambda, ..., C_\omega$ *the sequence* $\bar{w}$ *is possibly empty and this corresponds to the cases where* ε *is not a simpe term.*

2. *Intuitively the standard reduction contract redexes from the external to the internal and from left to right. For* λ*-calculus, a standard reduction from the term* $(\lambda x.u\,v)w_1...w_n$ *either start by reducing the head redex* $(\lambda x.u\,v)$, *else this one will never be converted after performing* u, v *or the arguments* w_i, *i.e. from external to internal. There is also another condition, standard reduction from the term* $((x\,u)\,v)$ *has to reduce firstly in* u *and then in* v, *in this given order from left to right, this last notion is not captured by the definition above since we consider that reduction in* u *and* v *are independent of each other.*

3. *Given a* $\lambda\mu^{\wedge\vee}$*-term of the form* $(R\,\bar{w})$ *standard reduction strategies can start with various permutations using the rule* $\succ_\delta$ ($\succ_\mu$, $\succ_\omega$ *according the form of* R) *until either we obtain a nice sequence and then reducing the possible head redex by the rule* $\Rightarrow$ *or starts performing the subterms of the arguments using the rules* V_λ, V_μ *or the rules* $C_\lambda, ..., C_\omega$.

Lemma 2.4.1 *Assume that* $\varepsilon \rhd_{st}^* \varepsilon'$.

1. *If* $v \rhd_{st}^* v'$, *then* $\varepsilon[x := v] \rhd_{st}^* \varepsilon'[x := v']$.

2. *If* $\theta \rhd_{st}^* \theta'$, *then* $\varepsilon[a :=^* \theta] \rhd_{st}^* \varepsilon'[a :=^* \theta']$.

Proof. Only the second assertion will be treated (a similar proof for the first). By induction on $\kappa(\varepsilon \rhd^*_{st} \varepsilon')$. We give just the case where the substitution intervenes. If $\varepsilon = (a\ u)$, then $\varepsilon' = (a\ u')$ where $u \rhd^*_{st} u'$, thus, $\varepsilon[a :=^* \theta] = (a\ (u[a :=^* \theta]\ \theta))$ and $\varepsilon'[a :=^* \theta'] = (a\ (u'[a :=^* \theta']\ \theta'))$. By the induction hypothesis, we have $u[a :=^* \theta] \rhd^*_{st} u'[a :=^* \theta']$, then by definition, $(a\ (u[a :=^* \theta]\ \theta)) \rhd^*_{st} (a\ (u'[a :=^* \theta']\ \theta'))$. ■

Theorem 2.4.1 (Standardization theorem) *If $\varepsilon \rhd^* \varepsilon'$, then $\varepsilon \rhd^*_{st} \varepsilon'$.*

Proof. By induction on the length of the reduction $\varepsilon \rhd^* \varepsilon'$ it suffices to prove the following lemma. ■

Lemma 2.4.2 *If $\varepsilon \rhd^*_{st} \varepsilon' \rhd \varepsilon''$, then $\varepsilon \rhd^*_{st} \varepsilon''$.*

Proof. By induction on $\kappa(\varepsilon \rhd^*_{st} \varepsilon')$, we examine how $\varepsilon \rhd^*_{st} \varepsilon'$ following the different forms of ε.

– The cases where ε is not a simple term are a direct consequences of the induction hypothesis (decreasing of c).

– If ε is a simple term, we will examine some cases, the others are similar or more simpler.

(C_λ) Let $\varepsilon = (\lambda x.u\ \bar{w}) \rhd^*_{st} (\lambda x.u'\ \bar{w}') = \varepsilon' \rhd \varepsilon''$, with $u \rhd^*_{st} u'$ and $\bar{w} \rhd^*_{st} \bar{w}'$ (here of course $\bar{w}$ is not empty since ε is simple). We distinguish three cases:

- If $\varepsilon'' = (\lambda x.u''\ \bar{w}')$ (resp $(\lambda x.u'\ \bar{w}'')$), where $u' \rhd u''$ (resp $\bar{w}' \rhd \bar{w}''$) then the induction hypothesis concludes.
- If $\varepsilon'' = (u'[x := v']\, w'_2 ... w'_n)$, then $\bar{w} = v w_2 ... w_n$ where $v \rhd^*_{st} v'$, $w_i \rhd^*_{st} w'_i$ and $\bar{w}' = v' w'_2 ... w'_n$. Therefore by the lemma 2.4.1, $u[x := v] \rhd^*_{st} u'[x := v']$, thus $(u[x := v]\, w_2 ... w_n) \rhd^*_{st} (u'[x := v']\, w'_2 ... w'_n)$. Finally by the rule $(\Rightarrow)$, we have $\varepsilon \rhd (u[x := v]\, w_2 ... w_n) \rhd^*_{st} (u'[x := v']\, w'_2 ... w'_n)$ is a standard sequence of reductions.
- The last case is $\varepsilon'' = (\lambda x.u'\ \bar{w}'')$ where $\bar{w}' \succ \bar{w}''$. Therefore $\bar{w} = w_1 ... [y.p, z.q] w_i ... w_n \rhd^*_{st} w'_1 ... [y.p', z.q'] w'_i ... w'_n$ and then $\bar{w}'' = w'_1 ... [y.(p'\, w'_i), z.(q'\, w'_i)] ... w'_n$. We have $[y.(p\, w_i), z.(q\, w_i)] \rhd^*_{st} [y.(p'\, w'_i), z.(q'\, w'_i)]$, hence the rule $(\succ_\delta)$ allows to conclude: $\varepsilon \rhd (\lambda x.u\, w_1 ... [y.(p\, w_i), z.(q\, w_i)] ... w_n) \rhd^*_{st} (\lambda x.u'\, w'_1 ... [y.(p'\, w'_i), z.(q'\, w'_i)] ... w'_n) = \varepsilon''$.

(C_μ) , (C_π) and (C_ω) are similar to the previous case.

(V_λ) Let $\varepsilon = (x\, \bar{w})$, then $\varepsilon = (x\, \bar{w}) \rhd^*_{st} \varepsilon' = (x\, \bar{w}') \rhd (x\, \bar{w}'') = \varepsilon''$ then,

- Either $\bar{w} = w_1 ... w_i ... w_n \rhd^*_{st} w'_1 ... w'_i ... w'_n = \bar{w}' \rhd w'_1 ... w''_i ... w'_n = \bar{w}''$, hence by the induction hypothesis $w_i \rhd^*_{st} w''_i$. Therefore, we have $\bar{w} \rhd^*_{st} \bar{w}''$ and $\varepsilon = (x\, \bar{w}) \rhd^*_{st} (x\, \bar{w}'') = \varepsilon''$.

- Or $\bar{w} = w_1...[y.p, z.q]w_i...w_n \triangleright^*_{st} w'_1...[y.p', z.q']w'_i...w'_n = \bar{w}' \succ w'_1...[y.(p'\, w'_i), z.(q'\, w'_i)]...w'_n = \bar{w}''$. We have $[y.(p\, w_i), z.(q\, w_i)] \triangleright^*_{st} [y.(p'\, w'_i), z.(q'\, w'_i)]$, therefore by the rule $(\succ_\delta)$, $(x\, \bar{w}) \triangleright (x\, w_1...[y.(p\, w_i), z.(q\, w_i)]...w_n) \triangleright^*_{st} (x\, w'_1...[y.(p'\, w'_i), z.(q'\, w'_i)]...w'_n) = (x\, \bar{w}'')$ is a standard sequence of reductions.

(V_μ) Is a direct consequence of the induction hypothesis (decreasing of c).

$(\Rightarrow)$, $(\succ \delta)$, $(\succ_\mu)$ and $(\succ^\omega)$ are direct consequences of the application of the induction hypothesis (decreasing of l).

■

Remark 2.4.2 *1. In [13], F. Joachimski and R. Matthes presented the ΛJ-calculus which is an extension of the λ-calculus by a generalized application that gives rise to permutative reductions, the resulting rewriting system is not orthogonal as the $\lambda\mu^{\wedge\vee}$-calculus. The definition of the standardization established here produced exactly the same treatment of the logical and the classical reductions ($\triangleright_\beta$, $\triangleright_{\pi_i}$, $\triangleright_D$ and $\triangleright_\mu$) like that of β-reduction in [13], this means that to avoid difficulties relative to the μ-reduction, we need to treat it similarly to the β-reduction. The ΛJ-calculus has served us as a model for studying our $\lambda\mu^{\wedge\vee}$-calculus. In fact, concerning the standardization, the restriction of the $\lambda\mu^{\wedge\vee}$-calculus to the $\lambda\mu$-one can serve as a model for the study of the rewriting system with permutative and structural reductions, as well as the ΛJ-calculus serves as a minimal model for the study of term rewriting systems with permutation.*

2. In the definition 2.4.1, we can be more restrictive, by this we mean that, we have also a standardization theorem when replacing the rules $\succ_\delta$ and $\succ_\omega$ by:

$(\succ_\lambda)$ $\varepsilon = ((\lambda x.u\, v)\, \bar{w}) \triangleright ((\lambda x.u\, v)\, \bar{w}') \triangleright^*_{st} ((\lambda x.u'\, v')\, \bar{w}''') = \varepsilon'$ *with* $u \triangleright^*_{st} u'$, $v \triangleright^*_{st} v'$ *and* $\bar{w} \succ \bar{w}' \succ^* \bar{w}'' \triangleright^*_{st} \bar{w}'''$.

$(\succ_\pi)$ $\varepsilon = ((\langle u_1, u_2\rangle\, \pi_i)\, \bar{w}) \triangleright ((\langle u_1, u_2\rangle\, \pi_i)\, \bar{w}') \triangleright^*_{st} ((\langle u'_1, u'_2\rangle\, \pi_i)\, \bar{w}''') = \varepsilon'$ *with* $u_i \triangleright^*_{st} u'_i$ *and* $\bar{w} \succ \bar{w}' \succ^* \bar{w}'' \triangleright^*_{st} \bar{w}'''$.

$(\succ_D)$ $\varepsilon = ((\omega_i u\, [x_1.v_1, x_2.v_2])\, \bar{w}) \triangleright ((\omega_i u\ \bar{w}') \triangleright^*_{st} (\omega_i u'\, \bar{w}''') = \varepsilon'$ *with* $u \triangleright^*_{st} u'$ *and* $[x_1.v_1, x_2.v_2]\bar{w} \succ \bar{w}' \succ^* \bar{w}'' \triangleright^*_{st} \bar{w}'''$.

This can be explained by the fact that:

- *For the rules $\succ_\lambda$ and $\succ_\pi$ there are no interactions between v, π_i and $\bar{w}$ via permutative reductions.*
- *For the rule $\succ_D$ even there are interactions between $[x_1.v_1, x_2.v_2]$ and $\bar{w}$ via permutative reductions, after the $\triangleright_D$-reduction we will always get $(v_i\, \bar{w})[x_i := u] = (v_i[x_i := u]\, \bar{w})$ since x_i is not free in $\bar{w}$.*

Contrarly for example to the case $\varepsilon = ((\mu a.u\,[x.p, y.q])\,[r.k, s.l]\theta)$ *in which there are more complications: suppose that we give the priority to the classical redex, i.e. if it is not converted in the beginning then it will be never performed, thus the lemma 2.4.2 does not hold for this sequence of reductions:* $\varepsilon \rhd ((\mu a.u\,[x.p, y.q])\,[r.(k\,\theta), s.(l\,\theta)]) \rhd$ $(\mu a.u\,[x.(p\,[r.(k\,\theta), s.(l\,\theta)]), y.(q\,[r.(k\,\theta), s.(l\,\theta)])]) \rhd^*_{st} (\mu a.u\,[x.P', y.Q']) = \varepsilon'$ $\rhd \mu a.u[a :=^* [x.P', y.Q']]] = \varepsilon''$, *where* $(p\,[r.(k\,\theta), s.(l\,\theta)]) \rhd^*_{st} P'$ *and* $(q\,[r.(k\,\theta), s.(l\,\theta)]) \rhd^*_{st} Q'$, *for the simple reason which is: we do not know how these two standard sequence reductions are made* $(p\,[r.(k\,\theta), s.(l\,\theta)]) \rhd^*_{st} P'$ *and* $(q\,[r.(k\,\theta), s.(l\,\theta)]) \rhd^*_{st} Q'$ *(it depends on* p *and* q*). To resolve this probleme we have to consider the rule* $\succ_\mu$.

2.5 Head and leftmost reductions

Definition 2.5.1 *1. A one step head reduction of a simple term* t *consists in reducing a head redex if any. We denote* $t \rhd_{hd} t'$ *if* t *is reduced to* t' *by a head reduction.*

2. *A one step head reduction of a term* $t = \mathbf{C}[t_1, ..., t_n]$ *corresponds to a one step head reduction of one of the simple terms* t_i $(1 \leq i \leq n)$.

3. *We denote by* $\rhd^*_{hd}$ *the reflexive and transitive closure of* $\rhd_{hd}$.

4. *A simple head normal form is a simple term in the form* $(x\ \bar{w})$ *or* $(a\ u)$, *the elements of the sequence* $\bar{w}$ *(resp* u*) are called the arguments of the head variable* x *(resp* a*). The sequences* $\bar{w}$ *is a nice one, and these are the only cases where we cannot reduce in the head, because there is no head since the arguments of the sequences cannot interacte between them via commutative reductions.*

5. *A head normal form is a term in the form* $\mathbf{C}[t_1, ..., t_n]$ *where all the* t_i *are simple head normal forms.*

Remark 2.5.1 *Observe that there is no unicity of "the" head normal form. Take the simple term* $t = ((x\,[y.u, z.v])\,[r.p, s.q]\epsilon)$, *then* $t \rhd^*_{hd} (x\,[y.((u\,[r.p, s.q])\,\epsilon), z.((v\,[r.p, s.q])\,\epsilon)]) = t_1$, $t \rhd^*_{hd} (x\,[y.(u\,[r.(p\,\epsilon), s.(q\,\epsilon)]), z.(v\,[r.(p\,\epsilon), s.(q\,\epsilon)])]) = t_2$, *and both of* t_1 *and* t_2 *are head normal froms. For this reason we will define an extra head reduction, which will be exactly the same on all the terms except of course on terms in the forms* $(x\,\bar{w})$ *and* $((\mu a.u\,\theta)\,\bar{w})$.

Definition 2.5.2 *1. The outer most redex of a simple term of the form* $((\mu a.u\,\theta)\,\bar{w})$ *(resp* $(x\,\bar{w})$*) is the classical redex* $(\mu a.u\,\theta)$ *(resp the possible permutative* $(x\,\bar{R}[y.p, z.q])\epsilon\bar{S}$, *where* $\bar{w} = \bar{R}[y.p, z.q]\epsilon\bar{S}$ *and* $\epsilon\bar{S}$ *is a nice sequence*).

2. *Let t be a term,*

   - *If t is a simple one, then*
     (a) *If t is of the form $(x\,\bar{w})$ or $((\mu a.u\,\theta)\,\bar{w})$, then the one step extra head reduction of t consists in reducing the outer most redex.*
     (b) *Else it consists to the head reduction of t.*
     (c) *We denote $t \rhd_{ehd} t'$ if t is reduced to t' by the extra head reduction*
   - *Else $t = \mathbf{C}[t_1, ..., t_n]$, then a one step extra head reduction of t, corresponds to the one step extra head reduction of one of the simple terms t_i.*

3. *We denote by $\rhd^*_{ehd}$ the reflexive and transitive closure of $\rhd_{ehd}$.*

4. *A simple extra head normal form is a simple term of the form $(x\,\bar{w})$ or $(a\,u)$, where $\bar{w}$ is nice and if the last $\mathcal{E}$-term w_n is in the form $[i.(u\,\bar{s}), j.(v\,\bar{s})]$, then $\bar{s}$ is a nice sequence (it can be empty also).*

5. *An extra head normal form is a term in the form $\mathbf{C}[t_1, ..., t_n]$, where all the t_i are simple extra head normal forms.*

Proposition 2.5.1 (The diamond property of the extra head reduction) *Let t, u and v be terms such that $t \rhd_{ehd} u$ and $t \rhd_{ehd} v$, where $u \neq v$, then there exists w such that: $u \rhd_{ehd} w$ and $v \rhd_{ehd} w$.*

Proof. We can suppose that $t = \mathbf{C}[t_1, .., t_i, ..., t_j, ..., t_n]$, $t_i \rhd_{ehd} \mathbf{C}'_i[t^i_1, ..., t^i_p]$, $t_j \rhd_{ehd} \mathbf{C}'_j[t^j_1, ..., t^j_q]$, $u = \mathbf{C}_1[t_1, ..., t^i_1, ..., t^i_p, ..., t_j, ..., t_n]$ and $v = \mathbf{C}_2[t_1, .., t_i, ..., t^j_1, ..., t^j_q, ..., t_n]$ where $\mathbf{C}_1 = \mathbf{C}[*_i := \mathbf{C}'_i]$ and $\mathbf{C_2} = \mathbf{C}[*_\mathbf{j} := \mathbf{C}'_\mathbf{j}]$. Therefore it suffices to reduce in u the simple term t_j and reduce in v the simple term t_i to construct the commun redectum $w = \mathbf{C}'[t_1, ..., t^i_1, ..., t^i_p, ..., t^j_1, ..., t^j_q, ..., t_n]$, where $\mathbf{C}' = \mathbf{C}[*_i := \mathbf{C}'_i, *_j := \mathbf{C}'_j]$. ■

Definition 2.5.3 *Let t and t' be two terms such that $t \rhd^*_{ehd} t'$, then $|t, t'|$ denotes the length of an extra head reduction which leads t to t'.*

Proposition 2.5.2 (Confluence of the extra head reduction) *Let t, t_1 and t_2 be terms such that $t \rhd^*_{ehd} t_1$ and $t \rhd^*_{ehd} t_2$, then there exists t_3 such that $t_1 \rhd^*_{ehd} t_3$ and $t_2 \rhd^*_{ehd} t_3$ and $|t_1, t_3| = |t, t_2|$, and $|t_2, t_3| = |t, t_1|$.*

Proof. A direct consequence of the diamond property. ■

One of the important consequences of confluence is the uniqueness of the normal form.

Corollary 2.5.1 *If an extra head reduction of a term t finishes by a term τ, then τ is the unique extra head normal form of t.*

Proof. Directly from the previous proposition. ■

Definition 2.5.4 *Let $u_1, ..., u_n$ be terms and $\varepsilon, \varepsilon_1, ..., \varepsilon_m$ $\mathcal{E}$-terms, then $\varepsilon[(x_i := u_i)_{1\leq i\leq n}; (a_j :=^* \varepsilon_j)_{1\leq j\leq m}]$ is obtained from the $\mathcal{E}$-term ε by replacing inductively each x_i by u_i and each subterm in the form $(a_j\, u)$ in ε by $(a_j\, (u\, \varepsilon_j))$.*

The followings lemmas show that, to carry out an extra head reduction of $t\sigma$ it is equivalent (the same result and the same number of steps) to carry out a certain number of steps of an extra head reduction of t, which gives t', then to make an extra head reduction of $t'\sigma$.

Lemma 2.5.1 *Let t be a simple term, t' a term and σ a substitution. If $t \rhd_{ehd} t'$, then $t\sigma \rhd_{ehd} t'\sigma$.*

Proof. Easy. ■

Lemma 2.5.2 *Let t, t' be terms and σ a substitution. If $t \rhd^*_{ehd} t'$, then, $t\sigma \rhd^*_{ehd} t'\sigma$ and $|t, t'| = |t\sigma, t'\sigma|$.*

Proof. By induction on the length of the reduction $t \rhd^*_{ehd} t'$ and we use the previous lemma. ■

Definition 2.5.5 *On the set of terms we define an equivalence relation $\simeq_{ehd}$ by $t \simeq_{ehd} t'$ iff there exists t'' such that, $t \rhd^*_{ehd} t''$ and $t' \rhd^*_{ehd} t''$.*

Corollary 2.5.2 *Let t, t' be terms and σ a substitution. If $t \simeq_{ehd} t'$, then, $t\sigma \simeq_{ehd} t'\sigma$.*

Proof. Directly from the previous lemma. ■

Lemma 2.5.3 *Let t be a simple term, t' a term and ε an $\mathcal{E}$-term, such that $t \rhd_{ehd} t'$. Then there exists a term τ such that $(t\, \varepsilon) \rhd^*_{ehd} \tau$, $(t'\, \varepsilon) \rhd^*_{ehd} \tau$ and $|(t\, \varepsilon), \tau| = |t, t'| + |(t'\, \varepsilon), \tau|$.*

Proof. Cases are examinated following the form of the simple term t, here $|t, t'| = 1$. In what follows $\bar{S}$ is a nice sequence of $\mathcal{E}$-terms and $\bar{T}$ a good one.

– If $t = (x\, \bar{w}) = (((x\, \bar{R})[y.p, z.q])w_i\bar{S}) \rhd_{ehd} ((x\, \bar{R})[y.(p\, w_i), z.(q\, w_i)]\bar{S}) = t'$,

  - If $\bar{S}$ is a good sequence, then $(t\, \varepsilon) \rhd_{ehd} (t'\, \varepsilon)$. We just take $\tau = (t'\, \varepsilon)$, so we have $|(t\, \varepsilon), \tau| = |t, t'| + |(t'\, \varepsilon), \tau|$.
  - If $\bar{S}$ is not a good sequence, then $\bar{S} = \bar{T}[r.K, s.L]$. This implies that $(t\, \varepsilon) \rhd_{ehd} ((x\, \bar{R})[y.p, z.q]w_i\bar{T}[r.(K\, \varepsilon), s.(L\, \varepsilon)]) \rhd_{ehd}$ $((x\, \bar{R})[y.(p\, w_i), z.(q\, w_i)]\bar{T}[r.(K\, \varepsilon), s.(L\, \varepsilon)])$, hence take $\tau = ((x\, \bar{R})[y.(p\, w_i), z.(q\, w_i)]\bar{T}[r.(K\, \varepsilon), s.(L\, \varepsilon)])$ and check that $(t'\, \varepsilon) \rhd_{ehd} \tau$. This provides also that $|(t\, \varepsilon), \tau| = |t, t'| + |(t'\, \varepsilon), \tau|$.

– The others are more simpler.

■

Lemma 2.5.4 *Let t, t' and ε be $\mathcal{E}$-terms, such that $t \rhd^*_{ehd} t'$. Then there exists a term τ such that $(t\,\varepsilon) \rhd^*_{ehd} \tau$, $(t'\,\varepsilon) \rhd^*_{ehd} \tau$ and $|(t\,\varepsilon), \tau| = |t, t'| + |(t'\,\varepsilon), \tau|$.*

Proof. By induction on the length of the reduction $t \rhd^*_{ehd} t'$, it suffices to examine the case where $|t, t'| = 1$. Therefore we process by induction on $\mathbf{C}$, where $t = \mathbf{C}[t_1, ..., t_i, ..., t_n]$.

- If $\mathbf{C} = *_i$, then t is a simple term and the result is by the previous lemma.

- If $\mathbf{C} = \mu a.\mathbf{C_1}$, then $t = \mu a.\mathbf{C}_1[t_1, ..., t_i, ..., t_n] \rhd_{ehd} \mu a.\mathbf{C}'_1[t_1, ..., t^i_1, ..., t^i_p, ..., t_n]$ $= t'$, where $t_i \rhd_{ehd} \mathbf{C}_i[t^i_1, ..., t^i_p] = t'_i$ and $\mathbf{C}'_1 = \mathbf{C}_1[*_i := \mathbf{C}_i]$. By the lemma 2.5.1, $t_i[a :=^* \varepsilon] \rhd_{ehd} t'_i[a :=^* \varepsilon] = \mathbf{C}_i[t^i_1[a :=^* \varepsilon], ..., t^i_p[a :=^* \varepsilon]]$, then $(t\,\varepsilon) = (\mu a.\mathbf{C}_1[t_1, ..., t_i, ..., t_n]\,\varepsilon) \rhd_{ehd}$ $\mu a.\mathbf{C}_1[t_1[a :=^* \varepsilon], ..., t_i[a :=^* \varepsilon], ..., t_n[a :=^* \varepsilon]]$, and $(t'\,\varepsilon) \rhd_{ehd}$ $\mu a.\mathbf{C}'_1[t_1[a :=^* \varepsilon], ..., t^i_1[a :=^* \varepsilon], ..., t^i_p[a :=^* \varepsilon], ..., t_n[a :=^* \varepsilon]]$. Hence $\tau =$ $\mu a.\mathbf{C}'_1[t_1[a :=^* \varepsilon], ..., t^i_1[a :=^* \varepsilon], ..., t^i_p[a :=^* \varepsilon], ..., t_n[a :=^* \varepsilon]]$, and we check easily that $|(t\,\varepsilon), \tau| = |t, t'| + |(t'\,\varepsilon), \tau|$.

- The other cases are similar to the previous.

■

This lemma shows that, to carry out the extra head reduction of $(t\,\bar{w})$ it is equivalent (the same result and the same number of steps) to carry out a certain number of steps of an extra head reduction of t, which gives t', then, to make the extra head reduction of $(t'\,\bar{w})$.

Lemma 2.5.5 *Let t, t' be two terms such that $t \rhd^*_{ehd} t'$, and $\bar{w} = w_1...w_n$ a sequence of $\mathcal{E}$-terms. Then there exists a term τ such that $(t\,\bar{w}) \rhd^*_{ehd} \tau$, $(t'\,\bar{w}) \rhd^*_{ehd} \tau$ and $|(t\,\bar{w}), \tau| = |t, t'| + |(t'\,\bar{w}), \tau|$.*

Proof. By induction on n where $\bar{w} = w_1...w_{n-1}w_n$. It is trivial when $n = 0$ (take $\tau = t'$). Suppose the result true until $(n-1)$, this implies that there exists a term τ_0 such that $(t\ w_1...w_{n-1}) \rhd^*_{ehd} \tau_0$, $(t'\ w_1...w_{n-1}) \rhd^*_{ehd} \tau_0$ and $|(t\ w_1...w_{n-1}), \tau_0| = |t, t'| + |(t'\ w_1...w_{n-1}), \tau_0|$. By the previous lemma, there exist τ_1 and τ_2 such that:

- $(t\,\bar{w}) \rhd^*_{ehd} \tau_1$, $(\tau_0\,w_n) \rhd^*_{ehd} \tau_1$ and $|(t\,\bar{w}), \tau_1| = |(t\,w_1...w_{n-1}), \tau_0| + |(\tau_0\,w_n), \tau_1|$.

- $(t'\,\bar{w}) \rhd^*_{ehd} \tau_2$, $(\tau_0\,w_n) \rhd^*_{ehd} \tau_2$ and $|(t'\,\bar{w}), \tau_2| = |(t'\,w_1...w_{n-1}), \tau_0| + |(\tau_0\,w_n), \tau_2|$.

Since $(\tau_0\ w_n) \rhd^*_{ehd} \tau_1$ and $(\tau_0\ w_n) \rhd^*_{ehd} \tau_2$ and, by the proposition 2.5.1, there exists a term τ_3 such that $\tau_1 \rhd^*_{ehd} \tau_3$, $\tau_2 \rhd^*_{ehd} \tau_3$ and $|\tau_0\ w_n, \tau_1| + |\tau_1, \tau_3| = |\tau_0\ w_n, \tau_2| + |\tau_2, \tau_3|$.

Therefore
$$\begin{aligned}|(t\,\bar{w}), \tau_3| &= |(t\ \bar{w}), \tau_1| + |\tau_1, \tau_3| \\ &= |(t\ w_1...w_{n-1}), \tau_0| + |(\tau_0\ w_n), \tau_1| + |\tau_1, \tau_3| \\ &= |(t\ w_1...w_{n-1}), \tau_0| + |(\tau_0\ w_n), \tau_2| + |\tau_2, \tau_3| \\ &= |t, t'| + |(t'\ w_1...w_{n-1}), \tau_0| + |(\tau_0\ w_n), \tau_2| + |\tau_2, \tau_3| \\ &= |t, t'| + |(t'\ \bar{w}), \tau_2| + |\tau_2, \tau_3| = |t, t'| + |(t'\ \bar{w}), \tau_3|. \end{aligned}$$ ■

Corollary 2.5.3 *Let t, t' be two terms such that $t \rhd^*_{ehd} t'$, and $\bar{w} = w_1...w_n$ a sequence of $\mathcal{E}$-terms. If $t \simeq_{ehd} t'$, then $(t\,\bar{w}) \simeq_{ehd} (t'\,\bar{w})$.*

Proof. By the previous lemma. ■

Definition 2.5.6 *A leftmost reduction of a term t consists to apply a head reduction on t until its head normal form $\tau = \mathbf{C}[\tau_\mathbf{1}, ..., \tau_\mathbf{n}]$ (if it exists) and reiterate it on the arguments of τ_i. We denote $t \rhd^*_l t'$ if t is reduced to t' by a leftmost reduction. When an argument of a head variable is an $\mathcal{E}$-term in the form $[x.u, y.v]$, the reduction consists simply to reduce in u and v.*

The following theorem says that every sequence of leftmost reductions is a standard one.

Theorem 2.5.1 *If $t \rhd^*_l t'$, then this sequence of reductions is a standard one.*

Proof. By induction on $\kappa(t \rhd^*_l t')$.

- The cases where t is not a simple term are a direct consequences of the induction hypothesis:
  If $t = \lambda x.u$, then $t' = \lambda x.u'$, where $u \rhd^*_l u'$. Therefore, by the induction hypothesis (decreasing of the complexity c), $u \rhd^*_l u'$ is a standard sequence reductions, thus $\lambda x.u \rhd^*_l \lambda x.u'$ is a standard one too.

- The cases where t is a simple term:
  If $t = (\mu a.u\,\varepsilon)\,\bar{w}$, then either $t \rhd_{hd} (\mu a.u[a :=^* \varepsilon]\,\bar{w}) \rhd^*_l t'$ or $t \rhd_{hd} ((\mu a.u\,\epsilon)\,\bar{r}) \rhd^*_l t'$ where $\varepsilon\bar{w} \succ \epsilon\bar{r}$. Therefore, by the induction hypothesis (decreasing of the length l), $(\mu a.u[a :=^* \varepsilon]\,\bar{w}) \rhd^*_l t'$ and $((\mu a.u\,\epsilon)\,\bar{r}) \rhd^*_l t'$ are two standard sequence reductions, thus, by definition, both of $t \rhd_{hd} (\mu a.u[a :=^* \varepsilon]\,\bar{w}) \rhd^*_l t'$ and $t \rhd_{hd} ((\mu a.u\,\epsilon)\,\bar{r}) \rhd^*_l t'$ are standard sequence reductions also.

- The other cases are similar to the previous.

