École polytechnique
Thèse de Doctorat
Spécialité: Informatique

Une étude logique du contrôle

appliquée à la programmation fonctionnelle et logique

Alexis Saurin

Thèse dirigée par
Dale Miller

Soutenue publiquement le
30 septembre 2008

Jury

Pierre-Louis	Curien	Rapporteur
Olivier	Danvy	Rapporteur
François	Fages	
Jean-Yves	Girard	
Olivier	Laurent	
Dale	Miller	Directeur
Luke	Ong	
Simona	Ronchi della Rocca	Rapporteur

À Laurence
Aux enfants de Cayenne

«La science ne doit pas faire oublier l'engagement...

«Pour être soi, il faut se projeter vers ce qui est étranger, se prolonger dans et par lui. Demeurer enclos dans son identité, c'est se perdre et cesser d'être. On se connaît, on se construit par le contact, l'échange, le commerce avec l'autre. Entre les rives du même et de l'autre, l'Homme est un pont.»
Jean-Pierre Vernant, *La Traversée des frontières*

l'engagement ne doit pas faire oublier la science.»
Pierre-Victor Tournier

«L'Homme construit trop de murs et pas assez de ponts.»
Construisons des ponts en traversant les murs.

Table des matières

Remerciements

Exercice de rigueur, une thèse est aussi un exercice de liberté. Mes premiers remerciements vont naturellement à mon directeur, Dale Miller, avec qui j'ai prononcé, il y a bien longtemps, mes premiers mots de logique linéaire (et mes premiers mots d'anglais...). Merci de m'avoir orienté, conseillé, dirigé et accompagné tout au long de cette thèse ; merci aussi pour la confiance et la liberté accordées pendant la thèse et lorsque s'est posée la question d'interrompre temporairement mon travail de recherche : si aucun travail de thèse n'est un processus continu, le mien a été particulièrement discontinu puisque ma thèse s'est déroulée en deux parties encadrant une parenthèse d'un an. Dale a su me remettre le pied à l'étrier à ma reprise alors que j'errais un peu : c'est parce qu'il m'a épaulé à ce moment-là que j'ai pu arriver au bout de cette recherche ; une discussion avec Alain Finkel n'est pas non plus étrangère au fait que je sois parvenu à terminer cette thèse : qu'il en soit sincèrement remercié. Je tiens également à remercier Dominique Grésillon, alors directeur de l'école doctorale de l'École Polytechnique, pour l'attention et le sérieux qu'il a apporté à ma demande d'interruption de thèse. Prudent au début car s'inquiétant d'un éventuel «décrochage», il m'a soutenu une fois qu'il a eu compris mes motivations.

Avant Dale, c'est avec François Fages que j'ai eu mon premier contact avec la recherche, à l'INRIA, alors que j'étais depuis peu à l'École normale. Je le remercie profondément de cette première immersion dans une équipe de recherche et je suis d'autant plus heureux qu'il ait accepté de faire partie de mon jury. Peu après, le cours de λ-calcul de Pierre-Louis Curien m'a ouvert des horizons (un peu à la manière des cours de maths de taupe), horizons qui ont continué de s'étendre en DEA sous sa direction et de cela, je lui suis infiniment reconnaissant. Je lui exprime également toute ma gratitude pour avoir accepté de participer à mon jury et d'être rapporteur, pour le travail de relecture impressionnant qu'il a effectué et pour ses remarques toujours pertinentes. Simona Ronchi della Rocca m'a fait l'honneur, en plus de m'accueillir à Turin, d'accepter de participer au jury et d'être rapporteur de cette thèse : un grand merci dans l'attente de collaborations scientifiques que j'espère fructueuses. Olivier Danvy a également accepté d'être rapporteur de ma thèse alors même qu'il connaissait peu mon travail et que je le lui ai demandé un peu tard : je tiens donc à lui exprimer toute ma gratitude ce qui est, paraît-il, fort banal ! Je tiens à remercier Jean-Yves Girard pour le très grand honneur qu'il m'a fait en acceptant de participer à mon jury, pour le temps qu'il m'a consacré à plusieurs occasions dans la seconde partie de mon travail de thèse et pour tous les mystères qu'il nous a révélés ! Un grand merci à Luke Ong d'avoir accepté de participer à mon jury malgré ma décision de rédiger cette thèse en français. Il m'a semblé aller de soi de demander à Olivier Laurent de participer à mon jury et je le remercie donc tout particulièrement d'avoir accepté d'en faire partie.

Cette thèse doit beaucoup à des discussions scientifiques avec de nombreuses personnes dont Claudia Faggian, Alexandre Miquel, Patrick Baillot, Paul-André Mélliès et Kazushige Terui que je remercie pour la clarté de leurs explications et l'enthousiasme communicatif dont ils font preuve en parlant de leurs recherches. Il me faut bien sûr remercier mes co-auteurs Michele, Kaustuv, Silvia et Hugo pour la richesse de nos séances de travail et de nos discussions. Un remerciement tout particulier à Hugo dont la richesse scientifique et le foisonnement d'idées m'ont émerveillé et guidé. Je remercie très amicalement Lorenzo Tortora di Falco pour l'accueil qu'il m'a réservé dans son laboratoire romain à l'automne 2007, période au cours de laquelle a été élaborée une partie substantielle de cette thèse. Merci aussi à François Maurel qui a quelque responsabilité dans mon sujet de thèse !

Le LIX est un environnement de recherche privilégié où la forte concentration en idées ne doit pas qu'au manque de place. Un grand merci à Jean-Pierre pour la direction du laboratoire et pour le soutien et les conseils qu'il n'a pas hésité à me prodiguer à plusieurs occasions. Merci aux membres du LIX et en particulier à toute l'équipe Parsifal, tout particulièrement à ses thésards David, Olivier et Vivek du bureau $\mathit{Tarski}^{\perp} \parr \mathit{Church}$. Merci à Miki, Romain, Florian et Frank pour leurs nombreuses contributions extra-scientifiques à cette thèse et à Assia, Pierre-Yves, Fabien et Stéphane en souvenir de la 0D4. Merci à Catuscia pour ses nombreux conseils qui, souvent, complétaient bien ceux de Dale, notamment en matière administrative. Merci à Catherine et Isabelle, à James et Matthieu sans l'aide précieuse desquelles tout serait moins simple ; merci aussi à M. Baronnet et Mme. Dessertaine de l'EDX. La mise en page de cette thèse doit beaucoup à Emmanuel Beffara et à son package LaTeX `thmbox`, qu'il en soit remercié.

L'enseignement est un ingrédient fort utile au développement d'une pensée scientifique. Merci à Frank Taïeb, mon prof de sup, de m'avoir permis de faire mes premiers pas d'enseignant en informatique, merci à Madhavan et Kumar pour la chance extraordinaire qu'a été l'enseignement d'un module de logique au Chennai Mathematical Institute et à Gilles, François et Jean-Marc pour l'intérêt que j'ai trouvé à enseigner dans leurs cours à l'X et pour la pédagogie et l'attention aux élèves qui y étaient de rigueur. Dans un autre ordre d'idée, je dois bien remercier l'administration pénitentiaire, et plus précisément la prison de la Santé, pour les longues heures d'attente dans ces réduits décrépits qu'on appelle parloirs-avocat dans lesquels certains éléments de cette thèse ont été élaborés.