■

Remark 2.5.2 *This theorem gives another way to define a leftmost reduction similarly to the definition of the standard reduction, this definition can be easily proven to be "equivalent" to the one given above.*

Definition 2.5.7 *Let ε, ε' be two $\mathcal{E}$-terms, $\bar{w}$ and $\bar{w}'$ two finite sequences of $\mathcal{E}$-terms. We define simultaneously $\varepsilon \triangleright_l^* \varepsilon'$ and $\bar{w} \triangleright_l^* \bar{w}'$ by induction on the ordered lexicographic pair $\kappa(\varepsilon \triangleright_l^* \varepsilon') = (l, c)$ (resp $\kappa(\bar{w} \triangleright_l^* \bar{w}')$) where l stands for the length of the reduction $\varepsilon \triangleright_l^* \varepsilon'$ (resp $(\bar{w} \triangleright_l^* \bar{w}')$) and c the complexity of ε (resp $\bar{w}$).*

1. $\varepsilon \triangleright_l^* \varepsilon'$
   - *If $\varepsilon = \lambda x.u$, then $\varepsilon' = \lambda x.u'$ with $u \triangleright_l^* u'$.*
   - *If $\varepsilon = \mu a.u$, then $\varepsilon' = \mu a.u'$ with $u \triangleright_l^* u'$.*
   - *If $\varepsilon = \langle u, v\rangle$, then $\varepsilon' = \langle u', v'\rangle$ with $u \triangleright_l^* u'$ and $v \triangleright_l^* v'$.*
   - *If $\varepsilon = \omega_i u$, then $\varepsilon' = \omega_i u'$ with $u \triangleright_l^* u'$.*
   - *If $\varepsilon = (a\, u)$, then $\varepsilon' = (a\, u')$ with $u \triangleright_l^* u'$.*
   - *If $\varepsilon = (x\, \bar{w})$, then $\varepsilon' = (x\, \bar{w}')$ with $\bar{w} \triangleright_l^* \bar{w}'$.*
   - *If $\varepsilon = ((\lambda x.u\, v)\, \bar{w})$, then $\varepsilon \triangleright_l (u[x := v]\, \bar{w}) \triangleright_l^* \varepsilon'$.*
   - *If $\varepsilon = ((\mu a.u\ \theta)\, \bar{w})$, then*
     - *Either $\varepsilon \triangleright_l (\mu a.u[a :=^* \theta]\, \bar{w}) \triangleright_l^* \varepsilon'$.*
     - *Or $\varepsilon \triangleright_l (\mu a.u\, \epsilon)\, \bar{s}) \triangleright_l^* \varepsilon'$ with $\theta\bar{w} \succ \epsilon\bar{s}$.*
   - *If $\varepsilon = ((\langle u_1, u_2\rangle\ \pi_i)\, \bar{w})$, then $\varepsilon \triangleright_l (u_i\, \bar{w}) \triangleright_l^* \varepsilon'$,*
   - *If $\varepsilon = ((\omega_i u\ [x_1.v_1, x_2.v_2])\, \bar{w})$, then $\varepsilon \triangleright_l (v_i[x_i := u]\, \bar{w}) \triangleright_l^* \varepsilon'$.*
   - *If $\varepsilon = \pi_i$, then $\varepsilon' = \pi_i$.*
   - *If $\varepsilon = [x_1.u_1, x_2.u_2]$, then $\varepsilon' = [x_1.u_1', x_2.u_2']$ with $u_i \triangleright_l^* u_i'$.*
2. $\bar{w} \triangleright_l^* \bar{w}'$
   - *If $\bar{w}$ is not nice, then $\bar{w} \succ \bar{s} \triangleright_l^* \bar{w}'$.*
   - *Else $\bar{w} = \varepsilon_1 ... \varepsilon_n$, then $\bar{w}' = \varepsilon_1' ... \varepsilon_n'$ with $\varepsilon_i \triangleright_l^* \varepsilon_i$ for each $1 \leq i \leq n$.*

2.6 Finiteness of developments

2.6.1 The marked terms

For the purpose of this section, which is the finiteness developments theorem, we introduce a coloured version of the $\lambda\mu^{\wedge\vee}$-calculus.

Definition 2.6.1 *1. We extend the syntax of the $\lambda\mu^{\wedge\vee}$-calculus by adding coloured λ, μ, $\langle\ ,\ \rangle$, ω_i and $[\ .\ ,\ .\ ]$. The sets $\mathcal{T}$ and $\mathcal{E}$ of coloured (or marked) terms are defined by the following grammars:*

$$\mathcal{T} := \mathcal{T} \mid (\lambda\mathcal{X}.\mathcal{T}\ \mathcal{T}) \mid (\langle\mathcal{T}, \mathcal{T}\rangle\ \pi_i) \mid (\omega_i\mathcal{T}\ [\mathcal{X}.\mathcal{T}, \mathcal{X}.\mathcal{T}]) \mid (\mathcal{T}\ \mathcal{E}) \mid \mu a.\mathcal{T}$$

$$\mathcal{E} := \mathcal{E} \mid \mathcal{T} \mid [\mathcal{X}.\mathcal{T}, \mathcal{X}.\mathcal{T}]$$

2. *The reduction rule* $\blacktriangleright$ *of* $\mathcal{E}$ *consists in the union of the following reduction rules. The meaning of these new reductions is to capture the definition of the finiteness development, where we reduce only redexes at the beginning or only their residus (which will be exactly the marked redexes).*

   - $(\lambda x.t\ u) \blacktriangleright_\beta t[x := u]$
   - $(\langle t_1, t_2\rangle\ \pi_i) \blacktriangleright_{\pi_i} t_i$
   - $(\omega_i t\ [x_1.u_1, x_2.u_2]) \blacktriangleright_D u_i[x_i := t]$
   - $((t\ [x_1.u_1, x_2.u_2])\ \varepsilon) \blacktriangleright_\delta (t\ [x_1.(u_1\ \varepsilon), x_2.(u_2\ \varepsilon)])$
   - $(\mu a.t\ \varepsilon) \blacktriangleright_\mu \mu a.t[a :=^* \varepsilon]$.

   We denote by $\blacktriangleright^*$ *the reflexive and transitive closure of* $\blacktriangleright$.

3. *Let* t *be a term and* r *a redex of* t*, we define* $\hat{r}$ *the coloured redex obtained by colouring* r *as follows:*

   - *If* $r = (\lambda x.u\ v)$*, then* $\hat{r} = (\lambda x.u\ v)$
   - *If* $r = (\langle t_1, t_2\rangle\ \pi_i)$*, then* $\hat{r} = (\langle t_1, t_2\rangle\ \pi_i)$
   - *If* $r = (\omega_i u\ [x_1.u_1, x_2.u_2])$*, then* $\hat{r} = (\omega_i u\ [x_1.u_1, x_2.u_2])$
   - *If* $r = ((u\ [x_1.u_1, x_2.u_2])\ \varepsilon)$*, then* $\hat{r} = ((u\ [x_1.u_1, x_2.u_2])\ \varepsilon)$
   - *If* $r = (\mu a.t\ \varepsilon)$*, then* $\hat{r} = (\mu a.t\ \varepsilon)$

   We denote $Red(t)$ *the set of all the redexes of* t*. Let* R *be a subset of* $Red(t)$*, we define* $\hat{t}_R$ *the marked term obtained from* t *by colouring each redex of* t *which belongs to* R*. We said that* $\hat{t}_R$ *is the corresponding marked term to* t *according to* R.

4. *Let* $\varepsilon \in \mathcal{E}$*, we define* $\check{\varepsilon}$ *by induction on* ε*:*

   - *If* $\varepsilon \in \mathcal{E}$*, then* $\check{\varepsilon} = \varepsilon$
   - *If* $\varepsilon = (\lambda x.u\ v)$*, then* $\check{\varepsilon} = (\lambda x.\check{u}\ \check{v})$
   - *If* $\varepsilon = (\langle t_1, t_2\rangle\ \pi_i)$*, then* $\check{\varepsilon} = (\langle \check{t}_1, \check{t}_2\rangle\ \pi_i)$
   - *If* $\varepsilon = (\omega_i t\ [x.u, y.v])$*, then* $\check{\varepsilon} = (\omega_i \check{t}\ [x.\check{u}, y.\check{v}])$
   - *If* $\varepsilon = (t\ \epsilon)$*, then* $\check{\varepsilon} = (\check{t}\ \check{\epsilon})$
   - *If* $\varepsilon = \mu a.u$*, then* $\check{\varepsilon} = \mu a.\check{u}$
   - *If* $\varepsilon = [x.u, y.v]$*, then* $\check{\varepsilon} = [x.\check{u}, y.\check{v}]$

The operation "$\check{}$" quite simply consists in projecting any marked term in the set $\mathcal{T}$, i.e. to consider any coloured term as any other term without colors.

Example 2.6.1 *Let $t = ((\mu a.u\,[x.p, y.q])\,\varepsilon)$, let also R, S, T, K and L be subsets of $Red(t)$ such that: $R = \{(\mu a.u\,[x.p, y.q])\}$, $S = \{t\}$, $K \subset Red(p)$, $L \subset (Red(u) \cup Red(\varepsilon))$ and $T = \{(\mu a.u\,[x.p, y.q]), t\}$ then,*

1. $\hat{t}_R = ((\mu a.u\,[x.p, y.q])\,\varepsilon)$
2. $\hat{t}_S = ((\mu a.u\,[x.p, y.q])\,\varepsilon)$
3. $\hat{t}_K = ((\mu a.u\,[x.\hat{p}_K, y.q])\,\varepsilon)$
4. $\hat{t}_L = ((\mu a.\hat{u}_L\,[x.p, y.q])\,\hat{\varepsilon}_L)$
5. $\hat{t}_T = ((\mu a.u\,[x.p, y.q])\,\varepsilon)$

Remark 2.6.1

1. *Let t be a term and $R \subseteq Red(t)$ it is clear that $\check{\hat{t}}_R = t$.*
2. *When there is only one given set R of redexes of a given term t, we use the abusive notation $\hat{t}$ to denote the corresponding marked term to t according to R.*
3. *Observe that "terms" in the forms: $\lambda x.t$, $\langle t_1, t_2\rangle$ and $\omega_i t$ are not elements of $\mathcal{E}$. Colours can only occur as colours of redexes except in terms of the form $\mu a.t$ and the form $(t\,[x.u, y.v])$. It follows that a $\blacktriangleright$-normal form never contains any colours except μ or $[.\,,\,.]$.*
4. *It is also clear that any term is a $\blacktriangleright$-normal marked term (since $\blacktriangleright$ consists only in reducing marked redexes).*

The following lemma shows that the set of marked terms is closed under β and μ-substitutions.

Lemma 2.6.1 *Let t, u and θ be $\mathcal{E}$-terms, then, $t[x := u]$ and $t[a :=^* \theta]$ are $\mathcal{E}$-terms.*

Proof. By induction on t. ■

The next lemma shows that the set $\mathcal{E}$ is closed under the $\blacktriangleright$ reduction.

Lemma 2.6.2 *If $t \in \mathcal{E}$ and $t \blacktriangleright^* t'$, then $t' \in \mathcal{E}$.*

Proof. By induction on the marked term t using the lemma 2.6.1. ■

Lemma 2.6.3 *Let t, ε and θ be $\mathcal{E}$-terms such that: $t \blacktriangleright t'$, $\varepsilon \blacktriangleright \varepsilon'$ and $\theta \blacktriangleright^* \theta'$, then $\varepsilon[x := t] \blacktriangleright^* \varepsilon[x := t']$ and $\varepsilon[a :=^* \theta] \blacktriangleright^* \varepsilon[a :=^* \theta']$.*

Proof. Easy. ■

2.6.2 Finiteness developments theorem

Definition 2.6.2 *Let t be a term and R a subset of $Red(t)$.*

1. *A sequence of reductions $\hat{t} = t_0 \blacktriangleright t_1 \blacktriangleright t_2 \blacktriangleright ...$ is called R-development of t (it is clear that the reduced redexes are only marked redexes). It is denoted by $\hat{t} \blacktriangleright^* t'$ if it finishes with the marked term t'. We denote it also by $t \rhd_R^* \check{t'}$ $(t = \check{t_0} \rhd_R \check{t_1} \rhd_R \check{t_2} ... \rhd_R \check{t'})$.*

2. *Let t be a term, t is said to be R-strongly normalizable iff there are no infinite R-developments of t, i.e. all the R-developments are finites.*

Remark 2.6.2 *If t is a term and R a set of some redexes of t, then note that $\rhd_R$ and $\blacktriangleright$ reduction are the same, any infinite sequence of $\blacktriangleright$ reductions starting from $\hat{t}$ corresponds to an infinite sequence of $\rhd_R^*$ reductions starting from t. Thus one can be able to identify them, this fact will be implicitly used in the next paragraph, where $\rhd_R$ and $\blacktriangleright$ will be confused.*

Lemma 2.6.4 *Let t and t' be terms, ε, ε', θ and θ' $\mathcal{E}$-terms. Let also R be a set of redexes of t, ε and θ such that: $\varepsilon \rhd_R^* \varepsilon'$.*

1. *If $t \rhd_R^* t'$, then, $\varepsilon[x := t] \rhd_R^* \varepsilon'[x := t']$.*

2. *If $\theta \rhd_R^* \theta'$, then, $\varepsilon[a :=^* \theta] \rhd_R^* \varepsilon'[a :=^* \theta']$.*

Proof. By induction on ε. ■

Definition 2.6.3 *Let $\bar{w} = \varepsilon_1 ... \varepsilon_m$ be a finite sequence of $\mathcal{E}$-terms.*

1. *The length of $\bar{w}$ is defined as follows: $lg(\bar{w}) = m$.*

2. *A semi-permutative redex of $\bar{w}$ is any initial segment in the form $(\bar{r}[x.p, y.q])\varepsilon_i$, where $(2 \leq i \leq m)$ and $\bar{r}$ is possibly empty.*

3. *A set of redexes of $\bar{w}$ is the union of sets of redexes of each ε_i and any possible semi-permutative redex of $\bar{w}$.*

Lemma 2.6.5 *Let t be a term, $\bar{u} = u_1 ... u_n$ a finite sequence of terms and $\bar{\varepsilon} = \varepsilon_1 ... \varepsilon_m$ a finite sequence of $\mathcal{E}$-terms. Let also $\sigma = [(x_i := u_i)_{1 \leq i \leq n}; (a_j :=^* \varepsilon_j)_{1 \leq j \leq m}]$ and R be a set of redexes of t, $\bar{u}$ and $\bar{\varepsilon}$. If t, $\bar{u}$ and $\bar{\varepsilon}$ are R-strongly normalizables, then, $t\sigma$ is R-strongly normalizable.*

Proof. A similar proof to the one of the next theorem. ■

Theorem 2.6.1 *Let t be a term and $R \subseteq Red(t)$, then t is R-strongly normalizable.*

Proof. First, let $t = (u\,\bar{w})$ as in the corollary 2.3.1. We prove this by induction on the ordered lexicographic pair $(lg(\bar{w}), c)$ where c denotes the complexity of t (for the simplicity and the clearness of the proof, we prefer to do not use coloured terms, we think that now everyone understand the idea behind marking terms).

– The cases where t is not simple are direct consequences of induction hypthesis (decreasing of c).

– The case $t = (x\,\bar{w})$

- If $\bar{w} = w_1...w_m$ is nice, then by the induction hypothesis, each w_i is R-strongly normalizable. Therefore t too.
- Else $t = (x\,w_1...w_{j-2})[y.p, z.q]w_j...w_m$. Suppose that there exists a sequence of infinite R-reductions starting from t, then by induction hypothesis, this sequence does not start from any w_i. Therefore $t = (x\,w_1...w_{j-2})[y.p, z.q]w_j...w_m \triangleright_R^* (x\,w'_1...w'_{j-2})[y.p', z.q']w'_j...w'_m \triangleright_R (x\,w'_1...w'_{j-2})[y.(p'\,w'_j), z.(q'\,w'_j)]...w'_m = t'_j \triangleright_R^* ....$ By the standardization theorem $t \triangleright_{st}^* t'_j$, this standard reduction is in the form $t = (x\,w_1...w_{j-2})[y.p, z.q]w_j...w_m \triangleright_R (x\,w_1...w_{j-2})[y.(p\,w_j), z.(q\,w_j)]...w_m \triangleright_R^* (x\,w'_1...w'_{j-2})[y.(p'\,w'_j), z.(q'\,w'_j)]...w'_m$. This means that $(x\,w_1...w_{j-2})[y.(p\,w_j), z.(q\,w_j)]...w_m \triangleright_R^* ...$, since we can not reduce the possible created redexes $(p\,w_j)$ or $(q\,w_j)$, this gives a contradiction with the induction hypothesis (decreasing of $lg(\bar{w})$, from m to $m-1$).

– The case $t = (\mu a.u\,\varepsilon)\bar{w}$, this gives by the induction hypothesis and the standardization theorem two possibilities to the form of the sequence of the R-infinite reductions:

- $t = (\mu a.u\,\varepsilon)\,w_1...w_m \triangleright_R (\mu a.u[a :=^* \varepsilon]\,w_1)\,w_2...w_m \triangleright_R^* (\mu a.u'[a :=^* \varepsilon']\,w'_1)\,w'_2...w'_m \triangleright_R^* ....$ This means that $(\mu a.u[a :=^* \varepsilon]\,w_1)\,w_2...w_m \triangleright_R^*$. By the lemma 2.6.5 $u[a :=^* \varepsilon]$ is R-strongly normalizable, therefore this gives a condradiction with the induction hypothesis (decreasing of $lg(\bar{w})$, from m to $m-1$).
- $t = (\mu a.u\,\varepsilon)\,w_1...[x.p, y.q]w_j...w_m \triangleright_R (\mu a.u\,\varepsilon)\,w_1...[x.(p\,w_j), y.(q\,w_j)]...w_m \triangleright_R^* (\mu a.u'\,\varepsilon')\,w'_1...[x.(p'\,w'_j), y.(q'\,w'_j)]...w'_m \triangleright_R^* ....$ This means that $(\mu a.u\,\varepsilon)\,w_1...[x.(p\,w_j), y.(q\,w_j)]...w_m \triangleright_R^* ....$ By a similar argument as the previous this gives a contradiction.

■

Theorem 2.6.2 (Local confluence of $\triangleright_R$) *Let t, t_1, t_2 be terms and $R \subseteq Red(t)$, such that $t \triangleright_R t_1$ and $t \triangleright_R t_2$, then there exists t_3 such that $t_1 \triangleright_R^* t_3$ and $t_2 \triangleright_R^* t_3$.*

Proof. By induction on the complexity of t. Let us check the following cases, the other cases are simple consequences of the induction hypothesis. In this proof we only marke the redexes which will be reduced.

- $t = (x\,w_1)w_2...[r.u, s.v][y.p, z.q]w_j...w_m$,
  $t_1 = (x\,w_1)w_2...[r.(u\,[y.p, z.q]), s.(v\,[y.p, z.q])]w_j...w_m$
  and $t_2 = (x\,w_1)w_2...[r.u, s.v][y.(p\,w_j), z.(q\,w_j)]...w_m$. Therefore check that
  $t_3 = (x\,w_1)w_2...[r.(u\,[y.(p\,w_j), z.(q\,w_j)]), s.(v\,[y.(p\,w_j), z.(q\,w_j)])]...w_m$
  is the commun marked redectum obtained of course by R-reductions, thus $\check{t_3}$ concludes.

- $t = ((\mu a.u\ [x.r, y.s])\ \varepsilon\bar{w})$.

  - If $t_1 = ((\mu a.u'\ [x.r, y.s])\ \varepsilon\bar{w})$ and $t_2 = (\mu a.u\ [a :=^* [x.r, y.s]]\ \varepsilon\bar{w})$, then, by the lemma 2.6.3, we take $t_3 = (\mu a.u'\ [a :=^* [x.r, y.s]]\ \varepsilon\bar{w})$.
  - If $t_1 = ((\mu a.u'\ [x.r, y.s])\ \varepsilon\bar{w})$ and $t_2 = ((\mu a.u\,[x.(r\ \varepsilon), y.(s\ \varepsilon)])\,\bar{w})$, then we take $t_3 = ((\mu a.u'\,[x.(r\ \varepsilon), y.(s\ \varepsilon)])\,\bar{w})$.
  - If $t_1 = (\mu a.u[a :=^* [x.r, y.s]]\ \varepsilon\bar{w})$ and $t_2 = ((\mu a.u\,[x.(r\ \varepsilon), y.(s\ \varepsilon)])\,\bar{w})$, then, both of the commutative and the classical redexes are marked redexes in $\hat{t} = ((\mu a.u\,[x.r, y.s])\ \varepsilon\bar{w})$, hence $\hat{t_1} = (\mu a.u[a :=^* [x.r, y.s]]\ \varepsilon\bar{w})$ $\triangleright_R (\mu a.u[a :=^* [x.r, y.s]\ \varepsilon]\ \bar{w})\ \triangleright_R\ (\mu a.u[a :=^* [x.(r\ \varepsilon), y.(s\ \varepsilon)]]\bar{w}) = t_3$, and $t_2 = ((\mu a.u\,[x.(r\ \varepsilon), y.(s\ \varepsilon)])\,\bar{w}) \triangleright_R (\mu a.u[a :=^* [x.(r\ \varepsilon), y.(s\ \varepsilon)]]\ \bar{w})$ $= t_3$.
    It is obvious that $\check{t_3} = (\mu a.u[a :=^* [x.(r\ \varepsilon), y.(s\ \varepsilon)]]\bar{w})$ is the commun redectum.

■

Theorem 2.6.3 (Confluence of $\triangleright_R$) *Let t, t_1, t_2 be terms and $R \subseteq Red(t)$, such that $t \triangleright_R^* t_1$ and $t \triangleright_R^* t_2$, then there exists t_3 such that $t_1 \triangleright_R^* t_3$ and $t_1 \triangleright_R^* t_3$.*

Proof. This is now a direct consequence of the theorem 2.6.2 and Newman lemma. ■

Lemma 2.6.6 *Let $\mathcal{R}$ be the reduction relation defined as follows: $t\,\mathcal{R}\,t'$ iff $t \triangleright_R t'$, where $R = Red(t)$. Then $\triangleright^*$ is the reflexive and transitive closure of $\mathcal{R}$.*

Proof. Let us denote by $\mathcal{R}^*$ the reflexive and transitive closure of $\mathcal{R}$. If $t\,\mathcal{R}\,t'$, then $t \triangleright^* t'$ and $\mathcal{R}^* \subseteq \triangleright^*$. If $t \triangleright t'$, then $t\,\mathcal{R}\,t'$ and $\triangleright^* \subseteq \mathcal{R}^*$. Therefore $\triangleright^* = \mathcal{R}^*$. ■

We are now in position to prove the main result of this chapter.

Theorem 2.6.4 (Confluence of $\triangleright$) *Let t, t_1, t_2 be terms such that $t \triangleright^* t_1$ and $t \triangleright^* t_2$, then there exists t_3 such that $t_1 \triangleright^* t_3$ and $t_1 \triangleright^* t_3$*

Proof. Directly from the previous lemma and theorem 2.6.3. ■

One of the consequences of the standardization theorem is that the normal form if it exists can be reached by the leftmost reduction.

Theorem 2.6.5 *If t' is the normal form of t, then $t \triangleright_l^* t'$.*

Proof. Since $t \triangleright^* t'$, then there exists a standard reduction from t to t', i.e. $t \triangleright_{st}^* t'$. We process by induction on $\kappa(t \triangleright_{st}^* t')$. The cases where t is not a simple term are direct consequences of the induction hypothesis (decreasing of c). Let us examine the two following cases:

– $t = (x\,\bar{w})$

- If $\bar{w} = w_1 ... w_n$ is nice, then $t' = (x\,\bar{w}')$ where $\bar{w}' = w_1' ... w_n'$ and each $w_i \triangleright^* w_i'$ normal. Therefore the induction hypothesis concludes.
- Else $t = ((x\,\bar{R}[y.p.z.q])w_i\bar{K})$ and the standard reduction is in the form: $t = ((x\,\bar{R})[y.p.z.q]w_i\bar{K}) \triangleright ((x\,\bar{R})[y.(p\,w_i).z.(q\,w_i)])\bar{K}) \triangleright_{st}^* t'$. Therefore by the induction hypothesis (decreasing of l), we have that $((x\,\bar{R})[y.(p\,w_i).z.(q\,w_i)])\bar{K}) \triangleright_l^* t'$. Thus $t = ((x\,\bar{R})[y.p.z.q]w_i\bar{K}) \triangleright_l ((x\,\bar{R})[y.(p\,w_i).z.(q\,w_i)])\bar{K}) \triangleright_l^* t'$.

- $t = ((\mu a.u\,\varepsilon)\,\bar{w})$, then the standard reduction from t to the normal form t' is in the following forms: Either $t = ((\mu a.u\,\varepsilon)\,\bar{w}) \triangleright ((\mu a.u\,\epsilon)\,\bar{r}) \triangleright_{st}^* t'$ or $t = ((\mu a.u\,\varepsilon)\,\bar{w}) \triangleright (\mu a.u[a :=^* \varepsilon]\,\bar{w}) \triangleright_{st}^* t'$. Therefore by the induction hypothesis (decreasing of l) we have that $((\mu a.u\,\epsilon)\,\bar{r}) \triangleright_l^* t'$ and $(\mu a.u[a :=^* \varepsilon]\,\bar{w}) \triangleright_l^* t'$, hence $t = ((\mu a.u\,\varepsilon)\,\bar{w}) \triangleright ((\mu a.u\,\epsilon)\,\bar{r}) \triangleright_l^* t'$ and $((\mu a.u\,\varepsilon)\,\bar{w}) \triangleright_l (\mu a.u[a :=^* \varepsilon]\,\bar{w}) \triangleright_l^* t'$.

■

2.7 Krivine machine

The Krivine machine (KAM) (see [15]), is a simple and natural implementation of the normal weak-head-reduction strategy for pures λ-terms. O. Laurent in [17], gave an extension of the KAM to the $\lambda\mu$-calculus (see also [5] and [6]).

Lemma 2.7.1 (*and definition*)

1. *A term t either has an extra weak head redex which is its head redex* (*or its outer most redex*) (*if t is simple*), *or is in an extra weak head normal form which is one of the following forms: A simple extra head normal form, $\lambda x.u$, $\mu a.u$, $\langle u, v\rangle$ or $\omega_i u$.*

2. *The extra weak head reduction of a term t consists in reducing its extra weak head redex, it is denoted by $\triangleright_w$. It is clear that any term which is not simple is in extra weak head normal form.*

Proof. Similar to the one of the lemma 2.3.3. ■

Definition 2.7.1 *We consider three areas in the memory: the environment, the stack and the terms area reserved to the terms which will be performed.*

1. *An environment E is a partial function with a finite domain* (*denoted $Dom(E)$*) *of λ and μ-variables such that:*

   - *If $x \in Dom(E)$, $E(x)$ is a closure.*
   - *If $a \in Dom(E)$, $E(a)$ is a stack.*

2. *A closure c is an ordered pair (ε, E), where ε is an $\mathcal{E}$-term and E is an environment. A nice closure is a closure (ε, E) such that ε is not in the form $[x.u, y.v]$.*

3. *A stack Π is a finite sequence of closures, sometimes it can be denoted also by $c_1 :: c_2 :: ... :: c_n$. A stack Π is called a nice stack iff all its closures are nice except the last* (*which can be nice or not nice*).

4. *A state is a triple (ε, E, Π), where ε is an $\mathcal{E}$-term, E an environment and Π a stack.*

Remark 2.7.1 *The intuition is that an environment defines the value of some variables, a closure contains a term and the definitions necessary to its free variables* (*which are in the environment*).

Definition 2.7.2 1. *Ξ (resp Φ) denotes the empty environment* (*resp the empty stack*).

2. *The stack $c :: \Pi$ is the stack obtained by pushing the closure c on the top of the stack Π.*

3. *Two environments are said to be compatibles if they do not associate two differents values to the same variable.*

4. *We denote $E' \subseteq E$, if E and E' are two compatibles environments such that $Dom(E') \subseteq Dom(E)$.*

5. *Let $\Pi = (\epsilon_1, E_1) :: (\epsilon_2\, E_2) :: ... :: (\epsilon_n, E_n)$, then, the state (ε, E, Π) is said to be good iff :*

   - $Fv(\varepsilon) \subseteq E$,
   - $E_n \subseteq ... \subseteq E_1 \subseteq E$.

6. *The transition rules of the machine describe how to pass from a state (ε, E, Π) to an other one (ε', E', Π').*

Push:

Terms	*Environments*	*Stacks*
$(t\,\varepsilon)$	E	Π
t	E	$(\varepsilon, E) :: \Pi$

Pop:

Terms	*Environments*	*Stacks*
$\lambda x.t$	E	$(v, E') :: \Pi$
t	$E + (x = (v, E))$	Π

Pop':

Terms	*Environments*	*Stacks*
$\langle t_1, t_2 \rangle$	E	$(\pi_i, E') :: \Pi$
t_i	E	Π

Pop":

Terms	*Environments*	*Stacks*
$\omega_i t$	E	$([x_1.u_1, x_2.u_2], E') :: \Pi$
u_i	$E + (x_i = (t, E))$	Π

Save:

Terms	*Environments*	*Stacks*
$\mu a.t$	E	Π
t	$E + (a = \Pi)$	Φ

Restore:

Terms	*Environments*	*Stacks*
$(a\,t)$	E	Φ
t	E	Π

where $E(a) = \Pi$.

Deref: *If Π is nice,*

Terms	*Environments*	*Stacks*
x	E	Π
t	E	Π

where $E(x) = (t, E')$

If no rule from the seven rules given above makes it possible to pass then, either one uses the following rule:

Save up: *If* Π' *nice,*

Terms	*Environments*	*Stacks*
x	E	$\Pi :: ([y.v, z.w], E') :: \Pi'$
x	E	$\Pi :: ([y.(v\,\bar{w}), z.(w\,\bar{w})], E')$

where $\Pi' = (\epsilon_1, E_1) :: (\epsilon_2\, E_2) :: ... :: (\epsilon_n, E_n)$ *and* $\bar{w}$ *is the nice sequence* $\epsilon_1\, \epsilon_2\, ...\epsilon_n$.
Or,

Stop: *The machine stops.*

Lemma 2.7.2 *For any transition of the machine from a good state* (t, E, Π) *to a state* (t', E', Π') *we have:*

1. $E \subseteq E'$.
2. *If* $\Pi' = (\epsilon'_1, E'_1) :: (\epsilon'_2\, E'_2) :: ... :: (\epsilon'_n, E'_n)$, *then* $E'_n \subseteq E'_{n-1} \subseteq ... \subseteq E'_2 \subseteq E'_1 \subseteq E'$.
3. (t', E', Π') *is a good state.*

Proof. (1) and (2): By a simultaneous induction. We prove these properties for each transition rule of the machine.
(3): We check that is a direct consequence of (1) and (2). ■

Definition 2.7.3 (*and notations*)

1. *Let* x *be a* λ*-variable and* E *an environment, such that* $E(x) = (v, E')$, *hence we denote* v *by* $E^{\lambda}(x)$ *and* E' *by* $E^{e}(x)$.
2. *Let* a *be a* μ*-variable and* E *an environment, such that* $E(a) = \Pi$, *hence we denote* Π *by* $E^{\mu}(a)$.
3. *Let* (ε, E) *be a closure and* $\Pi = (\epsilon_1, E_1) :: ... :: (\epsilon_n, E_n)$ *a stack. We define, by simultaneous induction, the* $\mathcal{E}$*-term* $\varepsilon\{E\}$ *and the sequence of* $\mathcal{E}$*-terms* $\widetilde{\Pi}$.
   - $\varepsilon\{E\} = \varepsilon[(x := E^{\lambda}(x)\{E^{e}(x)\})_{x \in Dom(E)}; (a :=^{*} \widetilde{E^{\mu}(a)})_{a \in Dom(E)}]$.
   - $\widetilde{\Pi}$ *is the sequence of* $\mathcal{E}$*-terms* $\epsilon_1\{E_1\}...\epsilon_n\{E_n\}$.

Lemma 2.7.3 *Let* (t, E, Π) *and* (t', E', Π') *be two consecutive states of the machine, then* $(t\{E\}\,\widetilde{\Pi})$ *is reduced by the extra weak head reduction to a term* t_w *such that* $(t'\{E'\}\,\widetilde{\Pi'})$ *is obtained from* t_w *by erasing the* μa_i *on the head of* t_w *and the occurences of the* μ*-variables* a_i *bounded by these* μa_i.

Proof. We prove this for each transition rule of the machine.

Push: Then $(t\ \varepsilon)\{E\}\ \widetilde{\Pi} = (t\{E\}\ \varepsilon\{E\})\ \widetilde{\Pi} = (t\{E\}\ \widetilde{(\varepsilon, E)} :: \Pi) = (t'\{E'\}\ \widetilde{\Pi'})$.

Pop: Then $((\lambda x.t)\{E\}\ \widetilde{(u, E_1)} :: \Pi) = (\lambda x.t\{E\}\ u\{E_1\})\ \widetilde{\Pi} \triangleright_w (t\{E\}[x := u\{E_1\}]\ \widetilde{\Pi})$ $= (t\{E + x = (u, E_1)\}\ \widetilde{\Pi}) = (t'\{E'\}\ \widetilde{\Pi'})$.

Pop': Similar to the rule **Pop**.

Pop": Similar to the rule **Pop**.