La formation d'un scientifique ne doit pas qu'à la pratique scientifique. Je tiens à exprimer toute ma gratitude à trois personnes rencontrées pendant ma parenthèse : Pierre-Victor Tournier, Alain Cugno et Maître Robert Badinter. Leur finesse, leur détermination, leur énergie et leur humanité sont pour moi des modèles, pas simples à suivre. Merci aux enfants de Cayenne, ceux du dedans, ceux du dehors. Merci à Carine, Vincent, Élise, Julian et Ève, eux qui m'ont accompagné, supporté ou tout simplement porté pendant une folle année. Merci à Farid et Zigor pour ces rencontres improbables mais si riches.

Je tiens à remercier Adrien Deloro, un ami comme on en fait peu[1], un frère. Je ne saurais exprimer à quel point je suis redevable à mes parents, Nathalie et William, et à ma sœur, Irlande, pour ce qu'ils m'ont apporté depuis toujours. Sans eux, rien n'aurait commencé : merci à vous trois.

Je remercie Laurence, pour tout, tout simplement : sans elle, cette thèse n'aurait pas beaucoup de sens et la vie manquerait de goût et de couleur.

[1] Même si je n'oublierai pas de si tôt son «tu verras, pour ta rédaction de thèse, tu t'enfermes avec les Sonates de Beethoven et en trois semaines, c'est bouclé», en tous les cas, avec Chopin, Brahms, Wagner ou Pergolese, cela ne marche pas ! Un coup de chapeau, au passage, à ces quatre-là qui m'ont accompagné pendant de dures semaines de rédaction dans la dernière phase de cette thèse.

Introduction

Théorie de la démonstration et théorie du calcul, et plus précisément théorie des langages de programmation, sont étroitement liées. Outre la coïncidence temporelle qui fait qu'on a vu se développer à la même période (les années 30) des cadres théoriques[2] et des thèses ou résultats fondamentaux[3] qui ont sans doute posé les fondements de la théorie de la démonstration moderne et contribué à marquer le début de l'ère informatique, on a très rapidement vu des scientifiques développer à la fois un intérêt pour la théorie des preuves et la théorie du calcul.

Si les liens entre calcul et démonstration formelle peuvent sembler évidents au premier abord du fait de leur caractère explicite, formel, vérifiable (voire mécanisable) et laissant peu de place à la subjectivité (quoique), c'est véritablement dans les années 60 avec la correspondance de Curry-Howard que se noue le nœud qui tient depuis lors théorie de la preuve et théorie des langages de programmation étroitement liées.

C'est ainsi que des avancées[4] d'un côté du pont ont tôt fait de produire de nouveaux résultats – ou mieux, de nouvelles questions et problématiques – sur l'autre rive. Ces échanges de bons procédés ont lieu dans les deux sens et vont des questions les plus fondamentales aux plus appliquées, des interrogations les plus traditionnelles aux innovations les plus récentes. Ainsi les recherches et les attentes concernant les langages de programmation ont-elles stimulé les investigations mathématiques dans le champ de la preuve.

Les liens qui ont été établis[5] entre les langages de programmation déclaratifs et la théorie mathématique des preuves sont tout à fait exemplaires, particulièrement intéressants et principalement de deux types :

- d'un côté, la correspondance de Curry-Howard déjà évoquée plus haut a permis de fournir un fondement théorique particulièrement solide à la programmation fonctionnelle en mettant en correspondance formule logique et type informatique, preuve intuitionniste et programme, réduction de coupure et étape de calcul et enfin preuve sans coupure et résultat, ou valeur. Elle a aussi permis de transférer des résultats de théorie de la démonstration en théorie de la programmation et a été l'occasion de développer des outils d'analyse formelle de ces langages s'appuyant sur des méthodes logiques. La traversée du pont dans l'autre direction ne fut pas moins riche, la correspondance ayant ainsi renouvelé les questions qui se posaient au sujet des résultats d'élimination des coupures ; des questions liées à l'efficacité de ces transformations ont notamment pu trouver un écho plus attentif du fait de ces connexions...

[2] λ-calcul, machines de Turing, calcul des séquents, déduction naturelle, voire fonctions récursives de la calculabilité.

[3] théorème de complétude de Gödel, théorème de Herbrand, théorèmes d'incomplétudes de Gödel et autres résultats d'indécidabilité, interprétation de Brouwer-Heyting-Kolmogorov des preuves intuitionnistes, théorème d'élimination des coupures de Gentzen, thèse de Church-Turing...

[4] doit-on parler de découverte ou d'invention dans nos champs ?

[5] «tissés» serait peut-être plus juste.

– dans un autre ordre d'idées, les questions liées à la recherche (automatique) de démonstration ont été à l'origine d'un paradigme de programmation apparu dans les années 70, la programmation logique dont Prolog est le représentant canonique. Dans cette direction également on a vu des contributions dans les deux directions, outre le fait que le langage lui-même vienne de considérations logiques. Ainsi en est-il de l'introduction des notions de preuves uniformes ou de focalisation en logique linéaire qui ont un grand intérêt logique en elles-mêmes et ont été introduites à l'occasion de recherches en programmation logique.

Les deux paradigmes de programmation sus-mentionnés, programmation fonctionnelle et programmation logique, sont également intéressants en ce qu'ils entretiennent avec la logique des relations de nature assez différente et que cette différence se cristallise autour de la notion de coupure :

– en programmation fonctionnelle, un programme est une démonstration et la dynamique du calcul réside dans la réduction de la coupure ; le résultat du calcul est une preuve sans coupure. Ce paradigme, celui du λ-calcul, consiste à considérer le calcul comme une ***réduction de preuve*** ;
– en programmation logique, un programme est une collection de formules (un séquent) et une étape de calcul est le passage d'un séquent à un autre par application d'une règle d'inférence. La dynamique du calcul réside donc dans la construction d'une preuve et le résultat du calcul est une preuve complètement construite. La recherche de preuve n'a de chance d'être effective que si l'on n'utilise pas la règle d'inférence qu'est la coupure, la seule règle du calcul des séquents qui demande véritablement de l'inventivité, de l'imagination. En ce sens, la recherche de preuve repose de manière fondamentale sur l'élimination des coupures, mais pas dans son aspect effectif. Le paradigme de la programmation logique est donc celui de la ***recherche de preuve sans coupure***.

Mais, alors que les résultats des calculs dans les deux paradigmes sont des objets identiques, des preuves sans coupure, la dynamique du calcul[6] et le lien que cette dynamique entretient avec la règle de coupure sont très différents d'un cas à l'autre.

On peut voir cela sous un autre angle : la règle de coupure peut être vue comme un moyen de faire interagir, ou communiquer, les preuves. Dans le paradigme de calcul par réduction de preuve, cette communication correspondra par exemple au passage d'un argument à une procédure. Dans le paradigme de calcul par recherche de preuve, la coupure servira plutôt[7] à relier le résultat d'un calcul (une preuve) à une propriété que l'on aura établie à propos du système (par des moyens mécaniques ou non) et à en déduire une information à propos d'autres calculs potentiels par exemple : on pourra ainsi déduire du fait qu'un calcul a abouti et du fait que l'on a élimination des coupures qu'un autre calcul serait possible (ou non) : c'est ici plus l'admissibilité de la coupure que l'élimination de la coupure qui compte et elle sert plus dans le raisonnement à propos des calculs que pour le déroulement des calculs eux-mêmes.

Une différence fondamentale entre logique et programmation (et qui est peut-être l'un des éléments qui font la richesse du lien preuves–programmes) est que la logique a un caractère «sacré» alors qu'un langage de programmation est, assez fondamentalement, fait pour être «bidouillé», «hacké». Si l'on ne sait pas très bien d'où vient la logique (c'est

[6]Et la question visant à savoir comment on calcule est au moins aussi importante que celle interrogeant ce qu'on calcule.

[7]Outre le fait de rendre possible la recherche de preuve par des moyens mécaniques.

peut-être le propre de son caractère mathématique), les langages de programmation sont très clairement œuvre humaine : il s'agit d'un objet technique, d'un outil artificiel dans le premier sens du terme. Loin de nous l'idée que la logique ne soit pas sujette au bidouillage (on la torture même parfois...) ni qu'il n'y ait pas de caractère universel dans les langages de programmation, bien au contraire, mais force est de constater que les critères de conception, de validation et d'évaluation de l'une et des autres diffèrent et c'est, nous semble-t-il, pour le meilleur.

L'un des exemples les plus frappants de la distinction que l'on vient de mentionner est peut-être la question du contrôle dans les langages de programmation. Les langages de programmation (qu'il s'agisse du paradigme fonctionnel ou logique) sont construits sur un modèle d'exécution sous-jacent (plus ou moins formel) et (presque) tous prévoient des manières de déroger à ce modèle standard d'exécution. Qu'il s'agisse des mécanismes de capture du contexte d'évaluation, de mécanismes d'échappement, de modifications des comportements de backtracking ou d'élagages de l'arbre de recherche de preuve, on peut généralement considérer les opérateurs de contrôle comme une manière de réifier l'environnement d'exécution du programme et de donner au programmeur une main sur cet environnement d'exécution (il y a souvent, derrière cette volonté, d'importantes questions d'efficacité).

Toujours est-il que les mécanismes de contrôle sont une donnée et un problème bien complexe. D'après la brève présentation que l'on vient de donner des mécanismes de contrôle, le lecteur ne sera pas surpris par le fait qu'il soit difficile d'interpréter logiquement les constructions de contrôle, mais c'est un défi intéressant qui a des implications (applications ?) pratiques loin d'être négligeables : c'est bien souvent dans ces constructions complexes à manipuler, car jouant avec les limites même du modèle de programmation que le programmeur a à l'esprit, que viennent se nicher les erreurs de programmation, voire les erreurs de conception du langage.

Les recherches que nous présentons dans cette thèse se situent au carrefour de la théorie de la démonstration et de la théorie des langages de programmation. On parle souvent de logique de la programmation pour désigner cette interface. Les résultats que nous présentons peuvent être reliés à trois thématiques, outre ce que nous avons déjà discuté plus haut : la propriété de séparation (ou théorème de Böhm), la propriété de focalisation et l'approche interactive de la logique et du calcul. Nos résultats concernent trois domaines, le $\lambda\mu$-calcul, la logique linéaire et la ludique. Cette thèse est composée de trois parties présentant chacun de ces domaines avec, en toile de fond, un domaine d'application commun : une meilleure compréhension du contrôle dans les langages fonctionnels et logiques par l'utilisation de méthodes issues de théorie de la démonstration.

Les trois parties de la thèse sont largement indépendantes, quant aux questions abordées ainsi qu'aux outils techniques utilisés et aux résultats qui y sont présentés. Il y a pourtant plusieurs éléments qui font le lien entre les différentes parties et qui justifie, nous l'espérons, que ces résultats soient regroupés dans la même thèse :

– Notre motivation pour l'étude d'une approche interactive de la recherche de preuve était une meilleure compréhension du contrôle dans les langages logiques et le développement du $\Lambda\mu$-calcul éclaire quant à lui certains éléments concernant le contrôle dans les langages fonctionnels ;
– L'origine de notre intérêt pour le $\lambda\mu$-calcul était la propriété de séparation. L'approche que nous avons adoptée, visant à analyser ce qui faisait échouer la séparation et à réparer le problème, quitte à modifier le calcul, était inspirée de la ludique où la

propriété de séparation est traitée comme un élément requis dans le développement de la théorie ;
– La propriété de focalisation est elle aussi au fondement de la ludique et c'est pour mieux comprendre ce sur quoi elle reposait que nous avons mené nos recherches dans cette direction ;
– Le $\Lambda\mu$-calcul fait partie de la tradition Curry-Howard, ou réduction de preuve, tandis que les résultats de focalisation et la programmation logique sont apparentés à la recherche de preuve. Nous tentons, via la recherche de preuve interactive, de proposer un cadre dans lequel la recherche de preuve se fait par élimination des coupures : la recherche de preuve interactive serait une sorte de recherche de preuve par élimination des coupures.

Contenu et contributions de la thèse :

– *Partie introductive.* Nous introduisons dans les deux premiers chapitres de la thèse les notions techniques qui seront nécessaires au développement de nos travaux.
  – **Chapitre 1 : Notions de théorie de la démonstration.** Nous introduisons dans ce chapitre des notions générales de théorie de la démonstration qui nous serviront dans le reste de la thèse, et tout particulièrement les calculs des séquents classique, LK, et linéaire, LL. Nous avons fait le choix, dans ce chapitre, de ne pas nous limiter aux notions strictement nécessaires, mais de présenter des notions ou résultats qui, s'ils ne sont pas forcément nécessaires aux développements techniques qui suivront, mettent en évidence le lien qu'il peut y avoir entre différentes questions, différents résultats. À ce dernier titre, ce chapitre s'inspire d'une série de cours sur la théorie de la démonstration que nous avons donnés aux graduate students du Chennai Mathematical Institute.
  – **Chapitre 2 : Notions de λ-calcul et de $\lambda\mu$-calcul.** Dans ce chapitre, nous introduisons le λ-calcul de Church et le $\lambda\mu$-calcul de Parigot. Nous donnons également une discussion détaillée du théorème de Böhm.

– *Partie I : $\Lambda\mu$-calcul.* Dans cette première partie, nous étudions le $\Lambda\mu$-calcul.
  – **Chapitre 3 : Du $\lambda\mu$-calcul au $\Lambda\mu$-calcul.** Ce chapitre est consacré à une analyse de l'échec de la séparation en $\lambda\mu$-calcul, prouvé par David et Py, ce qui nous conduit à proposer une variante du $\lambda\mu$-calcul, le $\Lambda\mu$-calcul, pour laquelle la séparation est vérifiée, et que nous introduisons dans ce chapitre.
  – **Chapitre 4 : Théorème de Böhm pour le $\Lambda\mu$-calcul.** Nous prouvons dans ce chapitre le théorème de séparation (théorème de Böhm) pour le $\Lambda\mu$-calcul. Les résultats de ce chapitre ont été publiés dans [Sau05].
  – **Chapitre 5 : Confluence du $\Lambda\mu$-calcul.** Ce chapitre est consacré à la preuve de la confluence du $\Lambda\mu$-calcul. La preuve est assez longue et complexe mais elle a pour corollaire une preuve de la confluence du $\lambda\mu\eta$-calcul, légèrement plus courte que la preuve précédemment connue. Certains résultats de ce chapitre ont été publiés dans [Sau08a].
  – **Chapitre 6 : $\Lambda\mu$-calcul simplement typé.** Nous étudions dans ce chapitre un système de types simples pour le $\Lambda\mu$-calcul. Les résultats de ce chapitre ont été publiés dans [Sau08a].
  – **Chapitre 7 : Une analyse du $\Lambda\mu$-calcul via des réseaux.** Ce chapitre est consacré à l'étude d'une variante des réseaux de démonstration de la logique linéaire, les SANE, grâce auxquels nous analysons les réductions du $\Lambda\mu$-calcul de