Save: Then $((\mu a.t)\{E\}\ \widetilde{\Pi}) = (\mu a.t\{E\}\ \widetilde{\Pi}) \triangleright_w^*\ \mu a.t\{E\}[a :=^*\ \widetilde{\Pi}] = (\mu a.t\{E + a = \widetilde{\Pi}\}$ = which gives $t\{E + a = \widetilde{\Pi}\} = (t'\{E'\}\ \widetilde{\Pi'})$ when erasing μa and the correspondings a.

Restore: Then $(a\,t)\{E\} = (t\{E\}\ \widetilde{\Pi'}) = (t'\{E'\}\ \widetilde{\Pi'})$, in this case $\widetilde{\Pi}$ is empty and $E(a) = \widetilde{\Pi'}$.

Save up: In this case $\Pi = \Pi_1 :: ([y.v, z.w], E_1) :: \Pi_2$, where Π_2 is nice. Then $(x\{E\}\ \widetilde{\Pi}) \triangleright_w^*\ (x\{E\}\ \widetilde{\Pi_1}[y.(v\{E_1\}\ \widetilde{\Pi_2}), z.(w\{E_1\}\ \widetilde{\Pi_2})])$
$= (x\{E\}\ \widetilde{\Pi_1}[y.(v\,\bar{w})\{E_1\}, z.(w\,\bar{w})\{E_1\}]) = (x\{E\}\ \widetilde{\Pi_1}(\widetilde{[y.(v\,\bar{w}), z.(w\,\bar{w})]}, E_1)$
$= (x\{E'\}\ \widetilde{\Pi'})$, $E = E'$, $\Pi_1([y.(v\,\bar{w}), z.(w\,\bar{w})], E_1) = \Pi'$, $\Pi_2 = (\epsilon_1, E^1) :: ... :: (\epsilon_n, E^n)$ and the nice sequence $\bar{w} = \epsilon_1 ... \epsilon_n$.

■

Corollary 2.7.1 (The extra weak head normal form) *Let t be a closed term, if the machine from the state (t, Ξ, Φ) stops at the state (t', E', Π') and t_w is the extra weak head normal form of t, then $(t'\{E'\}\ \widetilde{\Pi'})$ is obtained from t_w by erasing the μa_i on the head of t_w and the occurences of the μ-variables a_i bounded by these μa_i.*

Proof. Immediatly from the previous lemma. Observe that (t, Ξ, Φ) is a good state since t is a closed term. ■

Bibliography

[1] Y. Andou. *A normalization-procedure for the first order classical natural deduction with full logical symbols.* Tsukuba J. Math, vol 19, pp. 153-162, 1995.

[2] Y. Andou. *Church-Rosser property of simple reduction for full first-order classical natural deduction.* Annals of Pure and Applied Logic, vol 119, pp. 225-237, 2003.

[3] H. P. Barendregt. *The Lambda Calculus, its syntax and semantics.* North Holland 1984.

[4] P. Crégut. *Machines à environnement pour la réduction symbolique et l'évaluation partielle.* Phd Thesis, Paris 7 University, 1991.

[5] T. Crolard. *Extension de l'hisomorphisme de Curry-Howard au traitment des exceptions* (*application d'une étude de la dualité en logique intuitionniste*). Thèse de Doctorat, Université Paris 7, 1996.

[6] T. Crolard. *A confluent Lambda-calculus with a* `catch/throw` *mechanism.* Journal of Functional Programming, 9(6): 625-647, 1999.

[7] R. David and K. Nour. *A short proof of the Strong Normalization of Classical Natural Deduction with Disjunction.* Journal of Symbolic Logic, vol 68, num 4, pp. 1277-1288, 2003.

[8] R. David. *Une preuve simple de résultats classiques en λ-calcul.* C.R. Acad. Sci. Paris, t. 320, Série I, pp. 1401-1406, 1995

[9] P. De Groote. *An environment machine for the lambda-mu-calculus.* Mathematical Structures in Computer Science, 8(6), pp. 637-669, 1998.

[10] P. De Groote. *On the Strong Normalization of Natural Deduction with Permutation-Conversions.* In 10th International Conference on Rewriting Techniques and Application, RTA'99, volume 1631 of Lecture Notes in Computer Science, pp. 45-59. Springer Verlag, 1999.

[11] P. De Groote. *Strong Normalization of Classical Natural Deduction with Disjunction.* In 5th International Conference on typed lambda calculi and applications, TLCA'01. LNCS (2044), pp. 182-196. Springer Verlag, 2001.

[12] G. Gentzen. *Recherches sur la déduction logique.* Press Universitaires de France, 1955. Traduction et commentaires par R. Feys et J. Ladrière.

[13] F. Joachimski and R. Matthes. *Standardization and Confluence for a Lambda Calculus with a Generalized Applications.* Rewriting Techniques and Applications, 11th International Conference, RTA 2000, Norwich, UK, July 10-12, pp. 141-155, 2000.

[14] J.-L. Krivine. *Lambda calcul, types et modèle.* Masson, Paris, 1990.

[15] J.-L. Krivine. *Un interpréteur du λ-calcul.* Unpublished draft. Available at `http://www.pps.jussieu.fr/krivine/`.

[16] F. Lang, Z. Benaissa and P. Lescanne. *Super-Closures.* In Proc. of WPAM'98, as Technical Report of the University of SaarBruck, number A 02/98, 1998.

[17] O. Laurent. *Interprétation calculatoire de la logique classique: $\lambda\mu$-calcul et machine de Krivine.* Available at `http://www.pps.jussieu.fr/laurent/`.

[18] R. Matthes. *Non-strictly positive fixed point for classical natural deduction.* Annals of Pure and Applied Logic 133, pp.205-230, 2005.

[19] K. Nour and K. Saber. *A semantical proof of strong normalization theorem for full propositional classical natural deduction.* Archive for Mathematical Logic, vol 45, pp. 357-364, 2005.

[20] K. Nour and K. Saber. *Confluency property of the call-by-value $\lambda\mu^{\wedge\vee}$-calculus.* Computational Logic and Applications CLA'05. Discrete Mathematics and Theoretical Computer Science proc, pp. 97-108 ,2006.

[21] M. Parigot *$\lambda\mu$-calculus: An algorithm interpretation of classical natural deduction.* Lecture Notes in Artificial Intelligence, vol 624, pp. 190-201. Springer Verlag, 1992.

[22] W. Py. *Confluence en $\lambda\mu$-calcul.* PhD thesis, University of Chambéry, 1998.

[23] P. Sestoft. *Deriving a lazy abstract machine.* Journal of functional programming, 7(3), pp. 231- 264, 1997.

[24] Terese. *Term Rewriting Systems.* Cambridge Tracts in Theoritical Computer Science 55, CAMBRIDGE Univesity Press, 2003.

Chapter 3

The strong normalization

3.1 Introduction

This chapter gives a semantical proof of the strong normalization of the cut-elimination procedure for full propositional classical logic written in natural deduction style. By full we mean that all the logical connectives ($\bot$, $\rightarrow$, $\wedge$ and $\vee$) are considered as primitive. We also consider the three reduction relations (logical, commutative and classical reductions) necessary to obtain the subformula property (see [4]).

Until very recently (see the introduction of [4] for a brief history), no proof of the strong normalization of the cut-elimination procedure was known for full logic. In [4], Ph. De Groote gives such a proof by using a CPS-style transformation from full classical logic to implicative intuitionistic logic, i.e. the simply typed λ-calculus. However its proof is not finished yet, since the modified CPS-transformations do not preserve always the stricteness of reductions, and this for the same reasons pointed out in [8] and [9].

A very elegant and direct proof of the strong normalization of the full logic is given in [6] but only the intuitionistic case is given.

R. David and K. Nour give in [3] a direct and syntactical proof of this result. This proof is based on a characterization of the strongly normalizable deductions and a substitution lemma which stipulates the fact that, the deduction obtained while replacing in a strongly normalizable deduction an hypothesis by another strongly normalizable deduction is also strongly normalizable. The same idea is used in [3] to give a short proof of the strong normalization of the simply typed $\lambda\mu$-calculus of [12].

R. Matthes recently found another semantical proof of this result (see [7]). His proof uses a complicated concept of saturated subsets of terms.

Our proof is a generalization of M. Parigot's strong normalization result of the $\lambda\mu$-calculus (see [13]) for the types of J.-Y. Girard's system $\mathcal{F}$ using reducibility candidates. We also use a very technical lemma proved in [3] concerning com-

mutative reductions. To the best of our knowledge, this is the shortest proof of a such result.

This chapter is organized as follows. In section 2, we give the syntax of the terms and the reduction rules. In section 3, we define the reducibility candidates and establish some important properties. In section 4, we show an "adequation lemma" which allows to prove the strong normalization of all typed terms.

3.2 The typed system

We use notations inspired by the paper [1].

Definition 3.2.1 *1. The types are built from propositional variables and the constant symbol $\bot$ with the connectors $\to$, $\wedge$ and $\vee$.*

2. Let $\mathcal{X}$ and $\mathcal{A}$ be two disjoint infinite alphabets for distinguishing the λ-variables and μ-variables respectively. We code deductions by using a set of terms $\mathcal{T}$ which extends the λ-terms and is given by the following grammars:

$$\mathcal{T} := \mathcal{X} \mid \lambda\mathcal{X}.\mathcal{T} \mid (\mathcal{T}\ \ \mathcal{E}) \mid \langle\mathcal{T},\mathcal{T}\rangle \mid \omega_1\mathcal{T} \mid \omega_2\mathcal{T} \mid \mu\mathcal{A}.\mathcal{T} \mid (\mathcal{A}\ \ \mathcal{T})$$
$$\mathcal{E} := \mathcal{T} \mid \pi_1 \mid\ \ \pi_2 \mid [\mathcal{X}.\mathcal{T},\mathcal{X}.\mathcal{T}]$$

An element of the set $\mathcal{E}$ is said to be an $\mathcal{E}$-term.

3. The meaning of the new constructors is given by the typing rules below where Γ (resp. Δ) is a context, i.e. a set of declarations of the form $x : A$ (resp. $a : A$) where x is a λ-variable (resp. a is a μ-variable) and A is a type.

$$\frac{}{\Gamma, x : A\ \vdash x : A\ ;\ \Delta}ax$$

$$\frac{\Gamma, x : A \vdash t : B; \Delta}{\Gamma \vdash \lambda x.t : A \to B; \Delta}\to_i \qquad \frac{\Gamma \vdash u : A \to B; \Delta \quad \Gamma \vdash v : A; \Delta}{\Gamma \vdash (u\ \ v) : B; \Delta}\to_e$$

$$\frac{\Gamma \vdash u : A; \Delta \quad \Gamma \vdash v : B; \Delta}{\Gamma \vdash \langle u, v\rangle : A \wedge B; \Delta}\wedge_i$$

$$\frac{\Gamma \vdash t : A \wedge B; \Delta}{\Gamma \vdash (t\ \ \pi_1) : A; \Delta}\wedge_e^1 \quad \frac{\Gamma \vdash t : A \wedge B; \Delta}{\Gamma \vdash (t\ \ \pi_2) : B; \Delta}\wedge_e^2$$

$$\frac{\Gamma \vdash t : A; \Delta}{\Gamma \vdash \omega_1 t : A \vee B; \Delta}\vee_i^1 \quad \frac{\Gamma \vdash t : B; \Delta}{\Gamma \vdash \omega_2 t : A \vee B; \Delta}\vee_i^2$$

$$\frac{\Gamma \vdash t : A \vee B; \Delta \quad \Gamma, x : A \vdash u : C; \Delta \quad \Gamma, y : B \vdash v : C; \Delta}{\Gamma \vdash (t \ \ [x.u, y.v]) : C; \Delta} \vee_e$$

$$\frac{\Gamma \vdash t : A; \Delta, a : A}{\Gamma \vdash (a \ \ t) : \bot; \Delta, a : A} abs_i \qquad \frac{\Gamma \vdash t : \bot; \Delta, a : A}{\Gamma \vdash \mu a.t : A; \Delta} abs_e$$

4. *The cut-elimination procedure corresponds to the reduction rules given bellow. There are three kinds of cuts:*

   (a) *The logical cuts: They appear when the introduction of a connective is immediately followed by its elimination. The corresponding rules are:*

   - $(\lambda x.u \ \ v) \triangleright u[x := v]$
   - $(\langle t_1, t_2 \rangle \ \ \pi_i) \triangleright t_i$
   - $(\omega_i t \ \ [x_1.u_1, x_2.u_2]) \triangleright u_i[x_i := t]$

   (b) *The permutative cuts: They appear when the elimination of the disjunction is followed by the elimination rule of a connective.The corresponding rule is:*

   - $((t \ \ [x_1.u_1, x_2.u_2]) \ \ \varepsilon) \triangleright (t \ \ [x_1.(u_1 \ \varepsilon), x_2.(u_2 \ \varepsilon)])$

   (c) *The classical cuts: They appear when the classical rule is followed by the elimination rule of a connective. The corresponding rule is:*

   - $(\mu a.t \ \ \varepsilon) \triangleright \mu a.t[a :=^* \varepsilon]$, *where* $t[a :=^* \varepsilon]$ *is obtained from* t *by replacing inductively each subterm in the form* $(a \ v)$ *by* $(a \ (v \ \varepsilon))$.

Notation 3.2.1 *Let* t *and* t' *be* $\mathcal{E}$*-terms. The notation* $t \triangleright t'$ *means that* t *reduces to* t' *by using one step of the reduction rules given above. Similarly,* $t \triangleright^* t'$ *means that* t *reduces to* t' *by using some steps of the reduction rules given above.*

The following result is straightforward.

Theorem 3.2.1 *If* $\Gamma \vdash t : A; \Delta$ *and* $t \triangleright^* t'$ *then* $\Gamma \vdash t' : A; \Delta$.

We have also the confluence property (see [1], [4] and [11]).

Theorem 3.2.2 *If* $t \triangleright^* t_1$ *and* $t \triangleright^* t_2$, *then there exists* t_3 *such that* $t_1 \triangleright^* t_3$ *and* $t_2 \triangleright^* t_3$.

Definition 3.2.2 *An* $\mathcal{E}$*-term* t *is said to be strongly normalizable if there is no infinite sequence* $(t_i)_{i<\omega}$ *of* $\mathcal{E}$*-terms such that* $t_0 = t$ *and* $t_i \triangleright t_{i+1}$ *for all* $i < \omega$.

The aim of this chapter is to prove the following theorem.

Theorem 3.2.3 *Every typed term is strongly normalizable.*

In the rest of the chapter we consider only typed terms.

3.3 Reducibility candidates

Lemma 3.3.1 *Let t, u and u' be $\mathcal{E}$-terms such that $u \triangleright u'$, then:*

1. *$u[x := t] \triangleright u'[x := t]$ and $u[a :=^* t] \triangleright u'[a :=^* t]$.*
2. *$t[x := u] \triangleright^* t[x := u']$ and $t[a :=^* u] \triangleright^* t[a :=^* u']$.*

Proof. 1) By induction on u. 2) By induction on t. ■

Notation 3.3.1 *The set of strongly normalizable terms (resp. $\mathcal{E}$-terms) is denoted by $\mathcal{N}$ (resp. $\mathcal{N}'$). If $t \in \mathcal{N}'$, we denoted by $\eta(t)$ the maximal length of the reduction sequences of t. We denote also $\mathcal{N}'^{<\omega}$ the set of finite sequences of $\mathcal{N}'$.*

Definition 3.3.1 *Let $\bar{w} = w_1...w_n \in \mathcal{N}'^{<\omega}$, we say that $\bar{w}$ is a nice sequence iff w_n is the only $\mathcal{E}$-term in $\bar{w}$ which can be in the form $[x.u, y.v]$.*

Remark 3.3.1 *The intuition behind the notion of the nice sequences will be given in the proof of the lemma 3.3.3.*

Lemma 3.3.2 *Let $\bar{w} = w_1...w_n$ be a nice sequence and $\bar{w'} = w_1...w'_i...w_n$ where $w_i \triangleright w'_i$. Then $\bar{w'}$ is also a nice sequence.*

Proof. This comes from the fact that if $\varepsilon \triangleright [x.u, y.v]$ then $\varepsilon = [x.p, y.q]$, where $p \triangleright u$ or $q \triangleright v$. ■

Notation 3.3.2

1. *The empty sequence is denoted by $\emptyset$.*
2. *Let $\bar{w} = w_1...w_n$ a sequence of $\mathcal{E}$-terms and t a term. Then $(t\ \bar{w})$ is t if $n = 0$ and $((t\ w_1)\ w_2...w_n)$ if $n \neq 0$. The term $t[a :=^* \bar{w}]$ is obtained from t by replacing inductively each subterm in the form $(a\ v)$ by $(a\ (v\ \bar{w}))$.*
3. *If $\bar{w} = w_1...w_n$ is a nice sequence, we denote $\eta(\bar{w}) = \sum_{i=1}^{n} \eta(w_i)$.*

Lemma 3.3.3 *Let $\bar{w}$ be a nice sequence.*

1. *$(x\,\bar{w}) \in \mathcal{N}$.*
2. *If $u \in \mathcal{N}$ and $(t[x := u]\ \bar{w}) \in \mathcal{N}$, then $((\lambda x.t\ u)\ \bar{w}) \in \mathcal{N}$.*
3. *If $t_1, t_2 \in \mathcal{N}$ and $(t_i\ \bar{w}) \in \mathcal{N}$, then $((\langle t_1, t_2\rangle\ \pi_i)\ \bar{w}) \in \mathcal{N}$.*
4. *If $t, u_1, u_2 \in \mathcal{N}$ and $u_i[x_i := t] \in \mathcal{N}$, then $(\omega_i t\ \ [x_1.u_1, x_2.u_2]) \in \mathcal{N}$.*
5. *If $t[a :=^* \bar{w}] \in \mathcal{N}$, then $(\mu a.t\ \bar{w}) \in \mathcal{N}$.*

Proof.

1. Let $\bar{w} = w_1...w_n$. All reduction over $(x\ \bar{w})$ take place in some w_i, because $\bar{w}$ is a nice sequence, and therefore the w_i cannot interact between them via commutative reductions. Since all w_i are strongly normalizable, then $(x\ \bar{w})$ itself is strongly normalizable.

2. It suffices to prove that: If $((\lambda x.t\ u)\ \bar{w}) \triangleright s$, then $s \in \mathcal{N}$. We process by induction on $\eta(u) + \eta(t[x := u]\ \bar{w})$. Since $\bar{w} = w_1...w_n$ is a nice sequence, the w_i cannot interact between them via commutative reductions. We have four possibilities for the term s.

   - $s = ((\lambda x.t'\ u)\ \bar{w})$ where $t \triangleright t'$: By lemma 3.3.1, $(t'[x := u]\ \bar{w}) \in \mathcal{N}$ and $\eta(u) + \eta((t'[x := u]\ \bar{w})) < \eta(u) + \eta((t[x := u]\ \bar{w}))$, then, by induction hypothesis, $s \in \mathcal{N}$.
   - $s = ((\lambda x.t\ u')\ \bar{w})$ where $u \triangleright u'$: By lemma 3.3.1, $(t[x := u']\ \bar{w}) \in \mathcal{N}$ and $\eta(u') + \eta((t[x := u']\ \bar{w})) < \eta(u) + \eta((t[x := u]\ \bar{w}))$, then, by induction hypothesis, $s \in \mathcal{N}$.
   - $s = ((\lambda x.t\ u)\ \bar{w'})$ where $\bar{w'} = w_1...w'_i...w_n$ and $w_i \triangleright w'_i$: By lemma 3.3.2, $\bar{w'}$ is a nice sequence. We have $(t[x := u]\ \bar{w'}) \in \mathcal{N}$ and $\eta(u) + \eta((t[x := u]\ \bar{w'})) < \eta(u) + \eta((t[x := u]\ \bar{w}))$, then, by induction hypothesis, $s \in \mathcal{N}$.
   - $s = (t[x := u]\ \bar{w})$: By hypothesis, $s \in \mathcal{N}$.

3. Same proof as 2).

4. Same proof as 2).

5. It suffices also to prove that: If $(\mu a.t\ \bar{w}) \triangleright s$, then $s \in \mathcal{N}$. We process by induction on the pair $(lg(\bar{w}), \eta(t[a :=^* \bar{w}]) + \eta(\bar{w}))$ where $lg(\bar{w})$ is the number of the $\mathcal{E}$-terms in the sequence $\bar{w}$. We have three possibilities for the term s.

   - $s = (\mu a.t'\ \bar{w})$ where $t \triangleright t'$: By lemma 3.3.1, $t'[a :=^* \bar{w}] \in \mathcal{N}$ and $\eta(t'[a :=^* \bar{w}]) < \eta(t[a :=^* \bar{w}])$, then, by induction hypothesis, $s \in \mathcal{N}$.
   - $s = (\mu a.t\ \bar{w'})$ where $\bar{w'} = w_1...w'_i...w_n$ and $w_i \triangleright w'_i$: by lemma 3.3.2, $\bar{w'}$ is a nice sequence and, by lemma 3.3.1, $t[a :=^* \bar{w'}] \in \mathcal{N}$ and $\eta(t[a :=^* \bar{w'}]) + \eta(\bar{w'}) < \eta(t[a :=^* \bar{w}]) + \eta(\bar{w})$, then, by induction hypothesis, $s \in \mathcal{N}$.
   - $s = (\mu a.t[a :=^* w_1]\,\bar{w'})$ where $\bar{w'} = w_2...w_n$: It is obvious that $\bar{w'}$ is a nice sequence and $lg(\bar{w'}) < lg(\bar{w})$. We have $t[a :=^* w_1][a :=^* \bar{w'}] = t[a :=^* \bar{w}] \in \mathcal{N}$, then, by induction hypothesis, $s \in \mathcal{N}$.

■

Lemma 3.3.4 *Let $\bar{w}$ be a nice sequence.*
If $(t\ [x.(u\ \bar{w}), y.(v\ \bar{w})]) \in \mathcal{N}$, then $((t\ [x.u, y.v])\ \bar{w}) \in \mathcal{N}$.

Proof. This is proved by that, from an infinite sequence of reduction starting from $((t\ [x.u, y.v])\ \bar{w})$, an infinite sequence of reduction starting from $(t\ [x.(u\ \bar{w}), y.(v\ \bar{w})])$ can be constructed. A complete proof of this result is given in [3] in order to characterize the strongly normalizable terms. ■

Definition 3.3.2 *1. We define three functional constructions ($\rightsquigarrow$, $\curlywedge$ and $\curlyvee$) on subsets of terms:*

(a) $\mathcal{K} \rightsquigarrow \mathcal{L} = \{t \in \mathcal{T} /$ for each $u \in \mathcal{K}$, $(t\ u) \in \mathcal{L}\}$.

(b) $\mathcal{K} \curlywedge \mathcal{L} = \{t \in \mathcal{T} /\ (t\ \pi_1) \in \mathcal{K}$ and $(t\ \pi_2) \in \mathcal{L}\}$.

(c) $\mathcal{K} \curlyvee \mathcal{L} = \{t \in \mathcal{T} /$ for each $u, v \in \mathcal{N}$: If (for each $r \in \mathcal{K}$,$s \in \mathcal{L}$: $u[x := r] \in \mathcal{N}$ and $v[y := s] \in \mathcal{N}$), then $(t\ [x.u, y.v]) \in \mathcal{N}\}$.

2. The set $\mathcal{R}$ of the reductibility candidates is the smallest set of subsets of terms containing $\mathcal{N}$ and closed by the functional constructions $\rightsquigarrow$, $\curlywedge$ and $\curlyvee$.

3. Let $\bar{w} = w_1...w_n$ be a sequence of $\mathcal{E}$-terms, we say that $\bar{w}$ is a good sequence iff for each $1 \leq i \leq n$, w_i is not in the form $[x.u, y.v]$.

Lemma 3.3.5 *If $R \in \mathcal{R}$, then:*

1. $R \subseteq \mathcal{N}$.

2. R contains the λ-variables.

Proof. We prove, by simultaneous induction, that $R \subseteq \mathcal{N}$ and for each λ-variable x and for each good sequence $\bar{w} \in \mathcal{N}'^{<\omega}$, $(x\ \ \bar{w}) \in R$.

- $R = \mathcal{N}$: trivial.
- $R = R_1 \rightsquigarrow R_2$: Let $t \in R$. By induction hypothesis, we have $x \in R_1$, then $(t\ x) \in R_2$, therefore, by induction hypothesis, $(t\ x) \in \mathcal{N}$ hence $t \in \mathcal{N}$.

  Let $\bar{w} \in \mathcal{N}'^{<\omega}$ be a good sequence and $v \in R_1$. Since $\bar{w}v$ is a good sequence, then, by induction hypothesis $(x\ \bar{w}v) \in R_2$, therefore $(x\ \bar{w}) \in R_1 \rightsquigarrow R_2$.
- $R = R_1 \curlywedge R_2$: Let $t \in R$, then $(t\ \pi_i) \in R_i$ and, by induction hypothesis, $(t\ \pi_i) \in \mathcal{N}$, therefore $t \in \mathcal{N}$.

  Let $\bar{w} \in \mathcal{N}'^{<\omega}$ be a good sequence, then $\bar{w}\pi_i$ is also a good sequence and, by induction hypothesis, $(x\ \bar{w}\pi_i) \in R_i$, therefore $(x\ \bar{w}) \in R$.

- $R = R_1 \curlyvee R_2$: Let $t \in R$ and y, z two λ-variables. By induction hypothesis, we have, for each $u \in R_1 \subseteq \mathcal{N}$ and $v \in R_2 \subseteq \mathcal{N}$, $y[y := u] = u \in \mathcal{N}$ and $z[z := v] = v \in \mathcal{N}$, then $(t\ [y.y, z.z]) \in \mathcal{N}$, therefore $t \in \mathcal{N}$.

  Let $\bar{w} \in \mathcal{N}'^{<\omega}$ be a good sequence and $u, v \in \mathcal{N}$ such that for each $r \in R_1, s \in R_2, u[x := r] \in \mathcal{N}$ and $v[y := s] \in \mathcal{N}$. We have $[x.u, y.v] \in \mathcal{N}'$ because u and $v \in \mathcal{N}$. Thus $\bar{w}\ [x.u, y.v]$ is a nice sequence, and by lemma 3.3.3, $(x\ \bar{w}\ [x.u, y.v]) \in \mathcal{N}$, therefore $(x\ \bar{w}) \in R$.

■

Notation 3.3.3 *For $\mathcal{S} \subseteq \mathcal{N}'^{<\omega}$, we define $\mathcal{S} \rightsquigarrow \mathcal{K} = \{t \in \mathcal{T} /$ for each $\bar{w} \in \mathcal{S}, (t\ \bar{w}) \in \mathcal{K}\}$.*

Definition 3.3.3 *A set $\mathcal{X} \subseteq \mathcal{N}'^{<\omega}$ is said to be nice iff for each $\bar{w} \in \mathcal{X}$, $\bar{w}$ is a nice sequence.*

Lemma 3.3.6 *Let $R \in \mathcal{R}$, then there exists a nice set $\mathcal{X}$ such that $R = \mathcal{X} \rightsquigarrow \mathcal{N}$.*

Proof. By induction on R.

- $R = \mathcal{N}$: Take $\mathcal{X} = \{\emptyset\}$, it is clear that $\mathcal{N} = \{\emptyset\} \rightsquigarrow \mathcal{N}$.
- $R = R_1 \rightsquigarrow R_2$: We have $R_2 = \mathcal{X}_2 \rightsquigarrow \mathcal{N}$ for a nice set $\mathcal{X}_2$. Take $\mathcal{X} = \{u\ \bar{v}$ / $u \in R_1, \bar{v} \in \mathcal{X}_2\}$. We have $u\ \bar{v}$ is a nice sequence for all $u \in R_1$ and $\bar{v} \in \mathcal{X}_2$. Then $\mathcal{X}$ is a nice set and we can easily check that $R = \mathcal{X} \rightsquigarrow \mathcal{N}$.
- $R = R_1 \curlywedge R_2$: Similar to the previous case.
- $R = R_1 \curlyvee R_2$: Take $\mathcal{X} = \{[x.u, y.v]$ / for each $r \in R_1$ and $s \in R_2$, $u[x := r] \in \mathcal{N}$ and $v[y := s] \in \mathcal{N}\}$. We have $\mathcal{X}$ is a nice set and, by definition, $R = \mathcal{X} \rightsquigarrow \mathcal{N}$.

■

Remark 3.3.2 *Let $R \in \mathcal{R}$ and $\mathcal{X}$ a nice set such that $R = \mathcal{X} \rightsquigarrow \mathcal{N}$. We can suppose that $\emptyset \in \mathcal{X}$. Indeed, since $R \subseteq \mathcal{N}$, we have also $R = \mathcal{X} \cup \{\emptyset\} \rightsquigarrow \mathcal{N}$.*

Definition 3.3.4 *Let $R \in \mathcal{R}$, we define $R^{\perp} = \cup\{\mathcal{X}$ / $R = \mathcal{X} \rightsquigarrow \mathcal{N}$ and $\mathcal{X}$ is a nice set$\}$.*

Lemma 3.3.7 *Let $R \in \mathcal{R}$, then:*

1. *$R^{\perp}$ is a nice set.*
2. *$R = R^{\perp} \rightsquigarrow \mathcal{N}$.*

Proof.

1. By definition.
2. This comes also from the fact that: If, for every $i \in I$, $R = \mathcal{X}_i \rightsquigarrow \mathcal{N}$, then $R = \cup_{i \in I} \mathcal{X}_i \rightsquigarrow \mathcal{N}$.

■

Remark 3.3.3 *For $R \in \mathcal{R}$, $R^\perp$ is simply the greatest nice $\mathcal{X}$ such that $R = \mathcal{X} \rightsquigarrow \mathcal{N}$. In fact any nice $\mathcal{X}$ such that $\emptyset \in \mathcal{X}$ and $R = \mathcal{X} \rightsquigarrow \mathcal{N}$ would work as well as $R^\perp$.*

Lemma 3.3.8 *Let $R \in \mathcal{R}$, $t \in R$ and $t \rhd^* t'$. Then $t' \in R$*

Proof. Let $\bar{u} \in R^\perp$. We have $(t\ \bar{u}) \rhd^* (t'\ \bar{u})$ and $(t\ \bar{u}) \in \mathcal{N}$, then $(t'\ \bar{u}) \in \mathcal{N}$. We deduce that $t' \in R^\perp \rightsquigarrow \mathcal{N} = R$. ■

Remark 3.3.4 *Let $R \in \mathcal{R}$, we have not in general $\mathcal{N} \subseteq R$, but we can prove, by induction, that $\mu a \mathcal{N} = \{\mu a.t \ / \ t \in \mathcal{N}$ and a is not free in $t\} \subseteq R$.*

3.4 Proof of the theorem 3.2.3

Definition 3.4.1 *An interpretation is a function I from the propositional variables to $\mathcal{R}$, which we extend to any formula as follows: $I(\perp) = \mathcal{N}$, $I(A \to B) = I(A) \rightsquigarrow I(B)$, $I(A \wedge B) = I(A) \curlywedge I(B)$ and $I(A \vee B) = I(A) \curlyvee I(B)$.*

Lemma 3.4.1 (Adequation lemma) *Let $\Gamma = \{x_i : A_i\}_{1 \leq i \leq n}$, $\Delta = \{a_j : B_j\}_{1 \leq j \leq m}$, I an interpretation, $u_i \in I(A_i)$, $\bar{v_j} \in I(B_j)^\perp$ and t such that:*
$\Gamma \vdash t : A \ ; \Delta$, then, $t[x_1 := u_1, ..., x_n := u_n, a_1 :=^* \bar{v_1}, ..., a_m :=^* \bar{v_m}] \in I(A)$.

Proof. For each term s, we denote
$s[x_1 := u_1, ..., x_n := u_n, a_1 :=^* \bar{v_1}, ..., a_m :=^* \bar{v_m}]$ by s'.
We look at the last used rule in the derivation of $\Gamma \vdash t : A \ ; \Delta$.

- ax, $\to_e$ and $\wedge_e^j$: Easy.
- $\to_i$: In this case $t = \lambda x.t_1$ with $\Gamma, x : C \vdash t_1 : D \ ; \Delta$ and $A = C \to D$. Let $u \in I(C)$ and $\bar{w} \in I(D)^\perp$. By induction hypothesis, we have $t_1'[x := u] \in I(D)$, then $(t_1'[x := u]\ \bar{w}) \in \mathcal{N}$, and, by lemma 3.3.3 $((\lambda x.t_1'\ u)\, \bar{w}) \in \mathcal{N}$. Therefore $(\lambda x.t_1'\ u) \in I(D)$, hence $\lambda x.t_1' \in I(C) \rightsquigarrow I(D) = I(A)$.
- $\wedge_i$ similar to $\to_i$.

- $\vee_i^1$ (a similar proof for $\vee_i^2$): In this case $t = \omega_1 t_1$ with $\Gamma \vdash t_1 : A_1;\ \Delta$ and $A = A_1 \vee A_2$. By induction hypothesis $t'_1 \in I(A_1)$. Let $u,\ v \in \mathcal{N}$, let $r \in I(A_1)$ and $s \in I(A_2)$ such that: $u[x := r] \in \mathcal{N}$ and $v[y := s] \in \mathcal{N}$. We have to prove that $(\omega_1 t'_1\,[x.u, y.v] \in \mathcal{N})$, this is true since $u,\ v \in \mathcal{N}$ and $u[x := t'_1] \in \mathcal{N}$ (take $r = t'_1$).

- $\vee_e$: In this case $t = (t_1\ [x.u, y.v])$ with $\Gamma \vdash t_1 : B \vee C\ ;\ \Delta\ ,\ \Gamma, x : B \vdash u : A\ ;\ \Delta$ and $\Gamma, y : C \vdash v : A\ ;\ \Delta$. Let $r \in I(B)$ and $s \in I(C)$. By induction hypothesis, we have $t'_1 \in I(B) \curlyvee I(C)$, $u'[x := r] \in I(A)$ and $v'[y := s] \in I(A)$. Let $\bar{w} \in I(A)^{\perp}$, then $(u'[x := r]\,\bar{w}) \in \mathcal{N}$ and $(v'[y := s]\,\bar{w}) \in \mathcal{N}$, therefore $(t'_1\,[x.(u'\bar{w}), y.(v'\bar{w})]) \in \mathcal{N}$. By lemma 3.3.4, $((t'_1\ [x.u', y.v'])\bar{w}) \in \mathcal{N}$, therefore $(t'_1\ [x.u', y.v']) \in I(A)$.

- abs_e: In this case $t = \mu a.u$ and $\Gamma \vdash \mu a.u : A\ ;\ \Delta$. Let $\bar{v} \in I(A)^{\perp}$. It suffices to prove that $((\mu a.u')\ \bar{v}) \in \mathcal{N}$. By the induction hypothesis, $u'[a :=^* \bar{v}] \in I(\perp) = \mathcal{N}$, then, by lemma 3.3.3, $(\mu a.u'\ \bar{v}) \in \mathcal{N}$. Finally $(\mu a.u)' \in I(A)$.

- abs_i: In this case $t = (a_j\ u)$ and $\Gamma \vdash (a_j\ u) :\perp\ ;\ \Delta', a_j : B_j$. We have to prove that $t' \in \mathcal{N}$, by induction hypothesis, $u' \in I(B_j)$, then $(u'\ \bar{v_j}) \in \mathcal{N}$, therefore $t' = (a\ (u'\ \bar{v_j})) \in \mathcal{N}$.

■

Notation 3.4.1 *We denote $I_{\mathcal{N}}$ the interpretation such that, for each propositional variable X, $I_{\mathcal{N}}(X) = \mathcal{N}$.*

Proof.[of theorem 3.2.3]: If $x_1 : A_1, ..., x_n : A_n \vdash t : A; a_1 : B_1, ..., a_m : B_m$, then, by the lemma 3.3.5, $x_i \in I_{\mathcal{N}}(A_i)$, and, by definition, $\emptyset \in I_{\mathcal{N}}(B_j)^{\perp}$. Therefore by lemma 3.4.1, $t = t[x_1 := x_1, ..., x_n := x_n, a_1 :=^* \emptyset, ..., a_m :=^* \emptyset] \in I_{\mathcal{N}}(A)$ and finally, by lemma 3.3.5, $t \in \mathcal{N}$. ■

Remark 3.4.1 *We can give now another proof of remark 3.3.4: "if $R \in \mathcal{R}$, the $\mu a.\mathcal{N} \subseteq R$". Let $t = \lambda z.\mu a.z$, we have $\vdash t :\perp \to p$ for every propositional variable p. By lemma 3.4.1, for every $R \in \mathcal{R}$, $t \in \mathcal{N} \rightsquigarrow R$, then, for every $u \in \mathcal{N}$, $(t\ u) \in R$, therefore, by lemma 3.3.8, $\mu a.u \in R$.*

We end this chapter by giving characterization of the normal terms, then deriving the subformula property. All this work is already presented in [4].

Definition 3.4.2 *Consider the following grammar:*

$$\mathcal{P} \ :=\ \lambda\mathcal{X}.\mathcal{P} \mid \langle \mathcal{P}, \mathcal{P} \rangle \mid \omega_1\mathcal{P} \mid \omega_2\mathcal{P} \mid (\mathcal{Q}\,[\mathcal{X}.\mathcal{P}, \mathcal{X}.\mathcal{P}]) \mid \mu\mathcal{A}.\mathcal{P} \mid \mathcal{Q}$$

$$\mathcal{Q} \ :=\ \mathcal{X} \mid (\mathcal{Q}\ \ \mathcal{P}) \mid (\mathcal{P}\,\pi_1) \mid (\mathcal{P}\,\pi_2) \mid (\mathcal{A}\ \ \mathcal{P})$$

A term which belongs to $\mathcal{P}$ (resp $\mathcal{Q}$) is said to be $\mathcal{P}$-canonical (Q-canonical).

Lemma 3.4.2 *Let* Π *be a derivation of* $\Gamma \vdash t : A \ ; \Delta$*, then*

1. *If* t *is* $\mathcal{Q}$*-canonical then every type occuring in* Π *is either* $\bot$*, or a subformula of a type occuring in* Γ *or* Δ*.*

2. *If* t *is* $\mathcal{P}$*-canonical then every type occuring in* Π *is either* $\bot$*, or a subformula of a type occuring in* Γ *or* Δ*, or a subformula of* A*.*

Proof. By a simulateous induction on 1 and 2. We look at the last used rule in Π.

■

Lemma 3.4.3 *Let* $\Gamma \vdash t : A \ ; \Delta$*, if* t *is normal, then* t *is* $\mathcal{P}$*-canonical.*

Proof. By induction on the derivation of $\Gamma \vdash t : A \ ; \Delta$.

■

Corollary 3.4.1 (The subformula property) *Let* $\Gamma \vdash t : A \ ; \Delta$*, if* t *is normal, then every type occuring in* Π *is either* $\bot$*, or a subformula of a type occuring in* Γ *or* Δ*, or a subformula of* A*.*

Proof. A direct consequence of the two precedent lemmas.

■

Bibliography

[1] Y. Andou. *Church-Rosser property of simple reduction for full first-order classical natural deduction.* Annals of Pure and Applied logic 119 (2003) 225-237.

[2] R. David and K. Nour. *A short proof of the strong normalization of the simply typed $\lambda\mu$-calculus.* Schedae Informaticae vol.12, pp. 27-33, 2003.

[3] R. David and K. Nour. *A short proof of the Strong Normalization of Classical Natural Deduction with Disjunction.* Journal of symbolic Logic, vol 68, num 4, pp 1277-1288, 2003.

[4] Ph. de Groote. *Strong normalization of classical natural deduction with disjunction.* In 5th International Conference on typed lambda calculi and applications, TLCA'01. LNCS (2044), pp. 182-196. Springer Verlag, 2001.

[5] J.-Y. Girard, Y. Lafont, P. Taylor. Proofs and types. *Cambridge University Press*, 1986.

[6] F. Joachimski and R. Matthes. *Short proofs of normalization for the simply-typed lambda-calculus, permutative conversions and Gödel's T.* Archive for Mathematical Logic 42, pp 59-87 (2003).

[7] R. Matthes. *Non-Strictly Positive Fixed Point for Classical Natural Deduction.* APAL, vol 133, pp. 205-230, 2005.

[8] K. Nakazawa. *Confluency and strong normalizability of call-by-value $\lambda\mu$-calculus.* Theoritical Computer Science, vol 290, pp. 429-463. 2003.

[9] K. Nakazawa and M. Tatsuta. *Strong normalization proof with CPS-Translation for the second order classical natural deduction.* The Journal of Symbolic Logic, vol 68, num 3, pp. 851-859. Sept 2003.

[10] K. Nour and K. Saber *A semantical proof of the strong normalization theorem for full propositional classical natural deduction.* Archive for Mathematical Logic, vol 45, pp. 357-364, 2005.

[11] K. Nour and K. Saber *Some properties of the $\lambda\mu$-calculus.* Manuscript, 2007.

[12] M. Parigot *$\lambda\mu$-calculus: An algorithm interpretation of classical natural deduction.* Lecture Notes in Artificial Intelligence (624), pp. 190-201. Springer Verlag 1992.

[13] M. Parigot. *Proofs of strong normalization for second order classical natural deduction.* Journal of Symbolic Logic, 62 (4), pp. 1461-1479, 1997.

Chapter 4

A Semantics of Realisability

4.1 Introduction

Natural deduction system is one of the main logical system which was introduced by Gentzen [4] to study the notion of proof. The full classical natural deduction system is well adapted for the human reasoning. By full we mean that all the connectives ($\rightarrow$, $\wedge$ and $\vee$) and $\bot$ (for the absurdity) are considered as primitive and they have their intuitionistic meaning. As usual, the negation is defined by $\neg A = A \rightarrow \bot$. Considering this logic from the computer science of view is interesting because, by the Curry-Howard correspondence, formulas can be seen as types for the functional programming languages and correct programs can be extracted. By this isomorphism the corresponding calculus is an extension of the $\lambda\mu$-calculus with product and co-product.

Until very recently (see the introduction of [3] for a brief history), no proof of the strong normalization of the cut-elimination procedure was known for full logic. In [3], Ph. De Groote gives a such proof for classical propositional natural deduction, by using the CPS-transformations. However its proof is not finished yet, since the modified CPS-translation does not always preserve the stricteness of reductions, and this for the same reasons pointed out in [7] and [8]. R. David and K. Nour give in [2] a direct and syntactical proof of this result. R. Matthes recently found another semantical proof of this result (see [6]).

In order to prove the strong normalization of classical propositional natural deduction, we introduce in [10] a variant of the reducibility candidates, which was already present in [14]. This method has been introduced by J.Y. Girard. It consists in associating to each type A a set of terms $|A|$, such that every term is in the interpretation of its type (this is called "the adequation lemma"). To the best of our knowledge, we obtain the shortest proof of this result.

In this chapter, we define a semantics of realisability of classical propositional natural deduction inspired by [10] and we estabilish a correctness theorem. The idea is to replace the set of strongly normalizing terms used in the proof presented

in [10] by a set having the properties necessary to keep the adequation lemma. This result allows to characterize the operational behaviour of terms having some particular types.

The chapter is organized as follows. Section 2 is an introduction to the typed system and the relative cut-elimination procedure. In section 3, we define the semantics of realisability and we prove the correctness theorem. In section 4, we give some applications of this result.

4.2 Notations and definitions

Definition 4.2.1 *We use notations inspired by the paper [1].*

1. *Let $\mathcal{X}$ and $\mathcal{A}$ be two disjoint infinite alphabets for distinguishing the λ-variables and μ-variables respectively. We code deductions by using a set of terms $\mathcal{T}$ which extends the λ-terms and is given by the following grammars:*

$$\mathcal{T} \ := \ \mathcal{X} \mid \lambda\mathcal{X}.\mathcal{T} \mid (\mathcal{T} \ \ \mathcal{E}) \mid \langle \mathcal{T}, \mathcal{T} \rangle \mid \omega_1\mathcal{T} \mid \omega_2\mathcal{T} \mid \mu\mathcal{A}.\mathcal{T} \mid (\mathcal{A} \ \ \mathcal{T})$$
$$\mathcal{E} \ := \ \mathcal{T} \mid \pi_1 \mid \ \ \pi_2 \mid [\mathcal{X}.\mathcal{T}, \mathcal{X}.\mathcal{T}]$$

   An element of the set $\mathcal{E}$ is said to be an $\mathcal{E}$-term.

2. *Types are formulas of propositional logic built on the set of propositional variables and the constant type $\bot$, using the connectors $\rightarrow$, $\wedge$ and $\vee$.*

3. *The meaning of the new constructors is given by the typing rules below where Γ (resp. Δ) is a context, i.e. a set of declarations of the form $x : A$ (resp. $a : A$) where x is a λ-variable (resp. a is a μ-variable) and A is a formula.*

$$\frac{}{\Gamma, x : A \ \vdash x : A \ ; \Delta} ax$$

$$\frac{\Gamma, x : A \vdash t : B; \Delta}{\Gamma \vdash \lambda x.t : A \rightarrow B; \Delta} \rightarrow_i \qquad \frac{\Gamma \vdash u : A \rightarrow B; \Delta \quad \Gamma \vdash v : A; \Delta}{\Gamma \vdash (u \ \ v) : B; \Delta} \rightarrow_e$$

$$\frac{\Gamma \vdash u : A; \Delta \quad \Gamma \vdash v : B; \Delta}{\Gamma \vdash \langle u, v \rangle : A \wedge B; \Delta} \wedge_i$$

$$\frac{\Gamma \vdash t : A \wedge B; \Delta}{\Gamma \vdash (t \ \ \pi_1) : A; \Delta} \wedge_e^1 \quad \frac{\Gamma \vdash t : A \wedge B; \Delta}{\Gamma \vdash (t \ \ \pi_2) : B; \Delta} \wedge_e^2$$

$$\frac{\Gamma \vdash t : A; \Delta}{\Gamma \vdash \omega_1 t : A \vee B; \Delta} \vee_i^1 \quad \frac{\Gamma \vdash t : B; \Delta}{\Gamma \vdash \omega_2 t : A \vee B; \Delta} \vee_i^2$$

$$\frac{\Gamma \vdash t : A \vee B; \Delta \quad \Gamma, x : A \vdash u : C; \Delta \quad \Gamma, y : B \vdash v : C; \Delta}{\Gamma \vdash (t \ \ [x.u, y.v]) : C; \Delta} \vee_e$$

$$\frac{\Gamma \vdash t : A; \Delta, a : A}{\Gamma \vdash (a \ \ t) : \bot; \Delta, a : A} abs_i \quad \frac{\Gamma \vdash t : \bot; \Delta, a : A}{\Gamma \vdash \mu a.t : A; \Delta} abs_e$$

4. *The cut-elimination procedure corresponds to the reduction rules given below. They are those we need to the subformula property.*

   - $(\lambda x.u \ \ v) \triangleright u[x := v]$
   - $(\langle t_1, t_2 \rangle \ \ \pi_i) \triangleright t_i$
   - $(\omega_i t \ \ [x_1.u_1, x_2.u_2]) \triangleright u_i[x_i := t]$
   - $((t \ \ [x_1.u_1, x_2.u_2]) \ \ \varepsilon) \triangleright (t \ \ [x_1.(u_1 \ \varepsilon), x_2.(u_2 \ \varepsilon)])$
   - $(\mu a.t \ \ \varepsilon) \triangleright \mu a.t[a :=^* \varepsilon]$.

   where $t[a :=^* \varepsilon]$ is obtained from t by replacing inductively each subterm in the form $(a \ v)$ by $(a \ (v \ \varepsilon))$.

5. *Let t and t' be $\mathcal{E}$-terms. The notation $t \triangleright t'$ means that t reduces to t' by using one step of the reduction rules given above. Similarly, $t \triangleright^* t'$ means that t reduces to t' by using some steps of the reduction rules given above.*

The following result is straightforward

Theorem 4.2.1 (Subject reduction) *If $\Gamma \vdash t : A; \Delta$ and $t \triangleright^* t'$, then $\Gamma \vdash t' : A; \Delta$.*

We have also the following properties (see [1], [2], [3], [10] and [11]).

Theorem 4.2.2 (Confluence) *If $t \triangleright^* t_1$ and $t \triangleright^* t_2$, then there exists t_3 such that $t_1 \triangleright^* t_3$ and $t_2 \triangleright^* t_3$.*

Theorem 4.2.3 (Strong normalization) *If $\Gamma \vdash t : A; \Delta$, then t is strongly normalizable.*

4.3 The semantics

Definition 4.3.1 1. *We denote by $\mathcal{E}^{<\omega}$ the set of finite sequences of $\mathcal{E}$-terms. The empty sequence is denoted by $\emptyset$.*

2. *We denote by $\bar{w}$ the sequence $w_1 w_2 ... w_n$. If $\bar{w} = w_1 w_2 ... w_n$, then $(t \ \bar{w})$ is t if $n = 0$ and $((t \ w_1) \ w_2 ... w_n)$ if $n \neq 0$. The term $t[a :=^* \bar{w}]$ is the term obtained from t by replacing inductively each subterm in the form $(a \ v)$ by $(a \ (v \ \bar{w}))$.*

3. *A set of terms $\mathcal{S}$ is said to be μ-saturated iff:*
   - *For each terms u and v, if $u \in \mathcal{S}$ and $v \rhd^* u$, then $v \in \mathcal{S}$.*
   - *For each $a \in \mathcal{A}$ and for each $t \in \mathcal{S}$, $\mu a.t \in \mathcal{S}$ and $(a\ t) \in \mathcal{S}$.*
4. *Consider two sets of terms $\mathcal{K}$, $\mathcal{L}$ and a μ-saturated set $\mathcal{S}$, we define new sets of terms:*
   - $\mathcal{K} \rightsquigarrow \mathcal{L} = \{t \ / \ (t\ u) \in \mathcal{L}$, *for each* $u \in \mathcal{K}\}$.
   - $\mathcal{K} \curlywedge \mathcal{L} = \{t \ / \ (t\ \pi_1) \in \mathcal{K}$ *and* $(t\ \pi_2) \in \mathcal{L}\}$.
   - $\mathcal{K} \curlyvee \mathcal{L} = \{t \ /$ *for each* u, v*: if (for each* $r \in \mathcal{K}$,$s \in \mathcal{L}$*:* $u[x := r] \in \mathcal{S}$ *and* $v[y := s] \in \mathcal{S}$*), then* $(t\ [x.u, y.v]) \in \mathcal{S}\}$.
5. *Let $\mathcal{S}$ be a μ-saturated set and $\{\mathcal{R}_i\}_{i\in I}$ subsets of terms such that $\mathcal{R}_i = \mathcal{X}_i \rightsquigarrow \mathcal{S}$ for some $\mathcal{X}_i \subseteq \mathcal{E}^{<\omega}$. A model $\mathcal{M} = \langle \mathcal{S}; \{\mathcal{R}_i\}_{i\in I}\rangle$ is the smallest set of subsets of terms containing $\mathcal{S}$ and $\mathcal{R}_i$ and closed under constructors $\rightsquigarrow$, $\curlywedge$ and $\curlyvee$.*

Lemma 4.3.1 *Let $\mathcal{M} = \langle \mathcal{S}; \{\mathcal{R}_i\}_{i\in I}\rangle$ be a model and $\mathcal{G} \in \mathcal{M}$.*
There exists a set $\mathcal{X} \subseteq \mathcal{E}^{<\omega}$ such that $\mathcal{G} = \mathcal{X} \rightsquigarrow \mathcal{S}$.

Proof. By induction on $\mathcal{G}$.

- $\mathcal{G} = \mathcal{S}$: Take $\mathcal{X} = \{\emptyset\}$, it is clear that $\mathcal{S} = \{\emptyset\} \rightsquigarrow \mathcal{S}$.
- $\mathcal{G} = \mathcal{G}_1 \rightsquigarrow \mathcal{G}_2$: We have $\mathcal{G}_2 = \mathcal{X}_2 \rightsquigarrow \mathcal{S}$ for a certain set $\mathcal{X}_2$. Take $\mathcal{X} = \{u\ \bar{v}$ $/$ $u \in \mathcal{G}_1, \bar{v} \in \mathcal{X}_2\}$. We can easily check that $\mathcal{G} = \mathcal{X} \rightsquigarrow \mathcal{S}$.
- $\mathcal{G} = \mathcal{G}_1 \curlywedge \mathcal{G}_2$: Similar to the previous case.
- $\mathcal{G} = \mathcal{G}_1 \curlyvee \mathcal{G}_2$: Take $\mathcal{X} = \{[x.u, y.v]$ / for each $r \in \mathcal{G}_1$ and $s \in \mathcal{G}_2$, $u[x := r] \in \mathcal{S}$ and $v[y := s] \in \mathcal{S}\}$. By definition $\mathcal{G} = \mathcal{X} \rightsquigarrow \mathcal{S}$.

■

Definition 4.3.2 *Let $\mathcal{M} = \langle \mathcal{S}; \{\mathcal{R}_i\}_{i\in I}\rangle$ be a model and $\mathcal{G} \in \mathcal{M}$, we define the set $\mathcal{G}^{\perp} = \cup\{\mathcal{X} \ / \ \mathcal{G} = \mathcal{X} \rightsquigarrow \mathcal{S}\}$.*

Lemma 4.3.2 *Let $\mathcal{M} = \langle \mathcal{S}; \{\mathcal{R}_i\}_{i\in I}\rangle$ be a model and $\mathcal{G} \in \mathcal{M}$.*
We have $\mathcal{G} = \mathcal{G}^{\perp} \rightsquigarrow \mathcal{S}$ ($\mathcal{G}^{\perp}$ is the greatest $\mathcal{X}$ such that $\mathcal{G} = \mathcal{X} \rightsquigarrow \mathcal{S}$).

Proof. This comes from the fact that: if, for every $j \in J$, $\mathcal{G} = \mathcal{X}_j \rightsquigarrow \mathcal{S}$, then $\mathcal{G} = \cup_{j\in J}\mathcal{X}_j \rightsquigarrow \mathcal{S}$. ■

Definition 4.3.3 *1. Let $\mathcal{M} = \langle \mathcal{S}; \{\mathcal{R}_i\}_{i\in I}\rangle$ be a model. An $\mathcal{M}$-interpretation I is an application from the set of propositional variables to $\mathcal{M}$ which we extend for any type as follows:*

- $I(\bot) = \mathcal{S}$
- $I(A \to B) = I(A) \rightsquigarrow I(B)$.
- $I(A \wedge B) = I(A) \curlywedge I(B)$.
- $I(A \vee B) = I(A) \curlyvee I(B)$.

The set $|A|_{\mathcal{M}} = \cap\{I(A) \ / \ I$ an $\mathcal{M}$-interpretation$\}$ is the interpretation of A in $\mathcal{M}$.

2. The set $|A| = \cap\{|A|_{\mathcal{M}} \ / \ \mathcal{M}$ a model$\}$ is the interpretation of A.

Lemma 4.3.3 (Adequation lemma) *Let $\mathcal{M} = \langle \mathcal{S}; \{\mathcal{R}_i\}_{i\in I}\rangle$ be a model, I an $\mathcal{M}$-interpretation, $\Gamma = \{x_i : A_i\}_{1\leq i\leq n}$, $\Delta = \{a_j : B_j\}_{1\leq j\leq m}$, $u_i \in I(A_i)$, $\bar{v_j} \in I(B_j)^{\bot}$. If $\Gamma \vdash t : A; \Delta$, then,*
$t[x_1 := u_1, ..., x_n := u_n, a_1 :=^* \bar{v_1}, ..., a_m :=^* \bar{v_m}] \in I(A)$.

Proof. Let us denote by s' the term
$s[x_1 := u_1, ..., x_n := u_n, a_1 :=^* \bar{v_1}, ..., a_m :=^* \bar{v_m}]$.
The proof is by induction on the derivation, we consider the last rule:

1. ax, $\to_e$ and $\wedge_e$: Easy.

2. $\to_i$: In this case $t = \lambda x.u$ and $A = B \to C$ such that $\Gamma, x : B \vdash u : C \ ; \Delta$. By induction hypothesis, $u'[x := v] \in I(C) = I(C)^{\bot} \rightsquigarrow \mathcal{S}$ for each $v \in I(B)$, then $(u'[x := v]\ \bar{w}) \in \mathcal{S}$ for each $\bar{w} \in I(C)^{\bot}$, hence $((\lambda x.u'\ v)\ \bar{w}) \in \mathcal{S}$ because $((\lambda x.u'\ v)\ \bar{w}) \triangleright (u'[x := v]\ \bar{w})$. Therefore $t' = \lambda x.u' \in [I(B) \rightsquigarrow I(C)] = I(A)$.

3. $\wedge_i$ and $\vee_i^j$: A similar proof.

4. $\vee_e$: In this case $t = (t_1\ [x.u, y.v])$ with $(\Gamma \vdash t_1 : B \vee C; \Delta)$, $(\Gamma, x : B \vdash u : A; \Delta)$ and $(\Gamma, y : C \vdash v : A; \Delta)$. Let $r \in I(B)$ and $s \in I(C)$, by induction hypothesis, $t'_1 \in I(B) \curlyvee I(C)$, $u'[x := r] \in I(A)$ and $v'[y := s] \in I(A)$. Let $\bar{w} \in I(A)^{\bot}$, then $(u'[x := r]\ \bar{w}) \in \mathcal{S}$ and $(v'[y := s]\ \bar{w}) \in \mathcal{S}$, hence $(t'_1\ [x.(u'\ \bar{w}), y.(v'\ \bar{w})]) \in \mathcal{S}$, since $((t'_1\ [x.u', y.v')]\ \bar{w}) \triangleright (t'_1\ [x.(u'\ \bar{w}), y.(v'\ \bar{w})])$ then $((t'_1\ [x.u', y.v')]\ \bar{w}) \in \mathcal{S}$. Therefore $t' = (t'_1\ [x.u', y.v']) \in I(A)$.

5. abs_e: In this case $t = \mu a.t_1$ and $\Gamma \vdash \mu a.t_1 : A \ ; \Delta', a : A$. Let $\bar{v} \in I(A)^{\bot}$. It suffices to prove that $(\mu a.t'_1\ \bar{v}) \in \mathcal{S}$. By induction hypothesis, $t'_1[a :=^* \bar{v}] \in I(\bot) = \mathcal{S}$, then $\mu a.t'_1[a :=^* \bar{v}] \in \mathcal{S}$ and $(\mu a.t'_1\ \bar{v}) \in \mathcal{S}$.

6. abs_i: In this case $t = (a_j\ u)$ and $\Gamma \vdash (a_j\ u) : \bot; \Delta', a_j : B_j$. We have to prove that $t' \in \mathcal{S}$. By induction hypothesis $u' \in I(B_j)$, then $(u'\ \bar{v_j}) \in \mathcal{S}$, hence $t' = (a\ (u'\ \bar{v_j})) \in \mathcal{S}$.

■

Theorem 4.3.1 (Correctness theorem) *If $\vdash t : A$, then $t \in |A|$.*

Proof. Immediately from the previous lemma. ■

4.4 The operational behaviors of some typed terms

The following results are some applications of the correctness theorem.

Definition 4.4.1 *Let t be a term. We denote M_t the smallest set containing t such that: if $u \in M_t$ and $a \in \mathcal{A}$, then $\mu a.u \in M_t$ and $(a\ u) \in M_t$. Each element of M_t is denoted $\underline{\mu}.t$. For example, the term $\mu a.\mu b.(a\ (b\ (\mu c.(a\ \mu d.t))))$ is denoted by $\underline{\mu}.t$.*

In the rest of this chapter P denotes a propositional variable.

4.4.1 Terms of type $\bot \rightarrow P$ "Ex falso sequitur quodlibet"

Example 4.4.1 *Let $\mathcal{T} = \lambda z.\mu a.z$. We have $\vdash \mathcal{T} : \bot \rightarrow P$ and for every term t and $\bar{u} \in \mathcal{T}^{<\omega}$, $((\mathcal{T}\ t)\ \bar{u}) \triangleright^* \mu a.t$.*

Remark 4.4.1 *The term $(\mathcal{T}\ t)$ modelizes an instruction like* `exit`(t) (`exit` *is to be understood as in the C programming language). In the reduction of a term, if the sub-term $(\mathcal{T}\ t)$ appears in head position (the term has the form $((\mathcal{T}\ t)\ \bar{u})$), then, after some reductions, we obtain t as result.*

The general operational behavior of terms of type $\bot \rightarrow P$ is given in the following theorem:

Theorem 4.4.1 *Let T be a closed term of type $\bot \rightarrow P$, then for each λ-variable x and for each finite sequence $\bar{y}$ of λ-variables, we have $((T\ x)\ \bar{y}) \triangleright^* \underline{\mu}.x$.*

Proof. Let x be a term and $\bar{y}$ a finite sequence of λ-variables. Take $\mathcal{S} = \{t \,/\, t \triangleright^* \underline{\mu}.x\}$ and $\mathcal{R} = \{\bar{y}\} \rightsquigarrow \mathcal{S}$. It is clear that $\mathcal{S}$ is μ-saturated set and $x \in \mathcal{S}$. Let $\mathcal{M} = \langle \mathcal{S}; \mathcal{R} \rangle$ and I an $\mathcal{M}$-interpretation such that $I(P) = \mathcal{R}$. By the theorem 4.3.1, we have $T \in \mathcal{S} \rightsquigarrow (\{\bar{y}\} \rightsquigarrow \mathcal{S})$, then $((T\ x)\ \bar{y}) \in \mathcal{S}$. Therefore $((T\ x)\ \bar{y}) \triangleright^* \underline{\mu}.x$. ■

4.4.2 Terms of type $(\neg P \to P) \to P$ "Pierce law"

Example 4.4.2 *Let* $\mathcal{C}_1 = \lambda z.\mu a.(a\,(z\ \lambda y.(a\,y)))$ *and*
$\mathcal{C}_2 = \lambda z.\mu a.(a\,(z\ (\lambda x.(a\,(z\ \lambda y.(a\,x))))))$.
We have $\vdash \mathcal{C}_i : (\neg P \to P) \to P$ *for* $i \in \{1, 2\}$.
Let u, v_1, v_2 *be terms and* $\bar{t} \in \mathcal{E}^{<\omega}$, *we have :*
$((\mathcal{C}_1\ u)\ \bar{t}) \rhd^* \mu a.(a\ ((u\,\theta_1)\,\bar{t}))$ *and* $(\theta_1\ v_1) \rhd^* (a\,(v_1\ \bar{t}))$ *and*
$((\mathcal{C}_2\ u)\ \bar{t}) \rhd^* \mu a.(a\,((u\,\theta_1)\,\bar{t}))$, $(\theta_1\ v_1) \rhd^* (a\,((u\,\theta_2)\,\bar{t}))$ *and* $(\theta_2\ v_2) \rhd^* (a\,(v_1\,\bar{t}))$.

Remark 4.4.2 *The term* $\mathcal{C}_1$ *allows to modelizing the* `Call/cc` *instruction in the Scheme functional programming language.*

The following theorem describes the general operational behavior of terms with type $(\neg P \to P) \to P$.

Theorem 4.4.2 *Let* T *be a closed term of type* $(\neg P \to P) \to P$, *then for each* λ*-variable* x, *for each finite sequence* $\bar{y}$ *of* λ*-variables and for each sequence* $(z_i)_{i\in\mathbb{N}^*}$ *of* λ*-variables such that:* x *and each* y_j *are differents from any* z_i. *There exist* $m \in \mathbb{N}^*$ *and terms* $\theta_1, ..., \theta_m$ *such that we have:*

- $((T\ x)\ \bar{y}) \rhd^* \underline{\mu}.((x\,\theta_1)\,\bar{y})$
- $(\theta_k\ z_k) \rhd^* \underline{\mu}.((x\,\theta_{k+1})\,\bar{y})$ *for every* $1 \le k \le m-1$
- $(\theta_m\ z_m) \rhd^* \underline{\mu}.(z_l\,\bar{y})$ *for a certain* $1 \le l \le m$

Proof. Let x be a λ-variable, $\bar{y}$ a finite sequence of λ-variables and $(z_i)_{i\in\mathbb{N}^*}$ a sequence of λ-variables as in the theorem. Take $\mathcal{S} = \{t \ /\ \forall r \ge 0,\ \exists m \ge 0,\ \exists\,\theta_1, ..., \theta_m,\ \exists\, j\colon\ t \rhd^* \underline{\mu}.((x\ \theta_1)\,\bar{y}),\ (\theta_k\ z_{k+r}) \rhd^* \underline{\mu}.((x\,\theta_{k+1})\,\bar{y})$ for every $1 \le k \le m-1$ and $(\theta_m\ z_{m+r}) \rhd^* \underline{\mu}.(z_j\,\bar{y})\}$ and $\mathcal{R} = \{\bar{y}\} \rightsquigarrow \mathcal{S}$.