manière plus atomique. Nous prouvons en particulier un théorème de séparation pour ces réseaux, une simulation du $\Lambda\mu$-calcul, et expliquons dans quelle mesure un autre calcul est sous-jacent aux SANE. Ce chapitre a donné lieu à un rapport de recherche co-écrit avec Michele Pagani [PS08].
  - **Chapitre 8 : $\lambda\mu$-calculs, streams et contrôle.** Dans ce dernier chapitre concernant le $\Lambda\mu$-calcul, nous établissons quelques résultats comparant diverses versions du $\lambda\mu$-calcul que l'on trouve dans la littérature, nous développons l'interprétation de stream du $\Lambda\mu$-calcul et développons la connexion avec le contrôle délimité. Certains résultats de ce chapitre ont été publiés dans [Sau08a].

- *Partie II : Focalisation en logique linéaire.* Nous nous penchons dans cette partie sur les propriétés de focalisation de la logique linéaire.
  - **Chapitre 9 : Une preuve modulaire de la focalisation.** Ce chapitre est consacré à une preuve de la focalisation basée sur la structure de graphe de focalisation. Ce chapitre a donné lieu à publication dans [MS07].
  - **Chapitre 10 : Multifocalisation.** Nous développons dans ce chapitre une variante des systèmes focalisés que nous appelons multifocalisation. La multifocalisation trouve une origine très naturelle dans la preuve du chapitre précédent et les graphes de focalisation. Nous mettons en évidence l'existence d'une notion de maximalité parmi les preuves multifocalisées et établissons des résultats comparant preuves maximalement multifocalisées et réseaux de démonstration. Certains résultats de ce chapitre ont été publiés dans [CMS08].

- *Partie III : Programmation Ludique.* Cette dernière partie s'attache à proposer un modèle de recherche de preuve interactive, ou de recherche de preuve par élimination des coupures, dans le cadre de la ludique.
  - **Chapitre 11 : Introduction à la ludique.** Ce chapitre introduit la ludique.
  - **Chapitre 12 : Vers une programmation ludique : Recherche de preuve interactive.** Nous proposons dans ce chapitre de considérer différemment la recherche de preuve. Alors que dans les approches usuelles, en particulier les approches de la programmation logique via les preuves uniformes, la construction de preuve consiste en la spécification d'un programme par un séquent et la recherche d'une preuve de ce séquent en suivant une discipline spécifique (prouvabilité uniforme ou focalisation), voire en utilisant des stratégies particulières pour les instructions de contrôle, notre modèle de recherche de preuve interactive consiste à construire une preuve (un dessein de la ludique) qui soit capable d'interagir, de normaliser, avec un ensemble de tests, lui-même spécifié comme des desseins de la ludique. Nous proposons une machine abstraite pour exécuter cette recherche de preuve interactive et montrons comment traiter le backtracking de manière interactive. Ce chapitre a donné lieu à publication dans [Sau08b].

Chapitre 1

Notions de théorie de la démonstration.

Résumé:

On introduit dans ce chapitre des notions générales de théorie de la démonstration qui nous serviront dans le reste de la thèse.

Sommaire

The King of France was a man. I was a man.
Therefore I was the King of France.
F. ! I'm sinking again.
Leonard Cohen, *Beautiful Lovers*

Nous introduisons dans ce chapitre un certain nombre de notions de théorie de la démonstration en vue des développements à suivre dans le reste de cette thèse.

Nous avons été guidés dans la rédaction de ce chapitre par deux idées :
- la nécessité de poser les définitions des notions nécessaires au développement des résultats originaux de cette thèse ;
- plutôt qu'une liste, un catalogue de définitions sans cohérence sinon l'exhaustivité, nous avons souhaité mettre en perspective diverses notions de théorie de la démonstration, des thèmes récurrents ou encore des notions qui se font écho.

Nous avons ainsi renoncé à ne présenter que ce qui était strictement nécessaire pour développer nos résultats de manière à mettre en évidence certains liens et des connections entre divers thèmes de théorie de la démonstration et de ses liens avec le calcul.

Ainsi, l'approche de la complétude de LK de la section 1.2 ne nous sert pas dans la suite de la thèse mais nous permet de mettre en évidence des origines à l'approche de recherche de preuve et à la focalisation dans des thèmes classiques de théorie de la démonstration. De même, la présentation des deux systèmes de règles alternatifs pour conjonction et disjonction dans LK en section 1.1.5 n'a peut-être pas d'autre utilité que de rendre plus naturelle l'introduction des fragments multiplicatifs et additifs de la logique linéaire. Un autre exemple est la discussion sur la constructivité de LJ : les mentions en section 1.3.3 des propriétés d'existence et de disjonction de LJ, pour nécessaires qu'elles soient dans une telle introduction, pourraient paraître suffisantes et l'introduction des théories constructives de Harrop pourrait au contraire sembler superflue. Ce point a été guidé par la volonté de «préparer le terrain» pour l'introduction des formules héréditaires de Harrop et la prouvabilité uniforme à la section 1.6.2.

Le lecteur qui souhaiterait plus de détails sur l'un des points abordés dans ce chapitre pourra consulter l'un des ouvrages de Girard [Gir87b, GLT89, Gir06], Kleene [Kle67, Kle04] Miller [Mil99], David, Nour et Raffalli [DNR04] ou Troelstra et Schwichtenberg [TS96].

1.1. Calcul des séquents classique

1.1.1 Termes et formules

Définition 1.1 (*Langage du premier ordre*)

Un **langage du premier-ordre** $\mathcal{L}$ *est la donnée (i) d'ensembles* $\mathcal{P}_n$ *de symboles de prédicat pour chaque arité* $n \geq 0$*, (ii) d'ensembles* $\mathcal{F}_n$ *de symboles de fonction pour chaque arité* $n \geq 0$ *(les symboles de fonction d'arité nulle sont appelés des symboles de constante), et (iii) d'un ensemble dénombrable* $\mathcal{X}$ *de variables.*

Définition 1.2 (***Termes de $\mathcal{L}$***)

Les **termes** *de $\mathcal{L}$ sont définis inductivement comme suit :*
- *une variable x est un terme ;*
- *si $t_1, \dots, t_n$ sont des termes et si f est un symbole de fonction d'arité n ($f \in \mathcal{F}_n$), alors $f(t_1, \dots, t_n)$ est un terme.*

Définition 1.3 (***Formules de $\mathcal{L}$***)

Les **formules** *de $\mathcal{L}$ (notées $A, B, \dots, F, G, \dots$) sont définies inductivement comme suit :*
- *si $t_1, \dots, t_n$ sont des termes et si P est un symbole de prédicat d'arité n ($P \in \mathcal{P}_n$), alors $P(t_1, \dots, t_n)$ est une formule (dite* **formule atomique***) ;*
- *si A et B sont des formules, alors $A \wedge B$, $A \vee B$, $A \Rightarrow B$, $\neg A$ sont des formules ;*
- *si A est une formule et si x est une variable, alors $\forall x.A$ et $\exists x.A$ sont des formules.*

Dans la suite, quand on parlera d'une formule A, on parlera le plus souvent d'une occurrence particulière de la formule A. On ne détaille pas maintenant ce point qui sera précisé lorsque nous introduirons les calculs des séquents.