It is important to clarify the case $m = 0$ in the definition of $\mathcal{S}$, this corresponds exaclty to $\exists\, j\colon\ t \rhd^* \underline{\mu}.(z_j\ \bar{y})$, thus they do not exist terms θ_i.

It is clear that $\mathcal{S}$ is a μ-saturated set. Let $\mathcal{M} = \langle \mathcal{S}; \mathcal{R}\rangle$ and an $\mathcal{M}$-interpretation I such that $I(P) = \mathcal{R}$. By the theorem 4.3.1, $T \in [(\mathcal{R} \rightsquigarrow \mathcal{S}) \rightsquigarrow \mathcal{R}] \rightsquigarrow (\{\bar{y}\} \rightsquigarrow \mathcal{S})$. Let us check that $x \in (\mathcal{R} \rightsquigarrow \mathcal{S}) \rightsquigarrow \mathcal{R}$. For this, we take $\theta \in (\mathcal{R} \rightsquigarrow \mathcal{S})$ and we prove that $(x\ \theta) \in \mathcal{R}$, i.e, $((x\ \theta)\ \bar{y}) \in \mathcal{S}$. By the definition of $\mathcal{S}$, $(z_r\ \bar{y}) \in \mathcal{S}$ for each $r \ge 0$, hence $z_r \in \mathcal{R}$. Therefore $(\theta\ z_r) \in \mathcal{S}$, so we have:

$\forall r'$, $\exists m \ge 1$, $\exists \theta_1, ..., \theta_m$, $\exists j$:

- $(\theta\ z_r) \rhd^* \underline{\mu}.((x\ \theta_1)\ \bar{y})$
- $(\theta_k\ z_{k+r'}) \rhd^* \underline{\mu}.((x\ \theta_{k+1})\ \bar{y})$ for every $1 \le k \le m-1$

- $(\theta_m\ z_{m+r'}) \triangleright^* \underline{\mu}.(z_j\,\bar{y})$.

(when $m = 0$, this gives: $\exists j : (\theta\ z_r) \triangleright^* \underline{\mu}.(z_j\,\bar{y})$, then $((x\ \theta)\ \bar{y}) \triangleright^* \underline{\mu}.((x\ \theta_1')\ \bar{y})$ and $(\theta_1'\ z_r) \triangleright^* \underline{\mu}.(z_j\,\bar{y})$ with $m' = 1$ and $\theta_1' = \theta$. Therefore $((x\ \theta)\ \bar{y}) \in \mathcal{S}$).

More general, since this holds for any r', take $r' = r + 1$, then,
$\exists m \geq 1$, $\exists \theta_1, ..., \theta_m$, $\exists j$:

- $(\theta\ z_r) \triangleright^* \underline{\mu}.((x\ \theta_1)\ \bar{y})$
- $(\theta_k\ z_{k+1+r}) \triangleright^* \underline{\mu}.((x\ \theta_{k+1})\ \bar{y})$ for every $1 \leq k \leq m-1$
- $(\theta_m\ z_{m+1+r}) \triangleright^* \underline{\mu}.(z_j\,\bar{y})$.

Therefore, take also $m' = m+1$, and the terms $\theta_1' = \theta$, $\theta_2' = \theta_1$, ..., $\theta_{m+1}' = \theta_m$, therefore check easily that we have for any fixed r:
$\exists m' \geq 1$, $\exists \theta_1', ..., \theta_{m'}'$, $\exists j$:

- $((x\ \theta)\ \bar{y}) \triangleright^* \underline{\mu}.((x\ \theta_1')\ \bar{y})$
- $(\theta_1'\ z_r) \triangleright^* \underline{\mu}.((x\ \theta_2')\ \bar{y})$
- $(\theta_k'\ z_{k+r}) \triangleright^* \underline{\mu}.((x\ \theta_{k+1}')\ \bar{y})$ for every $1 \leq k \leq m'-1$
- $(\theta_{m'}'\ z_{m'+r}) \triangleright^* \underline{\mu}.(z_j\,\bar{y})$.

Thus $((x\ \theta)\ \bar{y}) \in \mathcal{S}$ which implies that $((T\,x)\,\bar{y}) \in \mathcal{S}$. By the fact that T is a closed term, the λ-variable x and the sequence $\bar{y}$ are differents from each z_i, one can ensure that the assertion $[\exists m = 0,\ \exists j : ((T\,x)\,\bar{y}) \triangleright^* \underline{\mu}.(z_j\,\bar{y})]$ can not hold. Then for $r = 0$, $\exists m \geq 1, \exists \theta_1, ..., \theta_m, \exists j$ such that:

- $((T\,x)\,\bar{y}) \triangleright^* \underline{\mu}.((x\ \theta_1)\,\bar{y})$
- $(\theta_k\ z_k) \triangleright^* \underline{\mu}.((x\,\theta_{k+1})\,\bar{y})$ for every $1 \leq k \leq m-1$
- $(\theta_m\ z_m) \triangleright^* \underline{\mu}.(z_j\,\bar{y})$ for a certain $1 \leq j \leq m$.

■

4.4.3 Terms of type $P \vee \neg P$ "Tertium non datur"

Example 4.4.3 *Let $\mathcal{W} = \mu b.(b\,\omega_1 \mu a.(b\,\omega_2 \lambda y.(a\,y)))$. We have $\vdash \mathcal{W} : P \vee \neg P$.*
Let x_1, x_2, v_1 be λ-variables, u_1, u_2 terms and $\bar{t} \in \mathcal{E}^{<\omega}$. We have:
$(\mathcal{W}\,[x_1.u_1, x_2.u_2]) \triangleright^* \mu b.(b\ u_1\,[x_1 := \theta_1^1])$
$(\theta_1^1\,\bar{t}) \triangleright^* \mu a.(b\,u_2\,[x_2 := \theta_2^2])$
$(\theta_2^2\,v_1) \triangleright^* (a(v_1\,\bar{t}))$
where $\theta_1^1 = \mu a.(b\ (\omega_2 \lambda y.(a\ y)\ [x_1.u_1, x_2.u_2]))$ and $\theta_2^2 = \lambda y.(a\ (y\ \bar{t}))$.

Remark 4.4.3 *The term $\mathcal{W}$ allows to modelizing the* `try...with...` *instruction in the Caml programming language.*

The following theorem gives the behavior of all terms with type $P \vee \neg P$.

Theorem 4.4.3 *Let T be a closed term of type $P \vee \neg P$, then for each λ-variable x_1, x_2, for each term u_1, u_2, for each sequence of sequences of λ-variables $(\bar{y}_i)_{i\in\mathbb{N}^*}$, for each sequence of λ-variables $(z_i)_{i\in\mathbb{N}^*}$ such that: x_1 and x_2 are differents from any z_i and from any element y_i of any sequence $\bar{y}_i$ and such that any variable z_i and any variable y_i of any sequence $\bar{y}_i$ are not free in u_1 or u_2. There exist $m \in \mathbb{N}^*$ and terms $\theta_1^j, ..., \theta_m^j$ $1 \leq j \leq 2$ such that we have:*

- $(T\,[x_1.u_1, x_2.u_2]) \rhd^* \underline{\mu}.u_j[x_j := \theta_1^j]$
- $(\theta_k^1\,\bar{y}_k) \rhd^* \underline{\mu}.u_j[x_j := \theta_{k+1}^j]$ *for all* $1 \leq k \leq m-1$
- $(\theta_k^2\,z_k) \rhd^* \underline{\mu}.u_j[x_j := \theta_{k+1}^j]$ *for all* $1 \leq k \leq m-1$
- $(\theta_m^1 \bar{y}_m) \rhd^* \underline{\mu}.(z_p\,\bar{y}_q)$ *for a certain* $1 \leq p \leq m$ *and a certain* $1 \leq q \leq m$
- $(\theta_m^2\,z_m) \rhd^* \underline{\mu}.(z_p\,\bar{y}_q)$ *for a certain* $1 \leq p \leq m$ *and a certain* $1 \leq q \leq m$

Proof. Let x_1, x_2, u_1, u_2, $(\bar{y}_i)_{i\in\mathbb{N}^*}$ and $(z_i)_{i\in\mathbb{N}^*}$ as in the theorem. Then take $\mathcal{S} = \{t \;/\; \forall r \geq 0, \exists m \geq 1, \exists \theta_1^j, ..., \theta_m^j (1 \leq j \leq 2)\, \exists l, \exists s\colon\ t \rhd^* \underline{\mu}.u_j[x_j := \theta_1^j]$, $(\theta_k^1\,\bar{y}_{k+r}) \rhd^* \underline{\mu}.u_j[x_j := \theta_{k+1}^i]$ for all $1 \leq k \leq m-1$, $(\theta_k^2\,z_{k+r}) \rhd^* \underline{\mu}.u_j[x_j := \theta_{k+1}^j]$ for all $1 \leq k \leq m-1$, $(\theta_m^1\,\bar{y}_{m+r}) \rhd^* \underline{\mu}.(z_l\,\bar{y}_s)$ and $(\theta_m^2\,z_{m+r}) \rhd^* \underline{\mu}.(z_l\,\bar{y}_s)\}$, take also $\mathcal{R} = \{\bar{y}_1, ..., \bar{y}_i, ...\} \rightsquigarrow \mathcal{S}$.

Let us clarify the case $m = 0$: this corresponds to $\exists l, \exists s : t \rhd^* \underline{\mu}.(z_l\,\bar{y}_s)$, they do not exist terms θ_i^j.

By definition, $\mathcal{S}$ is a μ-saturated set. Let $\mathcal{M} = \langle \mathcal{S}; \mathcal{R} \rangle$ and an $\mathcal{M}$-interpretation I such that $I(P) = \mathcal{R}$. By the theorem 4.3.1, $T \in \mathcal{R} \curlyvee [\mathcal{R} \rightsquigarrow \mathcal{S}]$.

We want to prove that $(T\,[x_1.u_1, x_2.u_2]) \in \mathcal{S}$. Then let $\theta^1 \in \mathcal{R}$ and $\theta^2 \in \mathcal{R} \rightsquigarrow \mathcal{S}$, it suffices to prove that $u_j[x_j := \theta^j] \in \mathcal{S}$ for $1 \leq j \leq 2$.

Let us check that $u_1[x_1 := \theta^1] \in \mathcal{S}$: Since $\theta^1 \in \mathcal{R}$, then for any $r \geq 0$, $(\theta^1\,\bar{y}_r) \in \mathcal{S}$. This gives: $\forall r'$, $\exists m \geq 1$, $\exists \theta_1, ..., \theta_m$, $\exists l$, $\exists s$:

- $(\theta^1\,\bar{y}_r) \rhd^* \underline{\mu}.u_j\,[x_j := \theta_1^j]$
- $(\theta_k^1\,\bar{y}_{k+r'}) \rhd^* \underline{\mu}.u_j[x_j := \theta_{k+1}^i]$ for all $1 \leq k \leq m-1$
- $(\theta_k^2\,z_{k+r'}) \rhd^* \underline{\mu}.u_j[x_j := \theta_{k+1}^j]$ for all $1 \leq k \leq m-1$
- $(\theta_m^1\,\bar{y}_{m+r'}) \rhd^* \underline{\mu}.(z_l\,\bar{y}_s)$ and $(\theta_m^2\,z_{m+r'}) \rhd^* \underline{\mu}.(z_l\,\bar{y}_s)$

This is true for any r', hence take $r' = r + 1$, we get:

- $(\theta^1\ \bar{y}_r) \triangleright^* \underline{\mu}.u_j\ [x_j := \theta^j_1]$
- $(\theta^1_k\ \bar{y}_{k+1+r}) \triangleright^* \underline{\mu}.u_j[x_j := \theta^i_{k+1}]$ for all $1 \leq k \leq m-1$
- $(\theta^2_k\ z_{k+1+r}) \triangleright^* \underline{\mu}.u_j[x_j := \theta^j_{k+1}]$ for all $1 \leq k \leq m-1$
- $(\theta^1_m\ \bar{y}_{m+1+r}) \triangleright^* \underline{\mu}.(z_l\ \bar{y}_s)$ and $(\theta^2_m\ z_{m+1+r}) \triangleright^* \underline{\mu}.(z_l\ \bar{y}_s)$

We process as in the proof of the theorem 4.4.2, take: $m' = m + 1$ and $\theta^{1\prime}_1 = \theta^1$, $\theta^{j\prime}_2 = \theta^j_1, ..., \theta^{j}_{m'}{}' = \theta^j_m$, therefore check that:

- $u_1[x_1 := \theta^1] \triangleright^* \underline{\mu}.u_j\ [x_j := \theta^j_1]$
- $(\theta^{1\prime}_1\ \bar{y}_r) \triangleright^* \underline{\mu}.u_j\ [x_j := \theta^j_1]$
- $(\theta^{1\prime}_k\ \bar{y}_{k+r}) \triangleright^* \underline{\mu}.u_j[x_j := \theta^i_{k+1}]$ for all $1 \leq k \leq m-1$
- $(\theta^{2\prime}_k\ z_{k+r}) \triangleright^* \underline{\mu}.u_j[x_j := \theta^j_{k+1}]$ for all $1 \leq k \leq m-1$
- $(\theta^1_{m'}{}'\ \bar{y}_{m'+r}) \triangleright^* \underline{\mu}.(z_l\ \bar{y}_s)$ and $(\theta^2_{m'}{}'\ z_{m'+r}) \triangleright^* \underline{\mu}.(z_l\ \bar{y}_s)$

  since x_1, x_2 are differents from any z_i and any variable y_i of any $\bar{y}_j$, z_l and $\bar{y}_s$ do not provide from x_1 or x_2. For similar reasons since z_l and $\bar{y}_s$ are not free in u_1 or u_2 this implies that $1 \leq l \leq m'$ and $1 \leq s \leq m'$.

By a similar way we check that $u_2[x_2 := \theta^2] \in \mathcal{S}$. ■

Bibliography

[1] Y. Andou. *Church-Rosser property of simple reduction for full first-order classical natural deduction.* Annals of Pure and Applied Logic 119 (2003) 225-237.

[2] R. David and K. Nour. *A short proof of the Strong Normalization of Classical Natural Deduction with Disjunction.* Journal of Symbolic Logic, vol 68, num 4, pp 1277-1288, 2003.

[3] Ph. De Groote. *Strong normalization of classical natural deduction with disjunction.* In 5th International Conference on Typed Lambda Calculi and Applications, TLCA'01. LNCS (2044), pp. 182-196. Springer Verlag, 2001.

[4] G. Gentzen. *Recherches sur la déduction logique.* Press Universitaires de France, 1955. Traduction et commentaires par R. Feys et J. Ladrière.

[5] J.-L. Krivine. *Lambda calcul, types et modèle.* Masson, Paris, 1990.

[6] R. Matthes. *Non-Strictly Positive Fixed Point for Classical Natural Deduction.* APAL, vol 133, pp. 205-230, 2005.

[7] K. Nakazawa. *Confluency and strong normalizability of call-by-value $\lambda\mu$-calculus.* Theoritical Computer Science, vol 290, pp. 429-463. 2003.

[8] K. Nakazawa and M. Tatsuta. *Strong normalization proof with CPS-Translation for the second order classical natural deduction.* The Journal of Symbolic Logic, vol 68, num 3, pp. 851-859. Sept 2003.

[9] K. Nour and K. Saber. *A Semantic of Realizability for the Classical Natural Deduction.* Electronic Notes in Theoretical Computer Science, vol 140, pp.31-39, 2005.

[10] K. Nour and K. Saber. *A semantical proof of strong normalization theorem for full propositional classical natural deduction.* Archive for Mathematical Logic, vol 45, pp.357-364, 2005.

[11] K. Nour and K. Saber. *Some properties of the $\lambda\mu^{\wedge\vee}$-calculus.* Manuscrit, 2007.

[12] K. Nour. *Mixed Logic and Storage Operators.* Archive for Mathematical Logic, vol 39, pp. 261-280, 2000.

[13] M. Parigot. *$\lambda\mu$-calculus: an algorithmic interpretation of classical natural deduction.* Lecture Notes in Artificial Intelligence, 624, Springer Verlag, 1992.

[14] M. Parigot. *Proofs of strong normalization for second order classical natural deduction.* Journal of Symbolic Logic, 62 (4), pp. 1461-1479, 1997.

Chapter 5

A completeness result for a class of types

5.1 Introduction

What came to be called the Curry-Howard correspondence has proven to be a robust technique to study proofs of intuitionistic logic, since it exhibits the structural bond between this logic and the λ-calculus. T. Griffin's works [7] in 1990 allowed to extend this correspondence to the classical logic, which had several consequences. On basis of this new contribution, the $\lambda\mu$-calculus was introduced by M. Parigot [19] and [20]. The $\lambda\mu$-calculus is a natural extension of the λ-calculus which exactly captures the algorithmic content of proofs written in the second order classical natural deduction system. The typed $\lambda\mu$-calculus enjoys all the good properties: the subject reduction, the strong normalization and confluence theorems.

The strong normalization theorem of the second order classical natural deduction [20] is based on a result known as the correctness lemma, which stipulates that each term is in the interpretation of its type. This is also based on the notion of the semantics of realizability. The idea of this semantics consists in associating to each type a set of terms which *realizes* it, this method has been very effective for establishing the strong normalization of type system "à la Tait and Girard". J.- Y. Girard used it to give a proof of the strong normalization of its system $\mathcal{F}$, method known also as the reducibility candidates, later M. Parigot extended this method to the classical case and provided a proof of the strong normalization of the typed $\lambda\mu$-calculus. In a previous work [16], we adapted Parigot's method and established a short semantical proof of the strong normalization of classical natural deduction with disjunction as primitive.

In general all the known semantical proof of strong normalization use a variant of the reducibility candidates based on a correctness lemma, which has been important also for characterizing the operational behavior of some typed terms

and this only through their types, as it was done in J.-L. Krivine's works [12]. This inspired us also to define a general semantics for the classical natural deduction in [15] and gave such characterizations.

The question that we can ask now is: "does the correctness lemma have a converse?". By this we mean: "can we find a class of types for which the converse of the correctness lemma (completeness result) holds?". J.R. Hindley was the first to study the completeness of simple type system property [8], [9] and [10]. R. Labib-sami has established in [14] completeness for a class of types in Girard's system $\mathcal{F}$ known as positive types, and this for a semantics based on the sets stable under the $\beta\eta$-equivalence. S. Farkh and K. Nour revisited this result, and generalized it, in fact they proved a refined result by indicating that weak-head-expansion is sufficient [4]. In [5], they established an other completeness result for a class of types in Krivine's system $\mathcal{AF}2$. Recently, F. Kamareddine and K. Nour improved the result of Hindley, to a system with an intersection type. Independently, T. Coquand established in [1] by methods using Kripke's models, the completeness for the simply typed λ-calculus.

In the present work we dealt with this problem and we prove the completeness for the simply typed $\lambda\mu$-calculus. The semantics that we define here is not completely different from that of [16] and [15], nevertheless we add a slight but an indispensable modification to the notion of the μ-saturation. In fact, to show that each element $\mathcal{R}$ of the model can be written in the form $\mathcal{R}^{\perp} \rightarrow \mathcal{S}$ (where $\mathcal{S}$ replaced the set $\mathcal{N}$ of strongly normalizable term and satisfied the saturation property), and under the constraint of the definition of these $\mathcal{R}$ imposed by the completeness side, we are compelled to bring this subtle modification.

Moreover the strong normalization, the semantics of the adequation lemma allows to give short proofs of theorems describing the computational behavior of closed typed terms through their types. Nevertheless, the proofs suppose well known the behaviors, therefore the models are exactly built to satisfy the required properties. This is not the case of the syntactical proofs, where we guess the behavior through the type, which is rather constructive, but of an other share these proofs are more complicated than the semantical ones. In what follows, we give at each time, both of semantics and syntactical proofs.

This chapter is organized as follows. Section 2 is an introduction to the simply typed $\lambda\mu$-calculus. In section 3, we define the semantics and prove a correctness lemma. In Section 4, we give characterizations of some closed typed terms. Finally, Section 5 is devoted to the completeness result.

5.2 The simply typed $\lambda\mu$-calculus

Definition 5.2.1 *1. Let $\mathcal{X}$ and $\mathcal{A}$ be two infinite sets of disjoint alphabets for distinguiching λ-variables and μ-variables. The $\lambda\mu$-terms are given by the following grammar:*

$$\mathcal{T} := \mathcal{X} \mid \lambda\mathcal{X}.\mathcal{T} \mid (\mathcal{T}\ \mathcal{T}) \mid\ \mu\mathcal{A}.\mathcal{T} \mid (\mathcal{A}\ \mathcal{T})$$

2. *Types are formulas of the propositional logic built from the infinite set of propositional variables* $\mathcal{P} = \{X, Y, Z, ...\}$ *and a constant of type* $\bot$*, using the connector* $\rightarrow$*.*

3. *As usual we denote by* $\neg A$ *the formula* $A \rightarrow \bot$*. Let* $A_1, A_2, ..., A_n, A$ *be types, we denote the type* $A_1 \rightarrow (A_2 \rightarrow (... \rightarrow (A_n \rightarrow A)...))$ *by* $A_1, A_2, ..., A_n \rightarrow A$*.*

4. *Proofs are presented in natural deduction system with two conclusions, such that formulas in the left of* $\vdash$ *are indexed by* λ*-variables and those in right of* $\vdash$ *are indexed by* μ*-variables, except one which is indexed by a term.*

5. *Let* t *be a* $\lambda\mu$*-term,* A *a type,* $\Gamma = \{x_i : A_i\}_{1 \leq i \leq n}$ *and* $\Delta = \{a_j : B_j\}_{1 \leq j \leq m}$*, using the following rules, we will define "*t *typed with type* A *in the contexts* Γ *and* Δ*" and we denote it* $\Gamma \vdash t : A\ ; \Delta$*.*

$$\frac{}{\Gamma \vdash x_i : A_i\ ;\ \Delta} ax \quad \text{for } 1 \leq i \leq n.$$

$$\frac{\Gamma, x : A \vdash t : B; \Delta}{\Gamma \vdash \lambda x.t : A \rightarrow B; \Delta} \rightarrow_i \qquad \frac{\Gamma \vdash u : A \rightarrow B; \Delta \quad \Gamma \vdash v : A; \Delta}{\Gamma \vdash (u\ v) : B; \Delta} \rightarrow_e$$

$$\frac{\Gamma \vdash t : \bot; \Delta, a : A}{\Gamma \vdash \mu a.t : A; \Delta} \mu \qquad \frac{\Gamma \vdash t : A; \Delta, a : A}{\Gamma \vdash (a\ t) : \bot; \Delta, a : A} \bot$$

We denote this typed system by S_μ*.*

6. *The basic reduction rules are* β *and* μ *reductions.*

   - $(\lambda x.u\ v) \triangleright_\beta u[x := v]$
   - $(\mu a.u\ v) \triangleright_\mu \mu a.u[a :=^* v]$
     where $u[a :=^* v]$ *is obtained from* u *by replacing inductively each subterm in the form* $(a\ w)$ *in* u *by* $(a\ (w\ v))$*.*

7. *We denote* $t \triangleright t'$ *if* t *is reduced to* t' *by one of the rules given above. As usual* $\triangleright^*$ *denotes the reflexive transitive closure of* $\triangleright$*, and* $\simeq$ *the equivalence relation induced by* $\triangleright^*$*.*

We have the following results (for more lecture, see [20]).

Theorem 5.2.1 (Confluence result) *If* $t \triangleright^* t_1$ *and* $t \triangleright^* t_2$*, then there exists* t_3 *such that* $t_1 \triangleright^* t_3$ *and* $t_2 \triangleright^* t_3$

Theorem 5.2.2 (Subject reduction) *If $\Gamma \vdash t : A; \Delta$ and $t \rhd^* t'$ then $\Gamma \vdash t' : A; \Delta$.*

Theorem 5.2.3 (Strong normalization) *If $\Gamma \vdash t : A; \Delta$, then t is strongly normalizable.*

Definition 5.2.2 *1. Let t be a term and $\bar{v}$ a finite sequence of terms (the empty sequence is denoted by $\emptyset$), then, the term $t\bar{v}$ is defined by $(t\ \emptyset) = t$ and $(t\ u\bar{u}) = ((t\ u)\ \bar{u})$.*

2. Let $t, u_1, ..., u_n$ be terms and $\bar{v}_1, ..., \bar{v}_m$ finite sequences of terms, then $t[(x_i := u_i)_{1\leq i\leq n}; (a_j :=^* \bar{v}_j)_{1\leq j\leq m}]$ is obtained from the term t by replacing inductively each x_i by u_i and each subterm in the form $(a_j\ u)$ in t by $(a_j\ (u\,\bar{v}_j))$.

Lemma 5.2.1 *Let $\sigma = [(x_i := u_i)_{1\leq i\leq n}; (a_j :=^* \bar{v}_j)_{1\leq j\leq m}]$ and t, t' two terms such that $t \rhd^* t'$, then, $t\sigma \rhd^* t'\sigma$.*

Proof. By induction on t. ■

5.3 The semantics of S_μ

Definition 5.3.1 *1. Let $\mathcal{S}$ be a set of terms, we say that $\mathcal{S}$ is a saturated set iff for each terms u and v, if $u \in \mathcal{S}$ and $v \rhd^* u$, then, $v \in \mathcal{S}$.*

2. Let us take a saturated set of terms $\mathcal{S}$ and a set $\mathcal{C}$ of an infinite classical variables (μ-variables). We said that $\mathcal{S}$ is $\mathcal{C}$-saturated iff for each $t \in \mathcal{S}$ and for each $a \in \mathcal{C}$, $\mu a.t \in \mathcal{S}$ and $(a\ t) \in \mathcal{S}$.

3. Consider two sets of terms $\mathcal{K}$ and $\mathcal{L}$, we define a new set of terms: $\mathcal{K} \rightsquigarrow \mathcal{L} = \{t\ /\ (t\ u) \in \mathcal{L}$, for each $u \in \mathcal{K}\}$. It is clear that when $\mathcal{L}$ is a saturated set, then $\mathcal{K} \rightsquigarrow \mathcal{L}$ is also a saturated one.

4. We denote $\mathcal{T} \cup \mathcal{A}$ by $\mathcal{T}'$ and $\mathcal{T}'^{<\omega}$ the set of finite sequences of $\mathcal{T}'$. Let t be a term and $\pi \in \mathcal{T}'^{<\omega}$, then the term $(t\ \pi)$ is defined by $(t\ \emptyset) = t$, $(t\ \pi) = ((t\ u)\ \pi')$ if $\pi = u\pi'$ and $(t\ \pi) = ((a\ t)\ \pi')$ if $\pi = a\pi'$.

5. Let $\mathcal{S}$ be a set of terms and $\mathcal{X} \subseteq \mathcal{T}'^{<\omega}$, then we define $\mathcal{X} \rightsquigarrow \mathcal{S} = \{t\ /\ (t\ \pi) \in \mathcal{S}$, for each $\pi \in \mathcal{X}\}$.

Remark 5.3.1 *The fact that the application $(a\,t)$ is denoted by $(t\,a)$ is not something new, it is already present in Saurin's work [22]. Except that for us, it is a simple notation in order to uniformize the definition of the application. But for Saurin, it is crucial to obtain the theorem of separation in the $\lambda\mu$-calculus.*

Definition 5.3.2 *Let $\mathcal{S}$ be a $\mathcal{C}$-saturated set and $\{\mathcal{R}_i\}_{i\in I}$ subsets of terms such that $\mathcal{R}_i = \mathcal{X}_{\mathcal{R}_i} \rightsquigarrow \mathcal{S}$ for some $\mathcal{X}_{\mathcal{R}_i} \subseteq \mathcal{T}'^{<\omega}$. A model $\mathcal{M}=\langle \mathcal{C}, \mathcal{S}, \{\mathcal{R}_i\}_{i\in I}\rangle$ is the smallest set of subsets of terms containing $\mathcal{S}$ and $\mathcal{R}_i$, and closed under the constructor $\rightsquigarrow$.*

Lemma 5.3.1 *Let $\mathcal{M} = \langle \mathcal{C}, \mathcal{S}, \{\mathcal{R}_i\}_{i\in I}\rangle$ be a model and $\mathcal{G} \in \mathcal{M}$. There exists a set $\mathcal{X}_\mathcal{G} \subseteq \mathcal{T}'^{<\omega}$ such that $\mathcal{G} = \mathcal{X}_\mathcal{G} \rightsquigarrow \mathcal{S}$.*

Proof. By induction on $\mathcal{G}$.

- If $\mathcal{G} = \mathcal{S}$, take $\mathcal{X}_\mathcal{G} = \{\phi\}$.
- If $\mathcal{G} = \mathcal{R}_i$, take $\mathcal{X}_\mathcal{G} = \mathcal{X}_{\mathcal{R}_i}$.
- If $\mathcal{G} = \mathcal{G}_1 \rightsquigarrow \mathcal{G}_2$, then, by the induction hypothesis, $\mathcal{G}_2 = \mathcal{X}_{\mathcal{G}_2} \rightsquigarrow \mathcal{S}$ where $\mathcal{X}_{\mathcal{G}_2} \subseteq \mathcal{T}'^{<\omega}$, and take $\mathcal{X}_\mathcal{G} = \{u\bar{v} \ / \ u \in \mathcal{G}_1 \text{ and } \bar{v} \in \mathcal{X}_{\mathcal{G}_2}\}$.

■

Definition 5.3.3 *Let $\mathcal{M} = \langle \mathcal{C}, \mathcal{S}, \{\mathcal{R}_i\}_{i\in I}\rangle$ be a model and $\mathcal{G} \in \mathcal{M}$. We define the set $\mathcal{G}^\perp = \cup\{\mathcal{X}_\mathcal{G} \ / \ \mathcal{G} = \mathcal{X}_\mathcal{G} \rightsquigarrow \mathcal{S}\}$.*

Lemma 5.3.2 *Let $\mathcal{M} = \langle \mathcal{C}, \mathcal{S}, \{\mathcal{R}_i\}_{i\in I}\rangle$ be a model and $\mathcal{G} \in \mathcal{M}$. We have $\mathcal{G} = \mathcal{G}^\perp \rightsquigarrow \mathcal{S}$.*

Proof. This comes from the fact that: if for every $j \in J$, $\mathcal{G} = \mathcal{X}_{\mathcal{G}_j} \rightsquigarrow \mathcal{S}$, then, $\mathcal{G} = (\cup_{j\in J}\mathcal{X}_{\mathcal{G}_j}) \rightsquigarrow \mathcal{S}$. ■

Definition 5.3.4

1. *Let $\mathcal{M} = \langle \mathcal{C}, \mathcal{S}, \{\mathcal{R}_i\}_{i\in I}\rangle$ be a model. An $\mathcal{M}$-interpretation $\mathcal{I}$ is an application $X \mapsto \mathcal{I}(X)$ from the set of propositional variables $\mathcal{P}$ in $\mathcal{M}$ which we extend for any formula as follows:*
   - $\mathcal{I}(\perp) = \mathcal{S}$
   - $\mathcal{I}(A \to B) = \mathcal{I}(A) \rightsquigarrow \mathcal{I}(B)$.
2. *For any type A, we denote $|A|_\mathcal{M} = \bigcap\{\mathcal{I}(A) \ / \ \mathcal{I}$ an $\mathcal{M}$-interpretation$\}$.*
3. *For any type A, $|A| = \bigcap\{|A|_\mathcal{M} \ / \ \mathcal{M}$ a model$\}$.*
4. *Let u, v be two terms. The expression $v \approx_\mathcal{C} u$ means that v is obtained from u by replacing the free classical variables of u by some others in $\mathcal{C}$, i.e. if we denote u by $u[a_1, ..., a_n]$ where the a_i are the free classical variables of u, then v will be $u[a_1 := b_1, ..., a_n := b_n]$ where $b_i \neq b_j$ for $(i \neq j)$ and $b_i \in \mathcal{C}$.*

Lemma 5.3.3 (Adequation lemma) *Let $\Gamma = \{x_i : A_i\}_{1\leq i\leq n}$, $\Delta = \{a_j : B_j\}_{1\leq j\leq m}$, $\mathcal{M} = \langle \mathcal{C}, \mathcal{S}, \{\mathcal{R}_i\}_{i\in I}\rangle$ a model, $\mathcal{I}$ an $\mathcal{M}$-interpretation, $u_i \in \mathcal{I}(A_i)$, $\bar{v_j} \in (\mathcal{I}(B_j))^\perp$, $\sigma = [(x_i := u_i)_{1\leq i\leq n}; (a_j :=^* \bar{v}_j)_{1\leq j\leq m}]$, and u, v two terms such that $v \approx_\mathcal{C} u$. If $\Gamma \vdash u : A \ ; \Delta$, then, $v\sigma \in \mathcal{I}(A)$.*

Proof. By induction on the derivation, we consider the last used rule.

ax: In this case $u = x_i = v$ and $A = A_i$, then $v\sigma = u_i \in \mathcal{I}(A)$.