Remarque 1.4

On considèrera également par la suite des logiques du second ordre, c'est-à-dire avec une quantification sur les prédicats et non plus seulement sur les termes. On introduira dans ce cas un ensemble supplémentaire $\mathfrak{X}$ de variables du second ordre et de nouvelles constructions pour les formules : une variable du second ordre X sera une formule et si $X \in \mathfrak{X}$ et si A est une formule, $\exists^2 X.A$ et $\forall^2 X.A$ seront des formules.

Définition 1.5 (***Variables libres et liées***)

Les **occurrences libres et liées** *d'une variable dans une formule sont définies comme suit :*
- *une occurrence de variable dans un terme (resp. une formule atomique) est toujours libre ;*
- *si une occurrence de x est libre dans A, alors elle est libre dans $A \wedge B$, $B \wedge A$, $A \vee B$, $B \vee A$, $A \Rightarrow B$, $B \Rightarrow A$ et dans $\neg A$;*
- *si une occurrence de x est libre dans A et si $y \neq x$, alors elle est libre dans $\forall y.A$ et dans $\exists y.A$;*
- *une occurrence libre de x dans A est liée dans $\forall x.A$ et dans $\exists x.A$;*
- *une occurence liée de x dans A est liée dans toute formule construite sur A.*

Comme d'habitude, le nom d'une variable liée n'importe pas, la seule chose qui importe étant le quantificateur responsable de la liaison de la variable. En conséquence, on autorise le renommage de variables liées (α-renommage) et on considérera toujours les formules à ce renommage près. En outre, on pourra également supposer que la convention de Barendregt s'applique : aucune variable n'apparaît à la fois libre et liée dans une formule et les variables liées sont toutes distinctes les unes des autres (c'est-à-dire que toute paire de quantificateurs apparaissant dans la formule lient des variables distinctes).

Définition 1.6 (***Modèle d'un langage du premier-ordre***)

Étant donné un langage du premier-ordre $\mathcal{L}$, un **modèle** *$\mathcal{M}$ de $\mathcal{L}$ consiste en un ensemble non-vide $|\mathcal{M}|$, le* **domaine de** *$\mathcal{M}$, une fonction $f^{\mathcal{M}} : |\mathcal{M}|^n \mapsto |\mathcal{M}|$ pour chaque symbole de fonction f dans $\mathcal{F}_n$ et une fonction $P^{\mathcal{M}} : |\mathcal{M}|^n \mapsto \{v, f\}$ (ou, de manière équivalente, une partie $P^{\mathcal{M}}$ de $|\mathcal{M}|^n$) pour tout symbole de prédicat P de $\mathcal{P}_n$.*

Étant donné un langage $\mathcal{L}$ et un modèle $\mathcal{M}$ pour $\mathcal{L}$, on dénotera par $\mathcal{L}[\mathcal{M}]$ le langage $\mathcal{L}$ étendu par les constantes $\overline{c}$ pour $c \in |\mathcal{M}|$ (et on étend bien sûr de manière canonique le modèle $\mathcal{M}$ en un modèle de $\mathcal{L}[\mathcal{M}]$ avec $\overline{c}^{\mathcal{M}} = c$).

On peut maintenant associer une valeur de vérité à chaque formule close de $\mathcal{L}[\mathcal{M}]$:

Définition 1.7 (*Interprétation d'une formule close*)

Étant donné un terme clos t ou une formule close F de $\mathcal{L}[\mathcal{M}]$, on peut définir leur interprétation comme suit :

- $\mathcal{M}(\overline{c}) = c$;
- $\mathcal{M}(f(t_1, \ldots, t_n)) = f^{\mathcal{M}}(\mathcal{M}(t_1), \ldots, \mathcal{M}(t_n))$;
- $\mathcal{M}(P(t_1, \ldots, t_n)) = P^{\mathcal{M}}(\mathcal{M}(t_1), \ldots, \mathcal{M}(t_n))$;
- $\mathcal{M}(F \wedge G) = v$ *si* $\mathcal{M}(F) = \mathcal{M}(G) = v$ *et vaut* f *sinon ;*
- $\mathcal{M}(F \vee G) = f$ *si* $\mathcal{M}(F) = \mathcal{M}(G) = f$ *et vaut* v *sinon ;*
- $\mathcal{M}(F \Rightarrow G) = v$ *si* $\mathcal{M}(F) = v$ *ou si* $\mathcal{M}(G) = v$, *et vaut* f *sinon ;*
- $\mathcal{M}(\neg F) = v$ *si* $\mathcal{M}(F) = f$ *et* f *sinon ;*
- $\mathcal{M}(\forall x.F) = v$ *si pour tout* $c \in |\mathcal{M}|$, $\mathcal{M}(F[\overline{c}/x] = v$ *et vaut* f *sinon ;*
- $\mathcal{M}(\exists x.F) = v$ *si, pour un certain* $c \in |\mathcal{M}|$, $\mathcal{M}(F[\overline{c}/x] = v$ *et vaut* f *sinon ;*

Un ***système déductif*** ou ***système d'inférence*** ou ***système de démonstration*** est un formalisme dans lequel on peut étudier des propriétés des formules logiques, éventuellement pour les relier aux interprétations possibles des formules. Une caractéristique importante des systèmes déductifs est qu'une démonstration formelle sera généralement un objet fini (ou au moins un objet admettant une description finie) et même un objet défini inductivement, là où les modèles de la logique font intervenir des objets infinis. Les démonstrations sont donc naturellement plus sujettes à un traitement symbolique automatique.

Nous introduirons dans la suite de ce chapitre plusieurs systèmes de démonstration et diverses logiques qui nous seront utiles pour la thèse et nous rappellerons certaines de leurs propriétés fondamentales. Nous nous pencherons en particulier sur deux calculs introduits par Gerhard Gentzen dans les années 30 [Gen69], la déduction naturelle et le calcul des séquents.

1.1.2 Calcul des séquents LK

Le calcul des séquents est un système déductif introduit par Gentzen dans les années 30 [Gen69]. Si ce système n'est pas très naturel ni pratique pour y faire effectivement des démonstrations mathématiques, il se révèle être un excellent outil pour étudier la structure des démonstrations.

Définition 1.8 (*Séquent pour LK*)

*Un séquent pour LK est la donnée de deux ensembles d'***occurrences de formules**, *les hypothèses Γ et les conséquences Δ, et sera noté $\Gamma \vdash \Delta$.*

Un séquent $\Gamma \vdash \Delta$ doit se comprendre comme l'assertion affirmant que la ***conjonction*** des formules de Γ ***entraîne*** (ou ***a pour conséquence***) la ***disjonction*** des formules de Δ. C'est volontairement que nous ne précisons pas le sens donné à «conjonction», «entraîner», «avoir pour conséquence» et «disjonction», puisque nous souhaitons justement étudier la signification et la valeur que ces termes prennent une fois qu'on a donné un système de démonstration particulier que nous souhaitons étudier.

$$\frac{}{A \vdash A}\ \textit{Initial} \qquad\qquad \frac{\Gamma_1 \vdash A, \Delta_1 \quad \Gamma_2, A \vdash \Delta_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\ \textit{Coupure}$$

FIG. 1.1 – Calcul des Séquents LK – Groupe Identité (*Règle initiale et Coupure*).

$$\frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta}\ \textit{LW} \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash A, \Delta}\ \textit{RW} \qquad \frac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta}\ \textit{LC} \qquad \frac{\Gamma \vdash A, A, \Delta}{\Gamma \vdash A, \Delta}\ \textit{RC}$$

FIG. 1.2 – Calcul des Séquents LK – Groupe Structurel (*Affaiblissement et Contraction*).