$\rightarrow_i$: In this case $u = \lambda x.u_1$ and $A = B \rightarrow C$ such that $\Gamma, x : B \vdash u_1 : C\ ;\Delta$. Then $v = \lambda x.v_1$ and $v_1 \approx_{\mathcal{C}} u_1$. By the induction hypothesis, $v_1\sigma[x := w] \in \mathcal{I}(C)$ for each $w \in \mathcal{I}(B)$, hence $(\lambda x.v_1\sigma\ w) \in \mathcal{I}(C)$, therefore $\lambda x.v_1\sigma \in \mathcal{I}(B) \rightsquigarrow \mathcal{I}(C)$. Finally $v\sigma \in \mathcal{I}(A)$.

$\rightarrow_e$: In this case $u = (u_1\ u_2)$, $\Gamma \vdash u_1 : B \rightarrow A\ ;\Delta$ and $\Gamma \vdash u_2 : B\ ;\Delta$. We have also $v = (v_1\ v_2)$ where $v_1 \approx_{\mathcal{C}} u_1$ and $v_2 \approx_{\mathcal{C}} v_1$. By the induction hypothesis, $v_1\sigma \in \mathcal{I}(B) \rightsquigarrow \mathcal{I}(A)$ and $v_2\sigma \in \mathcal{I}(B)$, therefore $(v_1\sigma\ v_2\sigma) \in \mathcal{I}(A)$, this implies that $v\sigma \in \mathcal{I}(A)$.

μ: In this case $u = \mu a.u_1$, then $v = \mu b.v_1$ where $v_1 \approx_{\mathcal{C}} u_1$ and b is a new variable which belongs to $\mathcal{C}$ and not free in u_1 (there is always such variable because $\mathcal{C}$ is infinite). Let $\bar{v} \in (\mathcal{I}(A))^{\perp}$. By the induction hypothesis, $v_1\sigma[b :=^* \bar{v}] \in \mathcal{S}$, and, by the definition of $\mathcal{S}$, we have, $\mu b.v_1\sigma[b :=^* \bar{v}] \in \mathcal{S}$, since $(\mu b.v_1\sigma\ \bar{v}) \triangleright^* \mu b.v_1\sigma[b :=^* \bar{v}]$, then, $\mu b.v_1\sigma \in \mathcal{I}(A)$, i.e, $v\sigma \in \mathcal{I}(A)$.

$\perp$: In this case $u = (a\ u_1)$, then, $v = (b\ v_1)$ where $v_1 \approx_{\mathcal{C}} u_1$ such that the free variable a was repalced by b in u_1 and $b \notin Fv(u_1)$ is new variable which belongs to $\mathcal{C}$. By the induction hypothesis, $v_1\sigma[b :=^* \bar{v}] \in \mathcal{I}(A)$ where $\bar{v} \in (\mathcal{I}(A))^{\perp}$, hence $(v_1\sigma[b :=^* \bar{v}]\ \bar{v}) \in \mathcal{S}$. Therefore, by the definition of $\mathcal{S}$, $(b\ (v_1\sigma[b :=^* \bar{v}]\ \bar{v})) \in \mathcal{S}$, and finally $v\sigma \in \mathcal{S}$.

■

Corollary 5.3.1 *Let A be a type and t a closed term. If $\vdash t : A$, then, $t \in |A|$.*

Proof. Let $\mathcal{M}$ be a model and $\mathcal{I}$ an $\mathcal{M}$-interpretation. Since $\vdash t : A$, then, by the adequation lemma, $t \in \mathcal{I}(A)$. This is true for any $\mathcal{M}$ model and for any $\mathcal{M}$-interpretation I, therefore $t \in |A|$. ■

5.4 Characterization of some typed terms

We start this section by adding to our system new propositional constants to obtain a new parametrized typed system. This will be useful for the proof of the lemma 5.4.3, which allows us to provide the syntactical proofs concerning the characterization of some typed terms.

5.4.1 The system $S_\mu{}^{\bar{O}}$

Definition 5.4.1 *Let $\bar{O} = O_1, ..., O_n$ be a sequence of new propositional constants.*

1. *We said that $\bar{O}$ is different from $\bot$ iff each O_i is different from $\bot$.*
2. *A type A is said to be ending by $\bar{O}$ iff A is obtained by the following rules:*
   - *Each O_i ends by $\bar{O}$.*
   - *If B ends by $\bar{O}$, then, $A \to B$ ends by $\bar{O}$.*
3. *The typed system $S_\mu{}^{\bar{O}}$ is the system S_μ at which we add the following conditions:*
   - *The rules ax is replaced by*

     $$\frac{}{\Gamma \vdash_{\bar{O}} x_i : A_i \ ; \Delta} ax$$

     where Δ does not contain declarations of the form $a : C$ such that C ends by $\bar{O}$.
   - *The rules $\to_e$ is replaced by*

     $$\frac{\Gamma \vdash_{\bar{O}} u : A \to B; \Delta \quad \Gamma \vdash_{\bar{O}} v : A; \Delta}{\Gamma \vdash_{\bar{O}} (u\ v) : B; \Delta} \to_e$$

     where B is not ending by $\bar{O}$.

Remark 5.4.1 *It is obvious that $S_\mu{}^{\bar{O}}$ can be seen as a subsystem of S_μ where the syntax of formulas is extended by the new constants $\bar{O}$, therefore in the remainder of this work we consider that, any typed term in the system $S_\mu{}^{\bar{O}}$ is strongly normalizable.*

Lemma 5.4.1 *If $\Gamma \vdash t : A \ ; \Delta$ then $\Gamma \vdash_{\bar{O}} t : A[X := F] \ ; \Delta$ where F does not end by $\bar{O}$.*

Proof. By induction on the derivation. ■

The following lemma stipulates that the new system $S_\mu{}^{\bar{O}}$ is closed under reduction (subject reduction).

Lemma 5.4.2 *If $\Gamma \vdash_{\bar{O}} t : A \ ; \Delta$ and $t \rhd^* t'$, then $\Gamma \vdash_{\bar{O}} t' : A \ ; \Delta$*

Proof. By induction on the length of the reduction $t \rhd^* t'$. It suffices to check this result for $t \rhd_\beta t'$ and $t \rhd_\mu t'$. We process by induction on t. ■

Lemma 5.4.3 *Let* $\Gamma = \{x_i : A_i\}_{1\leq i\leq n}$, $\Delta = \{a_j : B_j\}_{1\leq j\leq m}$ $\bar{O} = O_1, ..., O_k$ *different from* $\perp$ *and* $1 \leq l \leq k$. *If* $\Gamma \vdash_{\bar{O}} t : O_l \ ; \Delta$, *then,* $t = x_j$ *for certain* $1 \leq j \leq n$ *and* $A_j = O_l$.

Proof. By induction on the derivation.

ax: Then, $\Gamma \vdash x_j : A_j; \ \Delta$, hence $t = x_j$ and $O_l = A_j$

$\rightarrow_i$: A contradiction because this implies that O_l is not atomic.

$\rightarrow_e$: This implies that $t = (u\ v)$, then, $\Gamma \vdash u : A \rightarrow O_l; \Delta$, therefore this gives a contradiction with the restriction on the rule $\rightarrow_e$ since O_l ends by $\bar{O}$.

μ: Then, $t = \mu a.t_1$ and $\Gamma \vdash t_1 :\perp; \ \Delta', a : O_l$, where $\Delta = \Delta' \cup \{a : O_l\}$, therefore this gives a contradiction with the fact that Δ does not contain declarations in the form $a_j : O_j$.

$\perp$: A contradiction because O_l is different from $\perp$.

■

We give now some applications of the adequation lemma.

Definition 5.4.2 *Let* t *be a term. We denote* M_t *the smallest set containing* t *such that: if* $u \in M_t$ *and* $a \in \mathcal{A}$, *then* $\mu a.u \in M_t$ *and* $(a\ u) \in M_t$. *Each element of* M_t *is denoted* $\underline{\mu}.t$. *For example, the term* $\mu a.\mu b.(a\ (b\ (\mu c.(a\ \mu d.t))))$ *is denoted by* $\underline{\mu}.t$.

5.4.2 Terms of type $\perp\rightarrow X$

Example 5.4.1 *Let* $T_1 = \lambda z.\mu a.z$ *and* $T_2 = \lambda z.\mu b.(b\,\mu a.z)$, *we have* $\vdash T_i :\perp\rightarrow X$.

Let then, x *be* λ*-variable and* $\bar{y}$ *a finite sequence of* λ*-variables, we have:*

- $(T_1\ x)\ \bar{y} \rhd^* \mu a.x$
- $(T_2\ x)\ \bar{y} \rhd^* \mu b.(b\ \mu a.x)$

The operational behavior of closed terms with the type $\perp\rightarrow X$ is given in the following theorem.

Theorem 5.4.1 *Let* T *be a closed term of type* $\perp\rightarrow X$, *then, for each* λ*-variable* x *and for each finite sequence of* λ*-variables* $\bar{y}$, $(T\,x)\ \bar{y} \rhd^* \underline{\mu}.x$

Proof.

Semantical proof:
Let x be a λ-variable and $\bar{y}$ a finite sequence of λ-variables. Let $\mathcal{C} = \mathcal{A}$ and take $\mathcal{S} = \{t \,/\, t \rhd^* \underline{\mu}.x\}$ and $\mathcal{R} = \{\bar{y}\} \rightsquigarrow \mathcal{S}$. It is clear that $\mathcal{S}$ is $\mathcal{C}$-saturated set and $x \in \mathcal{S}$. So let $\mathcal{M} =< \mathcal{C}, \mathcal{S}; \mathcal{R} >$ and take $\mathcal{I}$ the interpretation which at X associates $\mathcal{I}(X) = \mathcal{R}$. By the adequation lemma, $T \in \mathcal{I}(\bot \rightarrow X)$, then, $T \in \mathcal{S} \rightsquigarrow \mathcal{R}$, i.e, $T \in \mathcal{S} \rightsquigarrow (\{\bar{y}\} \rightsquigarrow \mathcal{S})$, therefore $(T\ x) \in \{\bar{y}\} \rightsquigarrow \mathcal{S}$, and $(T\ x)\ \bar{y} \in \mathcal{S}$. Finally $(T\,x)\ \bar{y} \rhd^* \underline{\mu}.x$.

Syntactical proof:
We can give also a syntactical proof of this result. Let $\bar{O} = O_1, ..., O_n$ be a sequence of new constants different from $\bot$, $A = O_1, ..., O_n \rightarrow \bot$ and $\bar{y} = y_1 ... y_n$ a sequence of λ-variables. By the lemma 5.4.1, $\vdash_{\bar{O}} T : \bot \rightarrow A,$, then, $x : \bot, (y_i : O_i)_{1 \leq i \leq n} \vdash_{\bar{O}} (T\ x)\bar{y} : \bot$, hence $(T\ x)\bar{y} \rhd^* \tau$. It suffices to prove that, if τ is a normal term and $x : \bot, (y_i : O_i)_{1 \leq i \leq n} \vdash_{\bar{O}} \tau : \bot\ ; (b_j : \bot)_{1 \leq j \leq m}$, then $\tau = \underline{\mu}.x$. This can be proved easily by induction on τ. ■

Corollary 5.4.1 *Let T be a closed term of type $(\bot \rightarrow X)$, then, for each term u and for each $\bar{v} \in \mathcal{T}^{<\omega}$, $(T\,u)\,\bar{v} \rhd^* \underline{\mu}.u$*

Proof. Immediately from the previous theorem and the lemma 5.2.1. ■

Remark 5.4.2 *Let $\vdash T : \bot \rightarrow X$, the term $(T\ u)$ modelizes an instruction like* `exit`(u) (`exit` *is to be understood as in the C programming language). In the reduction of a term, if the sub-term $(T\ u)$ appears in head position (the term has the form $((T\ u)\ \bar{v})$), then after some reductions, we obtain u as result.*

5.4.3 Terms of type $(\neg X \rightarrow X) \rightarrow X$

Example 5.4.2 *Let the terms $T_1 = \lambda z.\mu a.a(z\ \lambda y.(a\ y))$*
and $T_2 = \lambda z.\mu a.(a\ (z\ (\lambda s.a(z\ \lambda y.(a\,s)))))$,
we have $\vdash T_i : (\neg X \rightarrow X) \rightarrow X$.

Let x, z_1, z_2 be λ-variables and $\bar{y}$ a finite sequence of λ-variables, we have:

- $(T_1\ x)\ \bar{y} \rhd^* \mu a.a\ ((x\ \theta_1)\ \bar{y})$ *and* $(\theta_1\ z_1) \rhd^* a(z_1\ \bar{y})$.
- $(T_2\ x)\ \bar{y} \rhd^* \mu a.((a((x\ \theta_1)\ \bar{y}))\ \bar{y})$, $(\theta_1\ z_1) \rhd^* (a\ ((x\ \theta_2)\ \bar{y}))$, *and* $(\theta_2\ z_2) \rhd^* (a\ (z_1\ \bar{y}))$.

The following theorem describes the computational behavior of closed terms with type $(\neg X \rightarrow X) \rightarrow X$.

Theorem 5.4.2 *Let T be a closed term of type $(\neg X \to X) \to X$, then, for each λ-variable x, for each finite sequence of λ-variables $\bar{y}$ and for each sequence of λ-variables $(z_i)_{i\in\mathbb{N}^*}$ such that: x, y_j are differents from any z_i. There exist $m \in \mathbb{N}^*$ and terms $\theta_1, ..., \theta_m$, such that we have:*

- $(T\ x)\bar{y} \rhd^* \underline{\mu}.(x\ \theta_1)\ \bar{y}$
- $(\theta_k\ z_k) \rhd^* \underline{\mu}.(x\ \theta_{k+1})\ \bar{y}$ *for all* $1 \leq k \leq m-1$
- $(\theta_m\ z_m) \rhd^* \underline{\mu}.(z_l\ \bar{y})$ *for a certain* $1 \leq l \leq m$

Proof.

Semantical proof:
Let x be a λ-variable, $\bar{y}$ a finite sequence of λ-variables and $(z_i)_{i\in\mathbb{N}^*}$ a sequence of λ-variables as in the theorem. Take $\mathcal{S} = \{t \ / \ \forall r \geq 0,\ \exists m \geq 0,\ \exists \theta_1, ..., \theta_m,\ \exists j\colon\ t \rhd^* \underline{\mu}.((x\ \theta_1)\,\bar{y}),\ (\theta_k\ z_{k+r}) \rhd^* \underline{\mu}.((x\,\theta_{k+1})\,\bar{y})$ for every $1 \leq k \leq m-1$ and $(\theta_m\ z_{m+r}) \rhd^* \underline{\mu}.(z_j\,\bar{y})\}$ and $\mathcal{R} = \{\bar{y}\} \rightsquigarrow \mathcal{S}$.

It is important to clarify the case $m = 0$ in the definition of $\mathcal{S}$, this corresponds exaclty to $\exists j\colon\ t \rhd^* \underline{\mu}.(z_j\ \bar{y})$, thus they do not exist terms θ_i.

It is clear that $\mathcal{S}$ is a μ-saturated set. Let $\mathcal{M} = \langle \mathcal{S}; \mathcal{R}\rangle$ and an $\mathcal{M}$-interpretation I such that $I(X) = \mathcal{R}$. By the theorem 5.3.1, $T \in [(\mathcal{R} \rightsquigarrow \mathcal{S}) \rightsquigarrow \mathcal{R}] \rightsquigarrow (\{\bar{y}\} \rightsquigarrow \mathcal{S})$. Let us check that $x \in (\mathcal{R} \rightsquigarrow \mathcal{S}) \rightsquigarrow \mathcal{R}$. For this, we take $\theta \in (\mathcal{R} \rightsquigarrow \mathcal{S})$ and we prove that $(x\ \theta) \in \mathcal{R}$, i.e, $((x\ \theta)\ \bar{y}) \in \mathcal{S}$. By the definition of $\mathcal{S}$, $(z_r\ \bar{y}) \in \mathcal{S}$ for each $r \geq 0$, hence $z_r \in \mathcal{R}$. Therefore $(\theta\ z_r) \in \mathcal{S}$, so we have:

$\forall r'$, $\exists m \geq 1$, $\exists \theta_1, ..., \theta_m$, $\exists j$:

- $(\theta\ z_r) \rhd^* \underline{\mu}.((x\ \theta_1)\ \bar{y})$
- $(\theta_k\ z_{k+r'}) \rhd^* \underline{\mu}.((x\ \theta_{k+1})\ \bar{y})$ for every $1 \leq k \leq m-1$
- $(\theta_m\ z_{m+r'}) \rhd^* \underline{\mu}.(z_j\,\bar{y})$.

(when $m = 0$, this gives: $\exists j : (\theta\ z_r) \rhd^* \underline{\mu}.(z_j\,\bar{y})$, then $((x\ \theta)\ \bar{y}) \rhd^* \underline{\mu}.((x\ \theta_1')\ \bar{y})$ and $(\theta_1'\ z_r) \rhd^* \underline{\mu}.(z_j\,\bar{y})$ with $m' = 1$ and $\theta_1' = \theta$. Therefore $((x\ \theta)\ \bar{y}) \in \mathcal{S}$).

More general, since this holds for any r', take $r' = r + 1$, then
$\exists m \geq 1$, $\exists \theta_1, ..., \theta_m$, $\exists j$:

- $(\theta\ z_r) \rhd^* \underline{\mu}.((x\ \theta_1)\ \bar{y})$
- $(\theta_k\ z_{k+1+r}) \rhd^* \underline{\mu}.((x\ \theta_{k+1})\ \bar{y})$ for every $1 \leq k \leq m-1$

- $(\theta_m\ z_{m+1+r}) \rhd^* \underline{\mu}.(z_j\,\bar{y})$.

Therefore take also $m' = m+1$, and the terms $\theta'_1 = \theta$, $\theta'_2 = \theta_1$, ..., $\theta'_{m+1} = \theta_m$, hence check easily that we have for any fixed r:
$\exists m' \geq 1$, $\exists \theta'_1, ..., \theta'_{m'}$, $\exists j$:

- $((x\ \theta)\ \bar{y}) \rhd^* \underline{\mu}.((x\ \theta'_1)\ \bar{y})$
- $(\theta'_1\ z_r) \rhd^* \underline{\mu}.((x\ \theta'_2)\ \bar{y})$
- $(\theta'_k\ z_{k+r}) \rhd^* \underline{\mu}.((x\ \theta'_{k+1})\ \bar{y})$ for every $1 \leq k \leq m'-1$
- $(\theta'_{m'}\ z_{m'+r}) \rhd^* \underline{\mu}.(z_j\,\bar{y})$.

Thus $((x\ \theta)\ \bar{y}) \in \mathcal{S}$ which implies that $((T\,x)\,\bar{y}) \in \mathcal{S}$. By the fact that T is a closed term, the λ-variable x and the sequence $\bar{y}$ are differents from each z_i, one can ensure that the assertion $[\exists m = 0,\ \exists j : ((T\,x)\,\bar{y}) \rhd^* \underline{\mu}.(z_j\,\bar{y})]$ can not hold. Then for $r = 0$, $\exists m \geq 1, \exists \theta_1, ..., \theta_m, \exists j$ such that:

- $((T\,x)\,\bar{y}) \rhd^* \underline{\mu}.((x\ \theta_1)\,\bar{y})$
- $(\theta_k\ z_k) \rhd^* \underline{\mu}.((x\,\theta_{k+1})\,\bar{y})$ for every $1 \leq k \leq m-1$
- $(\theta_m\ z_m) \rhd^* \underline{\mu}.(z_j\,\bar{y})$ for a certain $1 \leq j \leq m$.

Syntactical proof:

We give now a syntactical proof of this result. Let $\bar{O} = O_1, ..., O_n$ be new constants different from $\bot$, $A = O_1, ..., O_n \to \bot$ and $\bar{y} = y_1...y_n$ a sequence of variables. By the lemma 5.4.1 $\vdash_{\bar{O}} T : (\neg A \to A) \to A$, then, $x : \neg A \to A, (y_i : O_i)_{1\leq i\leq n} \vdash_{\bar{O}} (T\ x)\bar{y} : \bot$. Therefore, $(T\ x)\bar{y} \rhd^* \tau$, where τ is a normal term and $x : \neg A \to A, (y_i : O_i)_{1\leq i\leq n} \vdash_{\bar{O}} \tau : \bot$.

Following the form of τ we have only one case to examine, the others give always contradictions. This case is $\tau = \underline{\mu}.(x\ U_1)\ t_1...t_n$ where $U_1, t_1, ..., t_n$ are normal terms, $x : \neg A \to A, (y_i : O_i)_{1\leq i\leq n} \vdash_{\bar{O}} U_1 : \neg A\ ; (b_j : \bot)_{1\leq i\leq m}$ and for all $1 \leq k \leq n$, $x : \neg A \to A, (y_i : O_i)_{1\leq i\leq n} \vdash_{\bar{O}} t_k : O_k\ ; (b_j : \bot)_{1\leq j\leq m}$. We deduce, by lemma 5.4.3, that, for all $1 \leq k \leq n$, $t_k = y_k$.

We prove, by induction and using the lemma 5.4.3, that if $x : \neg A \to A, (y_i : O_i)_{1\leq i\leq n}, (z_k : A)_{1\leq k\leq i-1} \vdash_{\bar{O}} U_i : \neg A\ ; (b_j : \bot)_{1\leq j\leq m}$, then

$$\begin{cases} (\mathrm{U}_i\ z_i) \rhd^* \underline{\mu}.(x\ U_{i+1})\bar{y} \text{ and } x : \neg A \to A, (y_i : O_i)_{1\leq i\leq n}, (z_k : A)_{1\leq k\leq i} \vdash_{\bar{O}} U_{i+1} : \\ \neg A\ ; (b_j : \bot)_{1\leq j\leq m} \\ \textit{or} \\ \exists j : (1 \leq j \leq i), \text{ such that: } (U_i\ z_i) \rhd^* \underline{\mu}.z_j\bar{y} \end{cases}$$

The sequence $(U_i)_{i\geq 1}$ is not infinite, else the term $((T\ \lambda x.\mu a.(x\,z))\bar{y})$ is not normalizable, which is impossible, since
$x : \neg A, z : A, (y_i : O_i)_{1\leq i\leq n} \vdash_{\bar{O}} ((T\ \lambda x.\mu a.(x\,z))\bar{y}) : \bot$. ■

Corollary 5.4.2 *Let T be a closed term of type $(\neg X \to X) \to X$, then, for each term u, for each sequence $\bar{w} \in \mathcal{T}^{<\omega}$ and for each sequence $(v_i)_{i\in\mathbb{N}^*}$ of terms. There exist $m \in \mathbb{N}$ and terms $\theta_1, ..., \theta_m$ such that we have:*

- $(T\ u)\bar{w} \rhd^* \underline{\mu}.(u\ \theta_1)\ \bar{w}$
- $(\theta_i\ v_i) \rhd^* \underline{\mu}.(u\ \theta_{i+1})\ \bar{w}$ *for all* $1 \leq i \leq m-1$
- $(\theta_m\ v_m) \rhd^* \underline{\mu}.(v_i\ \bar{w})$ *for some* $1 \leq i \leq m$

Proof. Immediately from the previous theorem and the lemma 5.2.1. ■

Remark 5.4.3 *Let $\vdash T : (\neg X \to X) \to X$, the term T allows to modelizing the* `Call/cc` *instruction in the Scheme functional programming language.*

5.5 The completeness result

The following part is devoted to the construction of the completeness model.

Definition 5.5.1 (*and notation*)

1. *Let $\Omega = \{x_i \ / \ i \in \mathbb{N}\} \cup \{a_j \ / \ j \in \mathbb{N}\}$ an enumeration of infinite sets of λ and μ-variables.*
2. *Let $\Omega_1 = \{A_i \ / \ i \in \mathbb{N}\}$ an enumeration of all types where each type comes an infinite times.*
3. *Let $\Omega_2 = \{B_j \ / \ j \in \mathbb{N}\}$ an enumeration of all types where $\bot$ comes an infinite times.*
4. *We define $\mathbb{G} = \{x_i : A_i \ / \ i \in \mathbb{N}\}$ and $\mathbb{D} = \{a_j : B_j \ / \ j \in \mathbb{N}\}$.*
5. *Let u be a term, such that $Fv(u) \subseteq \Omega$, the contexts $\mathbb{G}_u$ (resp $\mathbb{D}_u$) are defined as the restrictions of $\mathbb{G}$ (resp $\mathbb{D}$) at the declarations containing the variables of $Fv(u)$.*
6. *The notation $\mathbb{G} \vdash u : C;\ \mathbb{D}$ means that $\mathbb{G}_u \vdash u : C;\ \mathbb{D}_u$, we denote $\mathbb{G} \vdash^* u : C;\ \mathbb{D}$ iff it exists a term u', such that $u \rhd^* u'$ and $\mathbb{G} \vdash u' : C;\ \mathbb{D}$.*
7. *Let $\mathbb{C} = \{a_j \ / \ (a_j : \bot) \in \mathbb{D}\}$ and $\mathbb{S} = \{t \ / \ \mathbb{G} \vdash^* t : \bot;\ \mathbb{D}\}$. For each propositional variable X we define a set of terms $\mathbb{R}_X = \{t \ / \ \mathbb{G} \vdash^* t : X;\ \mathbb{D}\}$.*

Lemma 5.5.1 *1. $\mathbb{S}$ is a $\mathbb{C}$-saturated set.*

2. *The sets $\mathbb{R}_X$ are saturated.*
3. *For each propositional variable X, $\mathbb{R}_X = \{a_j \ / \ a_j : X \in \mathbb{D}\} \rightsquigarrow \mathbb{S}$.*

4. $\mathbb{M} = \langle \mathbb{C}, \mathbb{S}, (\mathbb{R}_X)_{X \in \mathcal{P}} \rangle$ *is a model*

Proof. Easy. ■

Remark 5.5.1 *Observe that the model* $\mathbb{M}$ *is parametrized by the infinite sets of variables and the enumerations.*

Definition 5.5.2 *We define the* $\mathbb{M}$*-interpretation* $\mathbb{I}$ *as follows:*

- $\mathbb{I}(\bot) = \mathbb{S}$.
- $\mathbb{I}(X) = \mathbb{R}_X$ *for each propositional variable.*

Lemma 5.5.2 *Let* y *be a* λ*-variable,* $\sigma = [(x_i := y)_{1 \leq i \leq n}, (a_i :=^* y)_{1 \leq j \leq m}]$ *and* t *a term.*

1. *If* $(t\sigma\ y)$ *is normalizable, then* t *is normalizable.*
2. *If* $t\sigma$ *is normalizable, then* t *is normalizable.*

Proof. By a simultaneous induction on t, we use the standardization theorem of the $\lambda\mu$-calculus [21].

1. We examine the case where $t = \lambda x.u$. Then $(t\sigma\ y) = (\lambda x.u\sigma\ y)$ is normalizable, this implies that $u\sigma[x := y]$ is normalizable, hence by (2), u is normalizable, therefore t is normalizable too.
2. We examine the case where $t = (a\, u)$. Then $t\sigma = (a\ (u\sigma\ y))$ is normalizable, this implies that $(u\sigma\ y)$ is normalizable, hence by (1), u is normalizable, therfore t is normalizable too.

■

Corollary 5.5.1 *Let* t *by a term and* y *a* λ*-variable. If* $(t\ y)$ *is normalizable, then,* t *is normalizable also.*

Proof. Immediately from the previous lemma. ■

Lemma 5.5.3 *Let* t *and* τ *be two normal terms,* y *a* λ*-variable such that* $y \notin Fv(t)$, $(t\ y) \rhd^* \tau$, A *and* B *types, and* $\Gamma, y : A \vdash \tau : B;\ \Delta$. *Then* $\Gamma \vdash t : A \to B;\ \Delta$.

Proof. See the appendix. ■

Lemma 5.5.4 *Let* A *be a type and* t *a term.*

1. *If* $\mathbb{G} \vdash^* t : A\ ; \mathbb{D}$, *then* $t \in \mathbb{I}(A)$.

2. If $t \in \mathbb{I}(A)$, then $\mathbb{G} \vdash^* t : A \,;\mathbb{D}$.

Proof. By a simultaneous induction on the type A.

Proof of (1)

1. If $A = X$ or $\bot$, the result is immediately from the definition of $\mathbb{I}$.
2. Let $A = B \to C$ and $\mathbb{G} \vdash^* t : A \,;\mathbb{D}$, then $t \rhd^* t'$ such that: $\mathbb{G} \vdash t' : B \to C \,;\mathbb{D}$. Let $u \in \mathbb{I}(B)$. By the induction hypothesis (2), we have $\mathbb{G} \vdash^* u : B \,;\mathbb{D}$, this implies that $u \rhd^* u'$ and $\mathbb{G} \vdash u' : B \,;\mathbb{D}$. Hence $\mathbb{G} \vdash (t'\ u') : C \,;\mathbb{D}$, so, by the fact that $(t\ u) \rhd^* (t'\ u')$, we have $\mathbb{G} \vdash^* (t\ u) : C \,;\mathbb{D}$, then, by the induction hypothesis (1), $(t\ u) \in \mathbb{I}(C)$. Therefore $t \in \mathbb{I}(B \to C)$.