Remarque 1.9 (***Sur l'utilisation d'occurences de formules***)

On fera bien la distinction entre un **ensemble de formules**, *un* **multi-ensemble de formules** *et un* **ensemble d'occurrences de formules** *:*

- *dans le premier cas, on n'aura pas de répétition de formule ;*
- *dans le deuxième cas, on pourra avoir une multiplicité de formules identiques mais on ne pourra pas distinguer une occurrence d'une autre occurence ;*
- *dans le troisième cas, on aura à la fois la possibilité de considérer de multiples occurrences d'une même formules et de faire la distinction entre ces occurrences et en particulier de distinguer les inférences appliquées à chacune d'entre elles.*

La manière la plus commode de distinguer les occurrences de formules est de leur attribuer des indices, des adresses permettant de distinguer chaque occurrence.

Définition 1.10 (***Calcul des séquents LK***)

Les **règles d'inférence** *(ou* **inférences***) de LK sont données aux figures 1.1, 1.2 et 1.3. Elles sont réparties en trois groupes : le* **groupe identité** *qui contient la règle initiale et la coupure, le* **groupe structurel** *qui contient des règles permettant de gérer les formules du séquents (contraction, affaiblissement) et enfin le* **groupe logique** *qui définit les règles d'inférence des connecteurs logiques.*

Les inférences ont toujours un **séquent conclusion** *et un certain nombre (0, 1 ou 2) de* **séquents prémisses**.

Une **preuve** *$\mathcal{D}$ d'un séquent $\Gamma \vdash \Delta$ est un arbre fini de règles d'inférence de LK tel que*

- *$\Gamma \vdash \Delta$ est le séquent conclusion de la première inférence r de la preuve (on dit alors que $\Gamma \vdash \Delta$ est la* **conclusion** *ou la* **racine** *de la preuve et que r est l'***inférence conclusion***) ;*
- *si $(r_i)_{i \in I}$ sont immédiatement au-dessus de r dans $\mathcal{D}$, les conclusions des $(r_i)_{i \in I}$ sont les prémisses de r ;*
- *les feuilles de l'arbre sont des inférences sans prémisse.*

$$\frac{\Gamma \vdash A, \Delta}{\Gamma, \neg A \vdash \Delta} L\neg \qquad \frac{\Gamma, A \vdash \Delta}{\Gamma \vdash \neg A, \Delta} R\neg \qquad \frac{}{\Gamma, \bot \vdash \Delta} L\bot \qquad \frac{}{\Gamma \vdash \top, \Delta} R\top$$

$$\frac{\Gamma, A \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta} L\wedge 1 \quad \frac{\Gamma, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta} L\wedge 2 \quad \frac{\Gamma \vdash A, \Delta \quad \Gamma \vdash B, \Delta}{\Gamma \vdash A \wedge B, \Delta} R\wedge$$

$$\frac{\Gamma, A \vdash \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \vee B \vdash \Delta} L\vee \quad \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash A \vee B, \Delta} R\vee 1 \quad \frac{\Gamma \vdash B, \Delta}{\Gamma \vdash A \vee B, \Delta} R\vee 2$$

$$\frac{\Gamma_1 \vdash A, \Delta_1 \quad \Gamma_2, B \vdash \Delta_2}{\Gamma_1, \Gamma_2, A \Rightarrow B \vdash \Delta_1, \Delta_2} L\Rightarrow \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \Rightarrow B, \Delta} R\Rightarrow$$

$$\frac{\Gamma, A[t/x] \vdash \Delta}{\Gamma, \forall x.A \vdash \Delta} L\forall \qquad \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash \forall x.A, \Delta} R\forall\ (*) \qquad \frac{\Gamma, A \vdash \Delta}{\Gamma, \exists x.A \vdash \Delta} L\exists\ (*) \qquad \frac{\Gamma \vdash A[t/x], \Delta}{\Gamma \vdash \exists x.A, \Delta} R\exists$$

(*) Pour ces règles, $x \notin FV(\Gamma, \Delta)$.

FIG. 1.3 – Calcul des Séquents LK – Groupe Logique ($\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\forall$, $\exists$, $\top$, $\bot$).

Remarque 1.11

De la même manière que nous distinguons différentes occurrences de formules, nous distinguerons différentes occurrences d'un séquent ou d'une règle d'inférence dans une preuve.

De manière générale, et dès que cela n'introduit pas d'ambiguïté, nosu nous autoriserons à omettre la mention du terme «occurrences de» et parlerons de formules (resp. séquents, resp. règles d'inférence) à la place d'occurrences de formules (resp. de séquents, resp. de règles d'inférence).

Remarque 1.12

Au lieu de travailler sur un ensemble d'occurrences de formules, on pourrait considérer des séquents constitués à partir de listes (avec répétition) de formules en considérant une règle structurelle supplémentaire, l'échange :

$$\frac{\Gamma_1, B, A, \Gamma_2 \vdash \Delta}{\Gamma_1, A, B, \Gamma_2 \vdash \Delta} LEx \qquad \frac{\Gamma \vdash \Delta_1, B, A, \Delta_2}{\Gamma \vdash \Delta_1, A, B, \Delta_2} REx$$

On y gagnerait une certaine précision avec la possibilité d'étudier les preuves qui nécessitent l'utilisation de la règle d'échange et les cas où l'on peut se passer de cette règle (ce qui conduit à l'étude des logiques non-commutatives où la règle d'échange est restreinte (règle d'échange cyclique par exemple) voire interdite.

Le fait de considérer des listes de formules et la règle d'échange permet de faire disparaître l'ambiguïté liée à l'utilisation d'ensembles ou de multi-ensembles de formules, mais dans ce cas, il devient important que les règles d'inférences s'appliquent, par exemple, seulement aux formules de Γ *et* Δ *les plus proches du symbole* $\vdash$ *et non pas à n'importe quelle formule du séquent.*

Définition 1.13 (*Formule principale d'une inférence*)

Soit $\Gamma \vdash \Delta$ un séquent conclusion d'une inférence r. On dit qu'une occurrence de formule $F \in \Gamma \vdash \Delta$ est la **formule principale** *de r si F est la formule distinguée du séquent à laquelle s'applique l'inférence r.*

Remarque 1.14

Plus précisément, la notion de formule principale correspond à ce qui suit :

- *dans le cas d'une règle logique :*
  - *le connecteur le plus haut de F est le connecteur correspondant à la règle r. Ce connecteur est appelé* **connecteur logique principal***;*
  - *l'occurrence F n'apparaît dans aucune des prémisses de r. On dit que F est détruite ou consommée par l'inférence;*
  - *des sous-formules de F apparaissent dans chaque prémisse de r. Il s'agit des* **descendantes**, **occurences filles** *(ou plus simplement* **sous-occurrences***) de F ;*
- *dans le cas d'une règle structurelle, la formule principale est la formule contractée ou affaiblie.*

Remarque 1.15

Kleene [Kle52] utilise la terminologie de descendant dans le sens opposé : c'est la formule du séquent conclusion qui est descendante de la formule de la prémisse : nos «occurrences filles» sont appelées «ancêtres» chez Kleene. Nous utilisons pour notre part une terminologie compatible avec la notion de sous-formule et qui est plus naturelle vis-à-vis des questions posées en recherche de preuve.