Proof of (2)

1. If $A = X$ or $\bot$, the result is immediately from the definition of $\mathbb{I}$.
2. Let $A = B \to C$, $t \in \mathbb{I}(B) \rightsquigarrow \mathbb{I}(C)$ and y be a λ- variable such $y \notin Fv(t)$ and $(y : B) \in \mathbb{G}$. We have $y : B \vdash y : B$, hence, by the induction hypothesis (1), $y \in \mathbb{I}(B)$, then, $(t\ y) \in \mathbb{I}(C)$. By the induction hypothesis (2), $\mathbb{G} \vdash^* (t\ y) : C \,;\mathbb{D}$, then $(t\ y) \rhd^* t'$ such that $\mathbb{G} \vdash t' : C \,;\mathbb{D}$ and, by the corollary 5.5.1, t is a normalizable term. The normal form of t can be either $(x\ u_1)\ u_2 ... u_n$ either $\lambda x.u$ or $\mu a.u$ (the case $(a\ u)$ gives a contradiction for typing reasons).
   (a) If $t \rhd^* (x\ u_1)\ u_2 ... u_n$ with u_i normal terms, then $\mathbb{G} \vdash (x\ u_1)\ u_2 ... u_n y : C \,;\mathbb{D}$, $x : E_1, E_2, ..., E_n \to (B \to C) \in \mathbb{G}$, $\mathbb{G} \vdash u_i : E_i \,;\mathbb{D}$ and $\mathbb{G} \vdash y : B \,;\mathbb{D}$. Therefore $\mathbb{G} \vdash (x\ u_1)\ u_2 ... u_n : B \to C \,;\mathbb{D}$, and finally $\mathbb{G} \vdash^* t : B \to C \,;\mathbb{D}$.
   (b) If $t \rhd^* \lambda x.u$ where u is a normal term, then, since $\mathbb{G}$ contains an infinite number of declarations for each type, let y be a λ-variable such that $y : B \in \mathbb{G}$ and $y \notin Fv(u)$. We have $(t\ y) \rhd^* u[x := y]$ and $\mathbb{G} \vdash u[x := y] : C \,;\mathbb{D}$, hence $\mathbb{G} \vdash \lambda y.u[x := y] : B \to C \,;\mathbb{D}$ and, by the fact that $y \notin Fv(u)$, $\lambda y.u[x := y] = \lambda x.u$. Therefore $\mathbb{G} \vdash \lambda x.u : B \to C \,;\mathbb{D}$, and finally $\mathbb{G} \vdash^* t : B \to C \,;\mathbb{D}$.
   (c) If $t \rhd^* \mu a.u$ where u is a normal term, then let y be a λ-variable such that $y : B \in \mathbb{G}$ and $y \notin Fv(u)$. We have $(t\ y) \rhd^* \mu a.u[a :=^* y] \rhd^* \mu a.u'$ where u' is the normal form of $u[a :=^* y]$, so we have $\mathbb{G}, y : B \vdash \mu a.u' : C \,;\mathbb{D}$. By the lemma 5.5.3, we obtain $\mathbb{G} \vdash \mu a.u : B \to C \,;\mathbb{D}$, and finally $\mathbb{G} \vdash^* t : B \to C \,;\mathbb{D}$.

■

Theorem 5.5.1 *Let A be a type and t a term. We have $t \in |A|$ iff $t \rhd^* t'$ and $\vdash t' : A$.*

Proof. $\Leftarrow$) By the lemma 5.3.3.

$\Rightarrow$) We can suppose that the sets $\mathbb{G}$ and $\mathbb{D}$ do not contain declarations for the free variables of t. If $t \in |A|$, then $t \in \mathbb{I}(A)$, hence by (1) of the lemma 5.5.4 and by the fact that $Fv(t') \subseteq Fv(t)$, we have $t \rhd^* t'$ and $\vdash t' : A$. ■

Corollary 5.5.2 *Let A be a type and t a term.*

1. *If $t \in |A|$, then t is normalizable and $t \simeq t'$, where t' is a closed term.*
2. *$|A|$ is closed under equivalence (i.e. if $t \in |A|$ and $t \simeq t'$, then, $t' \in |A|$).*

Proof. (1) is a direct consequence of theorem 5.5.1. (2) can be deduced from theorem 5.5.1 and the lemma 5.3.3. ■

5.6 Future work

Through this work, we have seen that the propositional types of the system S_μ are complete for the semantics defined previously. But what about the types of the second order typed $\lambda\mu$-calclus? We know that, for the system $\mathcal{F}$, the $\forall^+$-types (types with positive quatifiers) are complete for a realizability semantics [14] and [4]. But for the classical system $\mathcal{F}_C$, we cannot generalize this result. We check easily that, if $t = \mu a.(a\, \lambda y_1 \lambda z \mu b.(a\, \lambda y_2 \lambda x.z))$ and $A = \forall Y\{Y \to \forall X(X \to X)\}$, then $t \in |A|$, but t does not have the type A. We need to add more restrictions on the positions of the quantifier $\forall$ in the $\forall^+$-types to obtain a smallest class which we suppose that it can be proved complete. The problem is not the same when we consider the propositional classical natural deduction system with the connectives $\wedge$ and $\vee$. The term $\mu a.(a\ \langle \mu b.(a\ \langle \lambda x.x, \mu c.(b\ \lambda y.\lambda z.z)\rangle), \lambda x.x\rangle)$ belongs to the interpretation of the type $A = (X \to X) \wedge (X \to X)$ [15] and [16] but it does not have the type A. The treatment of the disjunction is even hard, so we think that to circumventing this difficulties, and if we hope a completeness theorem, we have to modified the semantics.

We give here a brief details of the precedent paragraph.

5.6.1 Second order typed $\lambda\mu$-calculus

Definition 5.6.1 *1. Types are formulas of second order propositional logic built from the set of propositional variables $\mathcal{P}$: $X, Y, Z, ...$ and a constant of type $\bot$ using the connector $\to$ and the quantifier $\forall$. As usual we denote by $\neg A$ the formula $A \to \bot$.*

2. *The typing rules are those of S_μ at which we add the rules:*

$$\frac{\Gamma \vdash t : A; \Delta}{\Gamma \vdash t : \forall X A; \Delta} \forall_i * \qquad \frac{\Gamma \vdash t : \forall X A; \Delta}{\Gamma \vdash t : A[X := F]; \Delta} \forall_e **$$

   $*$ *X is not free in Γ and Δ.*
   $**$ *For any type F.*

3. *The basic reduction rules are β and μ reductions.*

   We denote this typed system by $\mathcal{F}_\mathcal{C}$.

In addition to the strong normalisation and confulence properties, $\mathcal{F}_\mathcal{C}$ enjoys also the subject reduction property [20]:

Theorem 5.6.1 (Subject reduction) *If $\Gamma \vdash t : A; \Delta$ and $t \triangleright^* t'$ then $\Gamma \vdash t' : A; \Delta$.*

Remark 5.6.1 *We check that we get a similar results of those of the section 5.3, we have just to consider that:*

1. *The $\mathcal{C}$-saturated set $\mathcal{S}$ of the definition 5.3.2 is also closed under arbitrary intersection.*

2. *The definition of an $\mathcal{M}$-interpretation $\mathcal{I}$ is extended as follows: $\mathcal{I}(\forall X A) = \cap_{\mathcal{G} \in \mathcal{M}} \{\mathcal{I}^X_\mathcal{G}(A)\}$, where $\mathcal{I}^X_\mathcal{G}$ is the $\mathcal{M}$-interpretation such that: $\mathcal{I}^X_\mathcal{G}(X) = \mathcal{G}$ and $\mathcal{I}^X_\mathcal{G}(Y) = \mathcal{I}(Y)$ for $Y \neq X$.*

5.6.2 $\forall^+$ types and $\mathcal{D}^+$ types

Definition 5.6.2 *The $\forall^+$ (resp. $\forall^-$) types are defined as follows :*

- $\forall^- = \bot \mid X \mid \forall^+ \to \forall^-$.
- $\forall^+ = \bot \mid X \mid \forall^- \to \forall^+ \mid \forall X \forall^+$, *where X is free in the type $\forall^+$.*

The class of the $\forall^+$ types satisfies the completeness result in the intuitionistic logic coded by the terms of the λ-calculus. But in the case of the classical logic according to the previous semantics, we do not have such a result, here is a counter example.

Lemma 5.6.1 *Let the term $t = \mu a.(a\, \lambda y_1.\lambda z.\mu b.(a\, \lambda y_2.\lambda x.z))$, and the types Y, $Id = X \to X$ and $Id' = \forall X(X \to X)$. Then,*

1. *$y : Y \vdash (t\, y) : Id'$ and $t \in |Y \to Id'|$.*

2. *But, $\not\vdash t : Y \to Id'$.*

3. Nevertheless, $\vdash t : Y \to Id$.

Proof. 1) and 3) : Easy.
2) Else suppose that $\vdash t : Y \to Id'$, then,
$\vdash \lambda y_1.\lambda z.\mu b.(a\ \lambda y_2.\lambda x.z) : Y \to \forall X(X \to X);\ a : Y \to \forall X(X \to X)$, hence
$y_1 : Y \vdash \lambda z.\mu b.(a\ \lambda y_2.\lambda x.z) : \forall X(X \to X);\ a : Y \to \forall X(X \to X)$, therefore
$y_1 : Y, z : X \vdash \mu b.(a\ \lambda y_2.\lambda x.z) : Y \to \forall X(X \to X);\ a : Y \to \forall X(X \to X)$. This implies that :
$y_1 : Y, z : X \vdash \lambda y_2.\lambda x.z : Y \to \forall X(X \to X);\ a : Y \to \forall X(X \to X), b : X$, finally
$y_1 : Y, z : X, y_2 : Y \vdash \lambda x.z : Y \to \forall X(X \to X);\ a : Y \to \forall X(X \to X), b : X$.
This gives a contradiction since we can not obtain a such derivation using and respecting the rule $\forall_i$. ■

This counter example let us think that we need some restrictions on the $\forall^+$ types to obtain a completeness result for another class of types, which we denote by $\mathcal{D}^+$.

Definition 5.6.3 *The* $\mathcal{D}^+$, $\mathcal{D}^{++}$ *and* $\mathcal{D}^-$ *types are defined as follows:*

- $\mathcal{D}^- = \bot \mid X \mid \mathcal{D}^+ \to \mathcal{D}^-$.
- $\mathcal{D}^{++} = \bot \mid X \mid \mathcal{D}^- \to \mathcal{D}^{++}$.
- $\mathcal{D}^+ = \mathcal{D}^{++} \mid \forall X\mathcal{D}^+$, *where* X *is free in the type* $\mathcal{D}^+$

Intuitively a $\mathcal{D}^+$ *type is type which does not contain quantifiers in the right of the arrows.*

J.-L. Krivine gave a syntactical definition of the main data type, which is the following:

Definition 5.6.4 (*Data types*) *A data type is a formula* D *defined inductively as follows:* $D \equiv \forall X\{\Delta_1[X], ..., \Delta_n[X] \to X\}$, *where for each* $1 \leq j \leq n$ *we have:* $\Delta_j[X] \equiv \{D_1^j, ..., D_m^j, X, ..., X \to X\}$, *and where the* D_i^j *are data types.*

Example 5.6.1 *Here are the main basic data types:*

- *The boolean types:* $Bool = \forall X\{X, X \to X\}$.
- *The type of integers:* $Nat = \forall X\{(X \to X), X \to X\}$.
- *The type of list* (*finite sequences*) *of elements of a given data type* U*:* $LU = \forall X\{(U, X \to X,\ \ X \to X\}$.

Lemma 5.6.2 *The class of data types is included in the class of the* $\mathcal{D}^+$ *types.*

Proof. Let $D \equiv \forall X\{\Delta_1[X], ..., \Delta_n[X] \to X\}$ be a data type. It suffices to prove that each $\Delta_i \in \mathcal{D}^-$. Let $\Delta_i = \{D_1^i, ..., D_m^i, X, ..., X \to X\}$. It suffices to prove that $D_i \in \mathcal{D}^+$, which is true by the induction hypothesis. ■

5.6.3 The normal typing

Using the methodes seen in the subsection 5.4.1, which we have to develop more, a proof of the completeness have to pass by a similar result to the "*lemma*" 5.6.4 (a proof of this *lemma* is one of our perspectives).

Definition 5.6.5
1. *A $\forall$-cut (resp $\mu\forall$-cut) is a succession of $\forall_i$ (resp μ) and $\forall_e$ rules.*
2. *A $\forall$-normal derivation (resp $\mu\forall$-normal derivation) is a derivation which does not contain $\forall$-cuts (resp $\mu\forall$-cut).*
3. *A normal derivation is a derivation which is $\forall$-normal and $\mu\forall$-normal.*
4. *A height of a derivation is the number of the rules used in this derivation.*

Notation 5.6.1 *Let $\bar{X} = X_1, ..., X_n$ be a finite sequence of propositional variables, we denote by $\forall\bar{X}A$ the formula $\forall X_1 ... \forall X_n A$. We said that $\bar{X}$ is not free in A iff $X_i (1 \leq i \leq n)$ is not free in A. If $\bar{F} = F_1, ..., F_n$ is a finite sequence of formulas, we denote by $A[\bar{X} := \bar{F}]$ the formula $A[X_1 := F_1]...[X_n := F_n]$.*

Lemma 5.6.3 *If $\Gamma \vdash t : A\ ; \Delta$, then, for each sequence of formulas $\bar{F}$: $\Gamma[\bar{X} := \bar{F}] \vdash t : A[\bar{X} := \bar{F}]\ ; \Delta[\bar{X} := \bar{F}]$.*

Proof. By induction on the derivation. ■

Lemma 5.6.4 *Each derivation can be supposed normal.*

Proof.

Idea of a possible proof: This *lemma* can be deduced as a corollary of a result of strong normalisation of the union of the $\forall$-cut and $\mu\forall$-cut. This result of termination is still under studies. ■

5.7 Appendix

This part is devoted to the proof of the lemma 5.5.3.

Notation 5.7.1 *Let y be a λ-variable. The expression $u \triangleright_{\beta y} v$ (resp $u \triangleright_{\mu y} v$) means that we reduce in u only a β (resp μ)-redex where y is the argument, i.e, a redex in the form $(\lambda z.u\ y)$ (resp $(\mu b.u\ y)$). We denote by $\triangleright_y$ the union of $\triangleright_{\beta y}$ and $\triangleright_{\mu y}$ and $\triangleright_y^*$ (resp $\triangleright_{\beta y}^*$, $\triangleright_{\mu y}^*$) the transitive and reflexive closure of $\triangleright_y$ (resp $\triangleright_{\beta y}$, $\triangleright_{\mu y}$).*

Lemma 5.7.1 *Let t be a normal term, $\sigma = [(a_i :=^* y)_{1 \leq i \leq n}]$ and τ the normal form of $t\sigma$, then, $t\sigma \triangleright_y^* \tau$.*

Proof. By induction on the normal term t, the important case is the one where $t = (a_i\ u)$ and u a normal term, the others are direct consequences of the induction hypothesis. Let us examine the different forms of the normal term u, here there are two important subcases $u = \lambda x.v$ and $u = \mu b.v$ with v a normal term (these are the two cases where there is a creation of redexes after the substitution).

1. If $u = \lambda x.v$, then, $u\sigma = \lambda x.v\sigma$ and $t\sigma = (a_i\ (\lambda x.v\sigma\ \ y)) \triangleright_{\beta y} (a_i\ v\sigma[x := y])$. By the induction hypothesis, $v\sigma \triangleright_y^* v'$ where v' is the normal form of $v\sigma$, hence $(a_i\ v\sigma[x := y]) \triangleright_y^* (a_i\ v'[x := y])$ which is the normal form of $t\sigma$.

2. If $u = \mu b.v$, then, $u\sigma = \mu b.v\sigma$ and $t\sigma = (a_i\ (\mu b.v\sigma\ \ y)) \triangleright_{\mu y} (a_i\ \mu b.v\sigma[b :=^* y])$. By the induction hypothesis, $v\sigma[b :=^* y]$ is normalizable only with $\triangleright_y^*$ reductions, therefore $t\sigma$ is also normalizable only by $\triangleright_y^*$ reductions.

■

Lemma 5.7.2 *Let t be a normal term, τ the normal form of $t[a :=^* y]$ and A, B two types. If $\Gamma, y : A \vdash \tau : B; \Delta$. Then $\Gamma, y : A \vdash t[a :=^* y] : B; \Delta$.*

Proof. By induction on the length of the reduction $t[a :=^* y] \triangleright_y^* \tau$. By the lemma 5.7.1, it suffices to prove the following lemma. ■

Lemma 5.7.3 *Let τ be a normal term, t a term and A, B two types. If $t \triangleright_{\beta y} \tau$ (resp $t \triangleright_{\mu y} \tau$) and $\Gamma, y : A \vdash \tau : B\ ; \Delta$ then $\Gamma, y : A \vdash t : B\ ; \Delta$.*

Proof. By induction on t, we examine how $t \triangleright_{\beta y} \tau$ (resp $t \triangleright_{\mu y} \tau$). The proof is similar to the proof of (2) of the lemma 5.5.4. ■

Lemma 5.7.4 *Let t be a normal term, y a λ-variable such that $y \notin Fv(t)$, $\sigma = [a :=^* y]$ and A, B, C types. If $\Gamma, y : A \vdash t\sigma : B\ ; \Delta, a : C$, then, $\Gamma \vdash t : B\ ; \Delta, a : A \to C$.*

Proof. By induction on t.

1. $t = (x\, u_1)\, u_2 ... u_n$, then, $t\sigma = (x\, u_1\sigma)\, u_2\sigma ... u_n\sigma$ and $\Gamma\,, y : A \vdash (x\, u_1\sigma)\, u_2\sigma ... u_n\sigma : B\ ; \Delta, a : C$. Therefore $x : E_1, ..., E_n \to B \in \Gamma$ and $\Gamma, y : A \vdash u_i\sigma : E_i; \Delta, a : C$. By the induction hypothesis, we have $\Gamma \vdash u_i\sigma : E_i; \Delta, a : A \to C$, hence $\Gamma \vdash (x\ u_1)\, u_2 ... u_n : B\ ; \Delta, a : A \to C$.

2. $t = \lambda x.u$, then, $t\sigma = \lambda x.u\sigma$ and $\Gamma\,, y : A \vdash \lambda x.u\sigma : B; \Delta, a : C$, this implies that $B = F \to G$ and $\Gamma\,, y : A, x : F \vdash u\sigma : G; \Delta, a : C$. By the induction hypothesis, $\Gamma\,, x : F \vdash u : G; \Delta, a : A \to C$, then, $\Gamma \vdash \lambda x.u : F \to G; \Delta, a : A \to C$, therefore $\Gamma \vdash \lambda x.u : B; \Delta, a : A \to C$.

3. $t = \mu b.u$, then, $t\sigma = \mu b.u\sigma$ and $\Gamma\,, y : A \vdash \mu b.u\sigma : B\ ; \Delta, a : C$, this implies that $\Gamma, y : A \vdash u\sigma :\bot\ ; \Delta, a : C, b : B$. By the induction hypothesis, $\Gamma \vdash u :\bot\ ; \Delta, a : A \to C, b : B$, therefore $\Gamma \vdash \mu b.u : B\ ; \Delta, a : A \to C$.

4. $t = (a\ u)$, then $t\sigma = (a\ (u\sigma\ y))$ and $\Gamma\,, y : A \vdash (a\ (u\sigma\ y)) :\bot\ ; \Delta, a : C$, this implies that $\Gamma\,, y : A \vdash (u\sigma\ y) : C\ ; \Delta, a : C$ and $\Gamma\,, y : A \vdash u\sigma : A \to C\ ; \Delta, a : C$. By the induction hypothesis, $\Gamma \vdash u : A \to C\ ; \Delta, a : A \to C$, therefore $\Gamma \vdash (a\ u) :\bot\ ; \Delta, a : A \to C$.

5. $t = (b\ u)$, then, $t\sigma = (b\ u\sigma)$ and $\Gamma\,, y : A \vdash (b\ u\sigma) :\bot\ ; \Delta, a : C$, this implies that $\Gamma\,, y : A \vdash u\sigma : G\ ; \Delta, b : G, a : C$. By the induction hypothesis, $\Gamma \vdash u : G\ ; \Delta, b : G, a : A \to C$, therefore $\Gamma \vdash (b\ u) :\bot\ ; \Delta, a : C$.

■

Proof.[of lemma 5.5.3] By induction on t, the cases where $t = (x\ u_1)\, u_2 ... u_n$ and $t = \lambda x.u$ are similars to those in the proof of (2) of the lemma 5.5.4. Let us examine the case where $t = \mu a.u$, then $(t\ y) \rhd^* \mu a.u[a :=^* y] \rhd^* \mu a.u' = \tau$ where u' is the normal form of $u[a :=^* y]$. We have $\Gamma\,, y : A \vdash \mu a.u' : B;\ \Delta$, then $\Gamma\,, y : A \vdash u' :\bot;\ \Delta, a : B$. By the lemma 5.7.1, $u[a :=^* y] \rhd_y^* u'$, then, by the lemma 5.7.2, $\Gamma\,, y : A \vdash u[a :=^* y] :\bot;\ \Delta, a : B$. Hence by the lemma 5.7.4, $\Gamma \vdash u :\bot;\ \Delta, a : A \to B$ and finally $\Gamma \vdash \mu a.u : A \to B;\ \Delta$. ■

Bibliography

[1] T. Coquand *Completeness theorem and λ-calculus.* The 7th International Conference, TLCA 2005, Nara, Japan, April 21-23, 2005, pp. 1-9, volume 3461/2005.

[2] R. David and K. Nour. *A short proof of the strong normalization of the simply typed $\lambda\mu$-calculus.* Schedae Informaticae vol 12, pp. 27-33, 2003.

[3] R. David. *Une preuve simple de résultats classiques en λ-calcul.* Compte Rendu de l'Académie des Sciences. Paris, Tome 320, Série 1, pp. 1401-1406, 1995.

[4] S. Farkh and K. Nour. *Un résultat de complétude pour les types $\forall^+$ du système $\mathcal{F}$.* CRAS. Paris 326, Série I, pp. 275-279, 1998.

[5] S. Farkh and K. Nour. *Types Complets dans une extension du système $\mathcal{AF}2$.* Informatique Théorique et Application, 31-6, pp. 513-537, 1998.

[6] J.-Y. Girard, Y. Lafont, P. Taylor. Proofs and types. *Cambridge University Press*, 1986.

[7] T. Griffin. *A formulae-as-types notion of control.* Proc. POLP, 1990.

[8] J. R. Hindley. *The simple semantics for Coppe-Dezani-Sallé types.* Proceeding of the 5th Colloquium on International Symposium on Programming, pp. 212-226, April 06-08, 1982.

[9] J. R. Hindley. *The completeness theorem for typing λ-terms.* Theoretical Computer Science, 22(1), pp. 1-17, 1983.

[10] J. R. Hindley. *Curry's type-rules are complete with respect to the F-semantics too.* Theoretical Computer Science, 22, pp. 127-133, 1983.

[11] F. Kamareddine and K. Nour. *A completeness result for a realizability semantics for an intersection type system.* Submitted.

[12] J.-L. Krivine. *Lambda calcul, types et modèles.* Masson, Paris, 1990.

[13] J.-L. Krivine. *Opérateurs de mise en mémoire et traduction de Gödel.* Archive for Mathematical Logic, vol 30, pp. 241-267, 1990.

[14] R. Labib-Sami. *Typer avec (ou sans) types auxiliaires.* Manuscrit, 1986.

[15] K. Nour and K. Saber. *A Semantics of Realizability for the Classical Propositional Natural Deduction.* Electronic Notes in Theoretical Computer Science, vol 140, pp. 31-39, 2005.

[16] K. Nour and K. Saber. *A semantical proof of strong normalization theorem for full propositional classical natural deduction.* Archive for Mathematical Logic, vol 45, pp. 357-364, 2005.

[17] K. Nour. *Opérateurs de mise en mémoire et types $\forall$-positifs.* Theoretical Informatics and Applications, vol 30, n° 3, pp. 261-293, 1996.

[18] K. Nour. *Mixed Logic and Storage Operators.* Archive for Mathematical Logic, vol 39, pp. 261-280, 2000.

[19] M. Parigot *$\lambda\mu$-calculus: An algorithm interpretation of classical natural deduction.* Lecture Notes in Artificial Intelligence, vol 624, pp. 190-201. Springer Verlag, 1992.

[20] M. Parigot. *Proofs of strong normalization for second order classical natural deduction.* Journal of Symbolic Logic, vol 62 (4), pp. 1461-1479, 1997.

[21] W. Py. *Confluence en $\lambda\mu$-calcul.* PhD thesis, University of Chambéry, 1998.

[22] A. Saurin. *Separation and the $\lambda\mu$-calculus.* Proceedings of the Twentieth Annual IEEE Symp. on Logic in Computer Science, LICS 2005, IEEE Computer Society Press, pp. 356-365, 2005.

[23] W. W. Tait, *A realizability interpretation of the theory of species.* In : R. Parikh (Ed.), Logic Colloquium Boston 1971/72, vol. 435 of Lecture Notes in Mathematics, Springer Verlag, pp. 240-251, 1975.

Chapter 6

A call-by-value $\lambda\mu^{\wedge\vee}$-calculus

6.1 Introduction

In [8], Gentzen introduced the natural deduction system to study the notion of proof. The full classical natural deduction system is well adapted for the human reasoning. By full we mean that all the connectives ($\rightarrow$, $\wedge$ and $\vee$) and $\bot$ (for the absurdity) are considered as primitive. As usual, the negation is defined by $\neg A = A \rightarrow \bot$. Considering this logic from the computer science of view is interesting because, by the Curry-Howard correspondence, formulas can be seen as types for the functional programming languages and correct programs can be extracted. The corresponding calculus is an extension of M. Parigot's $\lambda\mu$-calculus with product and coproduct, which is denoted by $\lambda\mu^{\wedge\vee}$-calculus.

Ph. De Groote introduced in [7] the typed $\lambda\mu^{\wedge\vee}$-calculus to code the classical natural deduction system, and showed that it enjoys the main important properties: the strong normalization (However this proof is not finished yet, since the modified CPS-transformations do not preserve always the stricteness of reductions, and this for the same reasons pointed out in [10] and [11]), the confluence and the subformula property. This would guarantee that proof normalization may be interpreted as an evaluation process. As far as we know the typed $\lambda\mu^{\wedge\vee}$-calculus is the first extension of the simply typed λ-calculus which enjoys all the above properties. In [20], E. Ritter, D. Pym and L. Wallen introduced an extension of the $\lambda\mu$-calculus that features disjunction as primitive (see also [21]). But their system is rather different since they take as primitive a classical form of disjunction that amounts to $\neg A \rightarrow B$. Nevertheless, in [22], they give another extension of the $\lambda\mu$-calculus with an intuitionistic disjunction. However, the reduction rules considered are not sufficient to guarantee that the normal forms satisfy the subformula property. The question of the strong normalization of the full logic has interested several authors, thus one finds in [3], [9] and [13] different proofs of this result.

From a computer science point of view, the $\lambda\mu^{\wedge\vee}$-calculus may be seen as the kernel of a typed call-by-name functional language featuring product, coproduct and control operators. However we cannot apply an arbitrary reduction for implementation of programming languages, we have to fix a reduction strategy and usually it is the call-by-value strategy. Many programming langagues and control operations were developed through the studies of the call-by-value variant like `ML` and `Lisp` for λ-calculus, the calculus of exception handling $\lambda^{\rightarrow}_{exn}$ and $\mu\mathtt{PCF}_V$ for the $\lambda\mu$-calculus. In [15], C.-H. L. Ong and C. A. Stewart showed that $\mu\mathtt{PCF}_V$ is sufficiently strong to express the various control constructs such as the `ML-style raise` and the first-class continuations `callcc`, `throw` and `abort`. In this sense, it seems to be important to study the call-by-value version of $\lambda\mu^{\wedge\vee}$-calculus.

Among the important properties required in any abstract reduction system, there is the confluence which ensures the uniqueness of the normal form (if it exists). The notion of parallel reduction which is based on the method of Tait and Martin-Löf is a good tool to prove the confluence property for several reduction systems. The idea is very clear and intuitive: it consists in reducing a number of redexes existing in the term simultaneously. However, this method does not work for the $\lambda\mu^{\wedge\vee}$-calculus. In fact the diamond property which stipulates that: If $t \succ t'$ then $t' \succ t^*$ (where t^* is usually referred as the complete development of t) does not hold because more complicated situations appear, and that is due to the presence of the permutative reductions "$((u\,[x.v, y.w])\,\varepsilon) \triangleright (u\,[x.(v\,\varepsilon), y.(w\,\varepsilon)])$". Hence the proof of the confluence becomes hard and not at all trivial as it seems to be.

Let $t = (((u\,[x.v, y.w])\,[r.p, s.q])\,\varepsilon)$, $t_1 = ((u\,[x.(v\,[r.p, s.q]), y.(w\,[r.p, s.q])])\,\varepsilon)$, and $t_2 = ((u\,[x.v, y.w])\,[r.(p\,\varepsilon), s.(q\,\varepsilon)])$. We have: $t \succ t_1$ and $t \succ t_2$, if we want the diamond property to hold, t_1 and t_2 must be reduced to the same term t^* by one reduction step, however this is not possible. To make it possible we need another step of permutative reduction. We consider such a successive sequence of reductions as a one parallel reduction step, i.e. we follow the permutative reductions in the term step by step to a certain depth which allows to join and consider this sequence as a one reduction step. The notion of Prawitz' s *segment* yields the formulation of this new parallel reduction. Therefore the difficulties are overcame by extending this notion to our system (see [1], [2], [17] and [18]) and considering the extended structural reductions along this *segment* which allow us to define a complete development to obtain directly the common reductum, hence the Church-Rosser property. This is exactly what is done in [2]; our proof is just a checking that this method is well adapted to provide the diamond property for the call-by-value $\lambda\mu^{\wedge\vee}$-calculus including the symmetrical rules. Thus $t_1 \succ t^*$ and $t_2 \succ t^*$, where $t^* = (u\,[x.(v\,[r.(p\,\varepsilon), s.(q\,\varepsilon)]), y.(w\,[r.(p\,\varepsilon), s.(q\,\varepsilon)])])$.

This chapter is organized as follows. Section 2 is an introduction to the typed system, the relative cut-elimination procedure of $\lambda\mu^{\wedge\vee}$-calculus and the call-by-value $\lambda\mu^{\wedge\vee}$-calculus. In section 3, we define the parallel reduction related to the notion of *segment-tree*, thus we give the key lemma from which the diamond-

property is directly deduced. Section 4 is devoted to the proof of the key lemma. We conclude with some future work.

6.2 Notations and definitions

Definition 6.2.1 *We use notations inspired by [2].*

1. *Let $\mathcal{X}$ and $\mathcal{A}$ be two disjoint infinite alphabets for distinguishing the λ-variables and μ-variables respectively. We code deductions by using a set of terms $\mathcal{T}$ which extends the λ-terms and is given by the following grammar (which gives terms at the untyped level):*

$$\mathcal{T} := \mathcal{X} \mid \lambda\mathcal{X}.\mathcal{T} \mid (\mathcal{T}\ \ \mathcal{E}) \mid \langle\mathcal{T},\mathcal{T}\rangle \mid \omega_1\mathcal{T} \mid \omega_2\mathcal{T} \mid \mu\mathcal{A}.\mathcal{T} \mid (\mathcal{A}\ \ \mathcal{T})$$
$$\mathcal{E} := \mathcal{T} \mid \pi_1 \mid\ \pi_2 \mid [\mathcal{X}.\mathcal{T},\mathcal{X}.\mathcal{T}]$$

   An element of the set $\mathcal{E}$ is said to be an $\mathcal{E}$-term. Application between two $\mathcal{E}$-terms u and ε is denoted by $(u\ \varepsilon)$.