Propriété 1.16

Toutes les règles de LK sauf celles du groupe identité ont exactement une formule principale.

La règle d'identité (ou d'axiome) a deux formules principales tandis que la coupure n'a aucune formule principale.

Définition 1.17 (*Formules du contexte*)

Une formule de la conclusion d'une inférence qui n'est pas principale est dite **formule du contexte**, **contextuelle** *ou* **formule auxiliaire** *de la règle.*

Remarque 1.18

On remarque que pour toutes les règles de LK, si $\Gamma \vdash \Delta$ est conclusion d'une inférence r et que $F \in \Gamma \vdash \Delta$ est une formule auxiliaire de $\Gamma \vdash \Delta$, alors il y a dans chaque prémisse $\Gamma' \vdash \Delta'$ de r au plus une occurrence de formule G correspondant à F, c'est-à-dire :

- *F et G sont des occurrences de la même formule;*
- *l'occurrence G provient de la recopie de tout ou partie des formules contextuelles de $\Gamma \vdash \Delta$ dans la prémisse $\Gamma' \vdash \Delta'$ et que cette occurrence correspond précisément à une copie de l'occurrence F.*

Par ailleurs, pour toute règle de LK, si $\Gamma \vdash \Delta$ est prémisse d'une règle d'inférence r, si $F \in \Gamma \vdash \Delta$, alors

- *soit F est descendante de la formule principale de r ;*
- *soit F correspond à une occurrence de formule auxiliaire de la conclusion de r ;*
- *soit r est une règle de coupure et F est une formule de coupure.*

La notion d'occurrence fille d'une autre formule permet de relier les occurrences de formules apparaissant dans les divers séquents de la preuve.

Définition 1.19 (*Sous-occurrence*)

Soit $\mathcal{D}$ une preuve d'un séquent $\Gamma \vdash \Delta$ et soit $\Gamma' \vdash \Delta'$ un séquent de $\mathcal{D}$. $G \in \Gamma' \vdash \Delta'$ est une **sous-occurence** *de $F \in \Gamma \vdash \Delta$ si :*

- *soit $\Gamma' \vdash \Delta'$ est la racine de $\mathcal{D}$ et G et F sont la même occurrence de formule ;*
- *soit $\Gamma \vdash \Delta$ est conclusion d'une inférence r, $\Gamma' \vdash \Delta'$ appartient au sous-arbre $\mathcal{D}_0$ enraciné en une prémisse $\Gamma_0 \vdash \Delta_0$ de r, il existe $F_0 \in \Gamma_0 \vdash \Delta_0$ telle que G est une sous-occurrence de F_0 et :*
  - *F est formule principale de $\Gamma \vdash \Delta$ et F_0 est une occurrence fille de F ;*
  - *F est une formule auxiliaire de $\Gamma \vdash \Delta$ et F_0 est l'occurrence de formule de $\Gamma_0 \vdash \Delta_0$ correspondant à F.*

Définition 1.20 (*Formule traçable jusqu'à un séquent*)

Soit $\mathcal{D}$ une preuve de $\Gamma \vdash \Delta$. On dit que $F \in \Gamma \vdash \Delta$ est **traçable** *jusqu'à un séquent $\Gamma' \vdash \Delta'$ de $\mathcal{D}$ si ce séquent contient une sous-occurrence de F.*

Remarque 1.21

La possibilité d'appliquer une inférence de LK à une séquent $\Gamma \vdash \Delta$, une fois la formule principale choisie, ne dépend jamais de la **structure des formules auxiliaires** *de la règle qu'on souhaite appliquer à l'exception des règles $R\forall$ et $L\exists$ (en revanche, l'applicabilité dépend éventuellement du nombre de formules du contexte, comme dans le cas de la règle initiale qui demande qu'il n'y ait aucune formule contextuelle).*

On peut donner une autre présentation du calcul des séquents dans lequel les séquents possède également une signature $\Sigma : \Gamma \vdash \Delta$ où la signature permet d'enregistrer les variables qui sont susceptibles d'apparaître dans le séquent (c'est-à-dire qu'on suppose l'invariant $FV(\Gamma, \Delta) \subseteq \Sigma$ vérifié pour chaque séquent d'une preuve) et on n'a plus besoin de vérifier les variables libres des formules contextuelles mais seulement de vérifier la signature et, par la même occasion, d'étendre cette signature.

Quelques théorèmes formels de LK :

– $A \vee B \vdash B \vee A$	Commutativité de la disjonction
– $\vdash A \vee \neg A$	Tiers exclu (*Tertium non datur*)
– $\vdash ((A \Rightarrow B) \Rightarrow A) \Rightarrow A$	Loi de Peirce
– $\vdash (\neg\neg A) \Rightarrow A$	Élimination de la Double Négation
– $\vdash \exists x \forall y (P(x) \Rightarrow P(y))$	La Propriété du Buveur
– $A \vee B \vdash \neg(\neg A \wedge \neg B)$	Une Instance des lois de de Morgan
– $\vdash (A \Rightarrow B) \vee (B \Rightarrow A)$	
– $\vdash \neg\neg(A \vee \neg A)$	*Tiers exclu* "intuitionniste"
– $(p \Rightarrow q) \vdash (\neg q \Rightarrow \neg p)$	
– $(\neg q \Rightarrow \neg p) \vdash (p \Rightarrow q)$	

On donne en figure 1.4 quelques exemples de preuves de LK.

1.1.3 Admissibilité de la coupure et élimination des coupures

L'une des propriétés fondamentales du calcul des séquents établie par Gentzen est relative à la règle de coupure :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A}\,ini}{\vdash A, \neg A}\,R\neg}{\vdash A, A \vee \neg A}\,R\vee 2}{\vdash A \vee \neg A, A \vee \neg A}\,R\vee 1}{\vdash A \vee \neg A}\,RC \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A}\,ini}{A \vdash B, A}\,RW}{\vdash A \Rightarrow B, A}\,R\Rightarrow \quad \dfrac{}{A \vdash A}\,ini}{(A \Rightarrow B) \Rightarrow A \vdash A, A}\,L\Rightarrow}{(A \Rightarrow B) \Rightarrow A \vdash A}\,RC}{\vdash ((A \Rightarrow B) \Rightarrow A) \Rightarrow A}\,R\Rightarrow$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A}\,ini}{A \vdash A \vee \neg A}\,R\vee 1}{\neg(A \vee \neg A), A \vdash}\,L\neg}{\neg(A \vee \neg A) \vdash \neg A}\,R\neg}{\neg(A \vee \neg A) \vdash A \vee \neg A}\,R\vee 2}{\neg(A \vee \neg A), \neg(A \vee \neg A) \vdash}\,L\neg}{\neg(A \vee \neg A) \vdash}\,LC}{\vdash \neg\neg(A \vee \neg A)}\,R\neg$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{P(y) \vdash P(y)}\,ini}{P(y) \vdash P(y), P(z)}\,RW}{P(t), P(y) \vdash P(y), P(z)}\,LW}{P(t) \vdash P(y), P(y) \Rightarrow P(z)}\,R\Rightarrow}{P(t) \vdash P(y), \forall z.(P(y) \Rightarrow P(z))}\,R\forall}{P(t) \vdash P(y), \exists x.\forall y.(P(x) \Rightarrow P(y))}\,R\exists}{\vdash P(t) \Rightarrow P(y), \exists x.\forall y.(P(x) \Rightarrow P(y))}\,R\Rightarrow}{\vdash \forall y.(P(t) \Rightarrow P(y)), \exists x.\forall y.(P(x) \Rightarrow P(y))}\,R\forall}{\vdash \exists x.\forall y.(P(x) \Rightarrow P(y)), \exists x.\forall y.(P(x) \Rightarrow P(y))}\,R\exists}{\vdash \exists x.\forall y.(P(x) \Rightarrow P(y))}\,RC$$

FIG. 1.4 – Quelques preuves de LK.