2. *Types are formulas of propositional logic built from the set of propositional variables and the constant type $\bot$, using the connectors $\to$, $\wedge$ and $\vee$.*

3. *The meaning of the new constructors is given by the typing rules below where Γ (resp. Δ) is a context, i.e. a set of declarations of the form $x : A$ (resp. $a : A$) where x is a λ-variable (resp. a is a μ-variable) and A is a formula.*

$$\frac{}{\Gamma, x : A\ \vdash x : A\ ;\ \Delta}\,ax$$

$$\frac{\Gamma, x : A \vdash t : B; \Delta}{\Gamma \vdash \lambda x.t : A \to B; \Delta}\to_i \qquad \frac{\Gamma \vdash u : A \to B; \Delta \quad \Gamma \vdash v : A; \Delta}{\Gamma \vdash (u\ \ v) : B; \Delta}\to_e$$

$$\frac{\Gamma \vdash u : A; \Delta \quad \Gamma \vdash v : B; \Delta}{\Gamma \vdash \langle u, v\rangle : A \wedge B; \Delta}\wedge_i$$

$$\frac{\Gamma \vdash t : A \wedge B; \Delta}{\Gamma \vdash (t\ \ \pi_1) : A; \Delta}\wedge_e^1 \quad \frac{\Gamma \vdash t : A \wedge B; \Delta}{\Gamma \vdash (t\ \ \pi_2) : B; \Delta}\wedge_e^2$$

$$\frac{\Gamma \vdash t : A; \Delta}{\Gamma \vdash \omega_1 t : A \vee B; \Delta}\vee_i^1 \quad \frac{\Gamma \vdash t : B; \Delta}{\Gamma \vdash \omega_2 t : A \vee B; \Delta}\vee_i^2$$

$$\frac{\Gamma \vdash t : A \vee B; \Delta \quad \Gamma, x : A \vdash u : C; \Delta \quad \Gamma, y : B \vdash v : C; \Delta}{\Gamma \vdash (t\ \ [x.u, y.v]) : C; \Delta}\vee_e$$

$$\frac{\Gamma \vdash t : A; \Delta, a : A}{\Gamma \vdash (a\ \ t) : \bot; \Delta, a : A}\bot_i \qquad \frac{\Gamma \vdash t : \bot; \Delta, a : A}{\Gamma \vdash \mu a.t : A; \Delta}\bot_e$$

4. *A term in the form* $(t\ [x.u, y.v])$ *(resp* $\mu a.t$*) is called an* $\vee_e$*-term (resp* $\bot_e$*-term).*

5. *The cut-elimination procedure corresponds to the reduction rules given below. They are those we need to the subformula property.*

   - $(\lambda x.u\ \ v) \rhd_\beta u[x := v]$
   - $(\langle t_1, t_2\rangle\ \ \pi_i) \rhd_\pi t_i$
   - $(\omega_i t\ \ [x_1.u_1, x_2.u_2]) \rhd_D u_i[x_i := t]$
   - $((t\ \ [x_1.u_1, x_2.u_2])\ \ \varepsilon) \rhd_\delta (t\ \ [x_1.(u_1\ \varepsilon), x_2.(u_2\ \varepsilon)])$
   - $(\mu a.t\ \ \varepsilon) \rhd_\mu \mu a.t[a :=^* \varepsilon]$
     where $t[a :=^* \varepsilon]$ *is obtained from* t *by replacing inductively each subterm in the form* $(a\ v)$ *by* $(a\ (v\ \varepsilon))$*.*

6. *Let* t *and* t' *be terms. The notation* $t \rhd t'$ *means that* t *reduces to* t' *by using one step of the reduction rules given above. Similarly,* $t \rhd^* t'$ *means that* t *reduces to* t' *by using some steps of the reduction rules given above.*

The following result is straightforward

Theorem 6.2.1 (Subject reduction) *If* $\Gamma \vdash t : A; \Delta$ *and* $t \rhd^* t'$*, then* $\Gamma \vdash t' : A; \Delta$*.*

We have also the following properties (see [2], [3], [7], [9], [13] and [14]).

Theorem 6.2.2 (Confluence) *If* $t \rhd^* t_1$ *and* $t \rhd^* t_2$*, then there exists* t_3 *such that* $t_1 \rhd^* t_3$ *and* $t_2 \rhd^* t_3$*.*

Theorem 6.2.3 (Strong normalization) *If* $\Gamma \vdash t : A; \Delta$*, then* t *is strongly normalizable.*

Remark 6.2.1 *Following the call-by-value evaluation discipline, in an application the evaluator has to diverge if the argument diverges. For example, in the call-by-value* λ*-calculus, we are allowed to reduce the* β*-redex* $(\lambda x.u\ v)$ *only when* v *is a value. In* $\lambda\mu$*-calculus, the terms* $\mu a.u$ *and* $(u\ [x_1.u_1, x_2.u_2])$ *cannot be taken as values, then the terms* $(\lambda x.t\ \mu a.u)$ *and* $(\lambda x.t\ (u\ [x_1.u_1, x_2.u_2]))$ *cannot be reduced. This will be able to prevent us from reaching many normal forms. To solve this problem, we introduce symmetrical rules* (δ'_v *and* μ'_v) *allowing to reduce these kinds of redexes.*

Now we introduce the call-by-value version of the $\lambda\mu^{\wedge\vee}$-calculus. From a logical point of view a value corresponds to an introduction of a connective; this is the reason why the Parigot's naming rule is considered as the introduction rule of $\bot$.

Definition 6.2.2 *1. The set of values $\mathcal{V}$ is given by the following grammar:*

$$\mathcal{V} := \mathcal{X} \mid \lambda\mathcal{X}.\mathcal{T} \mid \langle \mathcal{V}, \mathcal{V} \rangle \mid \omega_1\mathcal{V} \mid \omega_2\mathcal{V} \mid (\mathcal{A}\ \mathcal{T})$$

Values are denoted $U, V, W, ...$

2. The reduction rules of the call-by-value $\lambda\mu^{\wedge\vee}$-calculus are the followings:

- $(\lambda x.t\ \ V) \triangleright_{\beta_v} t[x := V]$
- $(\langle V_1, V_2 \rangle\ \ \pi_i) \triangleright_{\pi_v} V_i$
- $(\omega_i V\ \ [x_1.t_1, x_2.t_2]) \triangleright_{D_v} t_i[x_i := V]$
- $((t\ \ [x_1.t_1, x_2.t_2])\ \ \varepsilon) \triangleright_{\delta} (t\ \ [x_1.(t_1\ \varepsilon), x_2.(t_2\ \varepsilon)])$
- $(V\ (t\ \ [x_1.t_1, x_2.t_2])) \triangleright_{\delta'_v} (t\ \ [x_1.(V\ t_1), x_2.(V\ t_2)])$
- $(\mu a.t\ \ \varepsilon) \triangleright_{\mu} \mu a.t[a :=^* \varepsilon]$
- $(V\ \ \mu a.t) \triangleright_{\mu'_v} \mu a.t[a :=_* V]$
  where $t[a :=_* V]$ is obtained from t by replacing inductively each subterm in t in the form $(a\ u)$ by $(a\ (V\ u))$.

The first three rules are called logical rules and the others are called structural rules.

3. The one-step reduction $\triangleright_v$ of the call-by-value $\lambda\mu^{\wedge\vee}$-calculus is defined as the union of the seven rules given above. As usual $\triangleright_v^*$ denotes the transitive and reflexive closure of $\triangleright_v$.

The following lemma expresses the fact that the set of values is closed under reductions. In the remainder of this chapter, this fact will be used implicitly.

Lemma 6.2.1 *If V is a value and $V \triangleright_v^* W$, then W is a value.*

Proof. From the definition of the set of values. ■

Theorem 6.2.4 (Subject reduction) *If $\Gamma \vdash t : A\ ;\Delta$ and $t \triangleright_v^* t'$, then $\Gamma \vdash t' : A\ ;\Delta$.*

Proof. Since the reduction rules correspond to the cut-elimination procedure, we check easily that the type is preserved during the reduction from the redex to its reductom. ■

The rest of this chapter is an extension of [2] to our calculus according to the new considered reduction rules δ'_v and μ'_v. One can find all the notions given here in [2]. Since the new symmetrical rules that we add don't create any critical pair with the existing rules, then in the examples and proofs that we give, one will mention only the cases related to these new rules and check that they don't affect the core of [2]'s work.

6.3 The extended structural reduction

Definition 6.3.1 *1. Let t be a term, we define a binary relation denoted by $\sqsupset_t$ on subterms of t as follows:*

- $(u\ \ [x_1.u_1, x_2.u_2]) \sqsupset_t u_i$
- $\mu a.u \sqsupset_t v$, *where v occurs in u in the form $(a\ v)$*

If $u \sqsupset_t v$ holds, then v is called a segment-successor of u, and u is called a segment-predecessor of v. We denote by $\sqsupseteq_t$ the reflexive and transitive closure of $\sqsupset_t$.

2. *Let r be a subterm of a term t, such that r is a $\vee_e$- or $\perp_e$-term and r has no segment-predecessor in t. A segment-tree from r in t is a set $\mathcal{O}$ of subterms of t, such that for each $w \in \mathcal{O}$:*

   - $r \sqsupseteq_t w$
   - *w is a $\vee_e$- or $\perp_e$-term*
   - *For each subterm s of t, such that $r \sqsupseteq_t s \sqsupseteq_t w$ then $s \in \mathcal{O}$*

   r is called the root of $\mathcal{O}$.

3. *Let $\mathcal{O}$ be a segment-tree from r in t, a subterm v of t is called an acceptor of $\mathcal{O}$ iff v is a segment-successor of an element of $\mathcal{O}$ and v is not in $\mathcal{O}$.*

4. *A segment-tree $\mathcal{O}$ from r in t is called the maximal segment-tree iff no acceptor of $\mathcal{O}$ has a segment successor in t.*

5. *The acceptors of $\mathcal{O}$ are indexed by the letter $\mathcal{O}$.*

6. *Let $\mathcal{O}$ be a segment-tree from t in t itself, and $t \rhd_v^* t'$, then we define canonically a corresponding segment-tree to $\mathcal{O}$ in t' by the transformation of indexes from redexes to their residuals. This new segment-tree is denoted also by $\mathcal{O}$ if there is no ambiguity.*

Remark 6.3.1 *For typed terms, all the elements of a segment-tree have the same type.*

Definition 6.3.2 *Let $\mathcal{O}$ be a segment-tree from r in t, suppose that r occurs in t in the form $(V\, r)$ (resp $(r\, \varepsilon)$). The extended structural reduction of t along $\mathcal{O}$ is the transformation to a term t' obtained from t by replacing each indexed term $v_{\mathcal{O}}$ (the acceptors of $\mathcal{O}$) by $(V\, v)$ (resp $(v\, \varepsilon)$) and erasing the occurence of V (resp ε) in $(V r)$ (resp $(r\, \varepsilon)$) . This reduction is denoted by $t \succ_{\mathcal{O}} t'$.*

Remark 6.3.2 *By the definition above, every structural reduction is an extended structural reduction. It corresponds to the particular case where the segment-tree consists only of its root.*

Example 6.3.1 *Here are two examples of segment-trees and the extended structural reduction. Let $t = (u\ [x.\mu a.(a\ \langle x, (a\ w)\rangle), y.v])$ and V a value.*

1. *The set $\mathcal{O}_1 = \{t\}$ is a segment-tree from t in t itself. The acceptors of $\mathcal{O}_1$ are $\mu a.(a\ \langle x, (a\ w)\rangle)$ and v. Then t is represented as follows:*

   $t = (u\ [x.(\mu a.(a\ \langle x, (a\ w)\rangle))_{\mathcal{O}_1}, y.v_{\mathcal{O}_1}])$,
   and $(V\ t) \succ_{\mathcal{O}_1} (u\ [x.(V\ \mu a.(a\ \langle x, (a\ w)\rangle)), y.(V\ v)])$.

2. *The set $\mathcal{O}_2 = \{t, \mu a.(a\ \langle x, (a\ w)\rangle)\}$ is also a segment-tree from t in t. The acceptors of $\mathcal{O}_2$ are $\langle x, (a\ w)\rangle$, w and v. Then t is represented as follows:*

   $t = (u\ [x.\mu a.(a\ \langle x, (a\ w_{\mathcal{O}_2})\rangle_{\mathcal{O}_2}), y.v_{\mathcal{O}_2}])$,
   and $(V\ t) \succ_{\mathcal{O}_2} (u\ [x.\mu a.(a\ (V\ \langle x, (a\ (V\ w))\rangle)), y.(V\ v)])$.

Definition 6.3.3 *The parallel reduction $\succ$ is defined inductively by the following rules:*

- $x \succ x$
- *If $t \succ t'$, then $\lambda x.t \succ \lambda x.t'$, $\mu a.t \succ \mu a.t'$, $(a\ t) \succ (a\ t')$ and $\omega_i t \succ \omega_i t'$*
- *If $t \succ t'$ and $u \succ u'$, then $\langle t, u\rangle \succ \langle t', u'\rangle$*
- *If $t \succ t'$ and $\varepsilon \tilde{\succ} \varepsilon'$, then $(t\ \varepsilon) \succ (t'\ \varepsilon')$*
- *If $t \succ t'$ and $V \succ V'$, then $(\lambda x.t\ V) \succ t'[x := V']$*
- *If $V_i \succ V_i'$, then $(\langle V_1, V_2\rangle\ \pi_i) \succ V_i'$*
- *If $V \succ V'$ and $u_i \succ u_i'$, then $(\omega_i V\ [x_1, u_1, x_2, u_2]) \succ u_i'[x_i := V']$*
- *If $t \succ t'$, $V \succ V'$ (resp $\varepsilon \tilde{\succ} \varepsilon'$), and $\mathcal{O}$ is a segment-tree from t in t, and $(V'\, t') \succ_{\mathcal{O}} w$ (resp $(t'\, \varepsilon') \succ_{\mathcal{O}} w$), then $(V\, t) \succ w$ (resp $(t\, \varepsilon) \succ w$), where $\varepsilon \tilde{\succ} \varepsilon'$ means that:*

- $\varepsilon = \varepsilon' = \pi_i$, *or*
- $(\varepsilon = u$ *and* $\varepsilon' = u')$ *or* $(\varepsilon = [x.u, y.v]$ *and* $\varepsilon' = [x.u', y.v'])$ *such that* $u \succ u'$ *and* $v \succ v'$.

It is easy to see that $\rhd_v^*$ *is the transitive closure of* $\succ$.

Definition 6.3.4 *Let* t *be a term, we define the complete development* t^* *as follows:*

- $x^* = x$
- $(\lambda x.t)^* = \lambda x.t^*$
- $(\mu a.t)^* = \mu a.t^*$
- $\langle t_1, t_2\rangle^* = \langle t_1^*, t_2^*\rangle$
- $(\omega_i t)^* = \omega_i t^*$
- $(a\, t)^* = (a\, t^*)$,
- $(t\ \varepsilon)^* = (t^*\ \varepsilon^*)$, *if* $(t\,\varepsilon)$ *is not a redex*
- $(\lambda x.t\ V)^* = t^*[x := V^*]$
- $(\langle V_1, V_2\rangle\ \pi_i)^* = V_i^*$
- $(\omega_i V\ [x_1.u_1, x_2.u_2])^* = u_i^*[x := V^*]$
- *Let* $\mathcal{O}_m$ *be the maximal segment-tree from* t *in* t, *and* $(V^*\ t^*) \succ_{\mathcal{O}_m} w$ *(resp* $(t^*\ \tilde{\varepsilon^*}) \succ_{\mathcal{O}_m} w$*), then* $(V\ t)^* = w$ *(resp* $(t\ \varepsilon)^* = w$*), where* $\tilde{\varepsilon^*}$ *means:*
  - ε, *if* $\varepsilon = \pi_i$
  - u^*, *if* $\varepsilon = u$
  - $[x.u^*, y.v^*]$, *if* $\varepsilon = [x.u, y.v]$

Lemma 6.3.1 *1. If* $t \succ t'$ *and* $V \succ V'$, *then* $t[x := V] \succ t'[x := V']$.

2. If $t \succ t'$ *and* $\varepsilon \tilde{\succ} \varepsilon'$, *then* $t[a :=^* \varepsilon] \succ t'[a :=^* \varepsilon']$.

3. If $t \succ t'$ *and* $V \succ V'$, *then* $t[a :=_* V] \succ t'[a :=_* V']$.

Proof. By a straightforward induction on the structure of $t \succ t'$. ■

Lemma 6.3.2 (The key lemma) *If* $t \succ t'$, *then* $t' \succ t^*$.

Proof. The proof of this lemma will be the subject of the next section. ■

Theorem 6.3.1 (The Diamond Property) *If $t \succ t_1$ and $t \succ t_2$, then there exists t_3 such that $t_1 \succ t_3$ and $t_2 \succ t_3$.*

Proof. It is enough to take $t_3 = t^*$, then the theorem holds by the key lemma. ■

Since $\triangleright_v^*$ is identical to the transitive closure of $\succ$, we have the confluence of the call-by-value $\lambda\mu^{\wedge\vee}$-calculus.

Theorem 6.3.2 *If $t \triangleright_v^* t_1$ and $t \triangleright_v^* t_2$, then there exists a term t_3 such that $t_1 \triangleright_v^* t_3$ and $t_2 \triangleright_v^* t_3$.*

6.4 Proof of the key lemma

For technical reasons (see the example below, see also [2]), we start this section by extending the notion of the segment-tree.

Definition 6.4.1

1. *Let v be a subterm in a term t, v is called a bud in t iff v is t itself or v occurs in t in the form $(a\ v)$ where a is a free variable in t.*

2. *Let $\mathcal{O}_1$,...,$\mathcal{O}_n$ be segment-trees from respectively $r_1, ..., r_n$ in a term t, and $\mathcal{P}$ a set of buds (possibly empty) in t. Then a segment-wood is a pair $\langle \mathcal{O}_1 \cup ... \cup \mathcal{O}_n, \mathcal{P} \rangle$ such that:*

   - *r_i is a bud in t for each i,*
   - *$\mathcal{O}_1, ..., \mathcal{O}_n$ and $\mathcal{P}$ are mutually disjoints.*

3. *Let $\mathcal{Q} = \langle \mathcal{O}_1 \cup ... \cup \mathcal{O}_n, \mathcal{P} \rangle$ be a segment-wood in t, the elements of $\mathcal{O}_1 \cup ... \cup \mathcal{O}_n$ are called trunk-pieces of $\mathcal{Q}$, and those of $\mathcal{P}$ are called proper-buds of $\mathcal{Q}$.*

   (a) *We denote by $Bud(\mathcal{Q})$ the set of buds $\mathcal{P} \cup \{r_1, ..., r_n\}$ in t.*

   (b) *An acceptor of a segment-wood $\mathcal{Q}$ is either an acceptor of $\mathcal{O}_i$ for some i, either a proper-bud.*

   (c) *The acceptors of $\mathcal{Q}$ are indexed by $\mathcal{Q}$.*

   (d) *If the root r of a segment-tree $\mathcal{O}$ in t is a bud in t, then we identify $\mathcal{O}$ with the segment-wood $\langle \mathcal{O}, \emptyset \rangle$.*

4. *Let $\mathcal{Q}$ be a segment-wood in t, and s a subterm in t. The restriction of indexed subterms by $\mathcal{Q}$ to s constrcuts a segment-wood in s, which we will denote also by $\mathcal{Q}$ if there is no ambiguity.*

Remark 6.4.1

1. *If v is a bud in t, then v has no segment-predecessor in t. Therefore any segment-successor is not a bud.*

2. *Let $\mathcal{Q} = \langle \mathcal{O}_1 \cup ... \cup \mathcal{O}_n, \mathcal{P} \rangle$ be a segment-wood, since a segment-successor is not a bud, then any acceptor of any $\mathcal{O}_i$ is not in $Bud(\mathcal{Q})$.*

3. *The two conditions in (2) of the above definition are equivalent to the fact that all the elements of $\mathcal{P}$ and the buds $r_1, ..., r_n$ are distincts.*

4. *If $\mathcal{O}$ is a segment-tree from t in t, and s is a subterm in t, then the restriction of $\mathcal{O}$ to s constructs a segment-wood in s.*

5. *Proper-buds and trunk-pieces cannot be treated in a uniform way, since in a term, what will be indexed are the proper-buds themselves and the acceptors of the trunk-pieces, thing which is allowed by a formulation which makes difference between these two notions.*

Definition 6.4.2 *Let t, ε be $\mathcal{E}$-terms, V a value and $\mathcal{Q}$ a segment-wood in t, we define the term $t[V/\mathcal{Q}]$ (resp $t[\varepsilon/\mathcal{Q}]$) which is obtained from t by replacing each indexed term $v_{\mathcal{Q}}$ (the acceptors of $\mathcal{Q}$) in t by $(V\, v)$ (resp $(v\, \varepsilon)$).*

Remark 6.4.2 *It's clear that if $(V\, t) \succ_{\mathcal{O}} w$ (resp $(t\, \varepsilon) \succ_{\mathcal{O}} w$), then $w = t[V/\mathcal{O}]$ (resp $w = t[\varepsilon/\mathcal{O}]$) .*

Example 6.4.1 *Let $t = \mu a.(a\ \mu b.(b\ \omega_2 \lambda s.(a\ \omega_1 s)))$ be a term and r the subterm $\mu b.(b\ \omega_2 \lambda s.(a\ \omega_1 s))$ in t. We define two segment-trees from t in t, $\mathcal{O}_1 = \{t\}$ and $\mathcal{O}_2 = \{t, r\}$, observe that the acceptors of $\mathcal{O}_1$ are r and $\omega_1 s$, however those of $\mathcal{O}_2$ are $\omega_2 \lambda s.(a\ \omega_1 s)$ and $\omega_1 s$. The restriction $\mathcal{Q}_1$ (resp $\mathcal{Q}_2$) of $\mathcal{O}_1$ (resp $\mathcal{O}_2$) to r is the following segment-wood: $\mathcal{Q}_1 = \langle \emptyset, \{r, \omega_1 s\} \rangle$ (resp $\mathcal{Q}_2 = \langle \{r\}, \{\omega_1 s\} \rangle$). Remark also that $Bud(\mathcal{Q}_1) = Bud(\mathcal{Q}_2)$ and the set of trunk-pieces of $\mathcal{Q}_1$ is a subset of that of $\mathcal{Q}_2$. Suppose that V is a value then:*

- $t[V/\mathcal{Q}_1] = \mu a.(a\ (V\ \mu b.(b\ \omega_2 \lambda s.(a\ (V\ \omega_1 s)))))$.
- $t[V/\mathcal{Q}_2] = \mu a.(a\ \mu b.(b\ (V\ \omega_2 \lambda s.(a\ (V\ \omega_1 s)))))$.
- $t[V/\mathcal{Q}_1] \succ t[V/\mathcal{Q}_2]$

Lemma 6.4.1 *Let $\mathcal{Q}_1$ and $\mathcal{Q}_2$ be two segment-woods in a term t such that: $Bud(\mathcal{Q}_1) = Bud(\mathcal{Q}_2)$ and the set of all trunk-pieces of $\mathcal{Q}_1$ is a subset of that of $\mathcal{Q}_2$. Suppose also that $t \succ t'$ and $V \succ V'$ (resp $\varepsilon \tilde{\succ} \varepsilon'$), then $t[V/\mathcal{Q}_1] \succ t'[V'/\mathcal{Q}_2]$ (resp $t[\varepsilon/\mathcal{Q}_1] \succ t'[\varepsilon'/\mathcal{Q}_2]$).*

Proof. By induction on t. We look at the last rule used for $t \succ t'$. We examine only one case. The others are either treated similarly, either by a straightforward induction.

$t = (W\ u)$ and $t' = u'[W'/\mathcal{O}]$, where $\mathcal{O}$ is a segment-tree from u in u, $u \succ u'$ and $W \succ W'$.

- If t is not an acceptor of $\mathcal{Q}_1$ and then nor of $\mathcal{Q}_2$: By the induction hypothesis, $u[V/\mathcal{Q}_1] \succ u'[V'/\mathcal{Q}_2]$ and $W[V/\mathcal{Q}_1] \succ W'[V'/\mathcal{Q}_2]$. Since $\mathcal{O}$ is a segment-tree from u in u, we have:
$t[V/\mathcal{Q}_1] = (W[V/\mathcal{Q}_1]\ u[V/\mathcal{Q}_1]) \succ u'[V'/\mathcal{Q}_2][W'[V'/\mathcal{Q}_2]/\mathcal{O}] =$
$u'[W'/\mathcal{O}][V'/\mathcal{Q}_2] = t'[V'/\mathcal{Q}_2]$.

- If t is an acceptor of $\mathcal{Q}_1$ but not of $\mathcal{Q}_2$: Let $\mathcal{Q}_2 = \langle \mathcal{O}_t \cup \mathcal{O}_{r_1} \cup ... \cup \mathcal{O}_{r_n}, \mathcal{P} \rangle$ and $\mathcal{Q}_2^- = \langle \mathcal{O}_{r_1} \cup ... \cup \mathcal{O}_{r_n}, \mathcal{P} \rangle$, where $\mathcal{O}_s$ denotes a segment-tree from the bud s in t. By the induction hypothesis, $u[V/\mathcal{Q}_1] \succ u'[V'/\mathcal{Q}_2^-]$ and $W[V/\mathcal{Q}_1] \succ W'[V'/\mathcal{Q}_2^-]$. Moreover $(W'[V'/\mathcal{Q}_2^-]\ u'[V'/\mathcal{Q}_2^-]) \succ_{\mathcal{O}}$
$u'[V'/\mathcal{Q}_2^-][W'[V'/\mathcal{Q}_2^-]/\mathcal{O}]$. Hence $(W[V/\mathcal{Q}_1]\ u[V/\mathcal{Q}_1]) \succ$
$u'[V'/\mathcal{Q}_2^-][W'[V'/\mathcal{Q}_2^-]/\mathcal{O}]$. Therefore, $t[V/\mathcal{Q}_1] = (V\ (W[V/\mathcal{Q}_1]\ u[V/\mathcal{Q}_1]))$
$u'[V'/\mathcal{Q}_2][W'[V'/\mathcal{Q}_2^-]/\mathcal{O}][V'/\mathcal{O}_t] = u'[W'/\mathcal{O}][V'/\mathcal{Q}_2^-][V'/\mathcal{O}_t] =$
$u'[W'/\mathcal{O}][V'/\mathcal{Q}_2] = t'[V'/\mathcal{Q}_2]$.

- If t is an acceptor of $\mathcal{Q}_1$ and $\mathcal{Q}_2$, then $t[V/\mathcal{Q}_1] = (V\ (W[V/\mathcal{Q}_1]\ u[V/\mathcal{Q}_1]))$
$\succ (V'\ u'[V'/\mathcal{Q}_2][W'[V'/\mathcal{Q}_2]/\mathcal{O}]) = (V'\ u'[W'/\mathcal{O}][V'/\mathcal{Q}_2]) = t'[V'/\mathcal{Q}_2]$.

■

Proof.[of the key lemma]
By induction on t. We look at the last rule used in $t \succ t'$. Only one case is mentioned: $t = (V\ u)$ and $t' = u'[V'/\mathcal{O}]$ where $\mathcal{O}$ is a segment-tree from u in u, $u \succ u'$ and $V \succ V'$. In this case $t^* = u^*[V^*/\mathcal{O}_m]$, where $\mathcal{O}_m$ is the maximal segment-tree from u in u. Therefore, by the previous lemma (it' s clear that $\mathcal{O}$ and $\mathcal{O}_m$ as segment-woods satisfy the hypothesis of this lemma 6.4.1) and the induction hypothesis, $u'[V'/\mathcal{O}] \succ u^*[V^*/\mathcal{O}_m]$.

■

6.5 Future work

The strong normalization of this system cannot be directly deduced from that of $\lambda\mu^{\wedge\vee}$-calculus, since we consider the symmetric structural reductions μ'_v and δ'_v. Even if the strong normalization of $\lambda\mu\mu'$-calculus is well known (see [4]), the presence of μ'_v and δ'_v complicates the management of the duplication and the creation of redexes when the other reductions are considered.

Bibliography

[1] Y. Andou. *A normalization-procedure for the first order classical natural deduction with full logical symbols.* Tsukuba J. Math. 19 (1995) 153-162.

[2] Y. Andou. *Church-Rosser property of simple reduction for full first-order classical natural deduction.* Annals of Pure and Applied logic 119 (2003) 225-237.

[3] R. David and K. Nour. *A short proof of the Strong Normalization of Classical Natural Deduction with Disjunction.* Journal of symbolic Logic, vol 68, num 4, pp 1277-1288, 2003.

[4] R. David and K. Nour. *Arithmetical proofs of the strong normalization results for the symmetric $\lambda\mu$-calculus.* TLCA 2005, LNCS 3461, pp 162-178, 2005.

[5] R. David and K. Nour. *Why the usual candidates of reducibility do not work for the symmetric $\lambda\mu$-calculus.* Electronic Notes in Theoretical Computer Science, 2005.

[6] Ph. De Groote. *On the Strong Normalization of Natural Deduction with permutation-conversions.* In 10th International Conference on Rewriting Techniques and Application, RTA'99, volume 1631 of Lecture Notes in Computer Science, pages 45-59. Springer Verlag, 1999.

[7] Ph. De Groote. *Strong normalization of classical natural deduction with disjunction.* In 5th International Conference on typed lambda calculi and applications, TLCA'01. LNCS (2044), pp. 182-196. Springer Verlag, 2001.

[8] G. Gentzen. *Recherches sur la déduction logique.* Press Universitaires de France, 1955. Traduction et commentaires par R. Feys et J. Ladrière.

[9] R. Matthes. *Non-Strictly Positive Fixed Points for Classical Natural Deduction.* APAL, vol 133, pp. 205-230, 2005.

[10] K. Nakazawa. *Confluence and strong normalizability of call-by-value $\lambda\mu$-calculus* Theoretical Computer Science 290 (2003) 429-463.

[11] K. Nakazawa and M. Tatsuta. *Strong normalization proof with CPS-Translation for the second order classical natural deduction.* The Journal of Symbolic Logic, vol 68, num 3, pp. 851-859. Sept 2003.

[12] K. Nour and K. Saber. *A semantical proof of strong normalization theorem for full propositional classical natural deduction.* Archive of Mathematical Logic, 2005.

[13] K. Nour and K. Saber. *Confluency property of the call-by-value $\lambda\mu^{\wedge\vee}$-calculus.* Computational Logic and Applications CLA'05. Discrete Mathematics and Theoretical Computer Science proc, pp. 97-108, 2006.

[14] K. Nour and K. Saber. *Some properties of full propositional classical natural deduction.* Manuscript, 2007.

[15] C.-H. L. Ong and C. A. Stewart. *A Curry-Howard foundation for functional computation with control.* Conference Record of POPL'97: The 24th ACM SIGPLAN-SIGACT Symposium on Principes of Programming languages, pages. 215-227, Paris, France, 15-17 January 1997.

[16] M. Parigot *$\lambda\mu$-calculus: An algorithm interpretation of classical natural deduction.* Lecture Notes in Artificial Intelligence (624), pp. 190-201. Springer Verlag 1992.

[17] D. Prawitz *Natural deduction- A Proof Theoretical Study.* Almqvist & Wiksell. Stockholm, 1965.

[18] D. Prawitz *Idea and result in proof theory.* In: Proc . 2nd Scandinavian Logic Symp. North-Holland, Amsterdam, 1971, pp.235-307.

[19] W. Py. *Confluence en $\lambda\mu$-calcul.* PhD thesis, University of Chambéry, 1998.

[20] E. Ritter, D. Pym and L. Wallen *On the intuitionistic force of classical search.* Theoretical Computer Science, 232:299-333,2000.

[21] E. Ritter, D. Pym and L. Wallen *Proof-terms for classical and intuitionistic resolution.* Journal of Logic and Computation, 10(2):173-207, 2000.

[22] E. Ritter and D. Pym *On the semantic of classical disjunction.* Journal of Pure and Applied Algebra, vol 159, pp.315-338, 2001.

Résumé

Le $\lambda\mu^{\wedge\vee}$-calcul est une extension du λ-calcul associée à la déduction naturelle classique où sont considérés tous les connecteurs.

Les principaux résultats de cette thèse sont :

- La standardisation, la confluence et une extension de la machine de J.-L. Krivine en $\lambda\mu^{\wedge\vee}$-calcul.
- Une preuve sémantique de la forte normalisation du théorème d'élimination des coupures.
- Une sémantique de réalisabilité pour le $\lambda\mu^{\wedge\vee}$-calcul qui permet de caractériser le comportement calculatoire de certains termes typés et clos.
- Un théorème de complétude pour le $\lambda\mu$-calcul simplement typé.
- Une introduction à un $\lambda\mu^{\wedge\vee}$-calcul par valeur confluent.

Mots-clés : λ-calcul, $\lambda\mu$-calcul, standardisation, confluence, développements finis, forte normalisation, complétude, calcul par valeur, réductions parallèles.

Abstract

The $\lambda\mu^{\wedge\vee}$-calculus is an extension of the λ-calculus associated to the full classical natural deduction.

The main results of this thesis are:

- A standardization theorem, the confluence theorem, and an extension of J.-L. Krivine machine to the $\lambda\mu^{\wedge\vee}$-calculus.
- A semantical proof of the strong normalization theorem of the cut elimination procedure.
- A semantics of realizability for the $\lambda\mu^{\wedge\vee}$-calculus and characterization of the operational behavior of some closed typed terms.
- A completeness theorem for the simply typed $\lambda\mu$-calculus.
- A confluent call-by-value $\lambda\mu^{\wedge\vee}$-calculus.

Keywords: λ-calculus, $\lambda\mu$-calculus, standardization, confluence, finiteness developements, strong normalization, completeness, call-by-value, parallel reductions.