Théorème 1.22 (*Théorème fondamental de Gentzen, ou Hauptsatz*)

La règle de coupure est admissible dans LK, c'est-à-dire que :

$$\Gamma \vdash_{LK} \Delta \qquad \Longleftrightarrow \qquad \Gamma \vdash_{LK \setminus Coupure} \Delta$$

Le résultat de Gentzen est en fait plus que cela : non seulement il établit que les calculs des séquents classiques avec ou sans règle de coupure ont le même pouvoir déductif (ce qui établit l'admissibilité de la règle de coupure), mais il donne également une procédure pour transformer une preuve du système avec coupure en une preuve sans coupure du même séquent (c'est l'élimination des coupures).

La première conséquence de l'admissibilité de la coupure est la cohérence du calcul des séquents LK :

Théorème 1.23 (*Cohérence de LK*)

On ne peut démontrer à la fois une formule et sa négation dans LK.

En effet, si A est une formule, et qu'on dispose de preuves $\mathcal{D}_1$ et $\mathcal{D}_2$ de $\vdash A$ et $\vdash \neg A$, alors en appliquant une règle $L\neg$ à la conclusion de $\mathcal{D}_1$ puis en coupant les deux preuves, on obtient une preuve $\mathcal{D}$ du séquent vide, mais par l'admissibilité de la coupure, il existerait une preuve sans coupure du séquent vide or on vérifie immédiatement qu'aucune règle de LK ne peut s'appliquer au séquent vide à part la coupure, ce qui assure la cohérence de LK.

Propriété de la sous-formule. L'admissibilité de la coupure est également essentielle en cela qu'elle a pour conséquence la propriété de la sous-formule. En effet, au moins dans le calcul du premier ordre, la règle de coupure est la seule qui fasse intervenir dans les prémisses d'une inférence des formules qui ne soient pas des sous-formules des formules du séquent conclusion : toutes les règles de LK sont ***analytiques*** à part la coupure. On déduit alors immédiatement du Hauptsatz la propriété de sous-formule :

Théorème 1.24 (*Propriété de la sous-formule*)

Un séquent prouvable peut être prouvé en n'utilisant que des sous-formules des formules apparaissant dans le séquent.

En effet, une preuve sans coupure ne fait intervenir que des sous-formules du séquent racine. Ce résultat est essentiel en recherche de preuve puisqu'il diminue considérablement l'espace de recherche de preuve.

Non-déterminisme de LK**.** On présente maintenant le contre-exemple de Lafont qui illustre le non-déterminisme de l'élimination des coupures de LK.

Considérons la preuve suivante :

$$\mathcal{D} = \cfrac{\cfrac{\cfrac{\mathcal{D}_1}{\vdash \Gamma}}{\vdash \Gamma, A}\, W \qquad \cfrac{\cfrac{\mathcal{D}_2}{\vdash \Delta}}{\vdash \Delta, \neg A}\, W}{\vdash \Gamma, \Delta}\, coupure$$

L'élimination des coupures de cette preuve peut se réduire, au choix, sur l'une des deux preuves suivantes qui n'ont rien à voir l'une avec l'autre :

$$\mathcal{D}'_1 = \cfrac{\cfrac{\mathcal{D}_1}{\vdash \Gamma}}{\vdash \Gamma, \Delta}\, W^\star \qquad \text{ou} \qquad \mathcal{D}'_2 = \cfrac{\cfrac{\mathcal{D}_2}{\vdash \Delta}}{\vdash \Gamma, \Delta}\, W^\star$$

Il existe un autre non-déterminisme dans l'élimination des coupures de LK, nettement moins problématique et plus dû à la structure du calcul des séquents qu'au système classique lui-même. Nous discuterons ce point lorsque nous introduirons les séquents linéaires en section 1.5.

1.1.4 Séquents dissymétriques

Quand on considère le système LK, on constate une très grande symétrie entre des règles gauche et des règles droite : les règles structurelles gauche et droite, les règles de la conjonction et les règles de la disjonction, les règles des quantificateurs universels et existentiels, etc... Cela nous amène à considérer un système dérivé de LK dans lequel on considère des séquents dissymétriques, n'ayant des formules que d'un côté du symbole $\vdash$, c'est-à-dire de la forme : $\vdash' \Gamma$. Le calcul perd sa symétrie mais est beaucoup plus simple. Dans ce système, l'implication devient un connecteur défini : $A \Rightarrow B \equiv \neg A \vee B$ et la négation n'apparaît qu'au niveau des formules atomiques, c'est-à-dire qu'en notant $\neg A$, on désignera la ***forme normale de de Morgan*** de la formule $\neg A$ qui est définie par le système de réécriture suivant, évidemment confluent et terminant :

$\neg(A \vee B) \to (\neg A \wedge \neg B)$	$\neg\forall x A \to \exists x \neg A$	$\neg\neg A \to A$
$\neg(A \wedge B) \to (\neg A \vee \neg B)$	$\neg\exists x A \to \forall x \neg A$	$A \Rightarrow B \to \neg A \vee B$

On est maintenant à même de définir le système LK' de la figure 1.5

Groupe Identité :

$$\frac{}{\vdash' A, \neg A}\ \textit{Axiome} \qquad\qquad \frac{\vdash' A, \Gamma \quad \vdash' \neg A, \Delta}{\vdash' \Gamma, \Delta}\ \textit{Coupure}$$

Groupe Structurel :

$$\frac{\vdash' \Gamma, B, A, \Delta}{\vdash' \Gamma, A, B, \Delta}\ Ex \qquad\qquad \frac{\vdash' \Gamma}{\vdash' A, \Gamma}\ W \qquad\qquad \frac{\vdash' A, A, \Gamma}{\vdash' A, \Gamma}\ C$$

Groupe Logique :

$$\frac{\vdash' A, \Gamma \quad \vdash' B, \Gamma}{\vdash' A \wedge B, \Gamma}\ \wedge \qquad\qquad \frac{\vdash' A, \Gamma}{\vdash' A \vee B, \Gamma}\ \vee 1 \qquad\qquad \frac{\vdash' B, \Gamma}{\vdash' A \vee B, \Gamma}\ \vee 2$$

$$\frac{\vdash' A, \Gamma}{\vdash' \forall x A, \Gamma}\ \forall \quad (*) \qquad\qquad \frac{\vdash' A[t/x], \Gamma}{\vdash' \exists x A, \Gamma}\ \exists$$

(*) Pour cette règle, $x \notin FV(\Gamma)$.

FIG. 1.5 – Calcul des séquents classique avec séquents à un côté.

1.1.5 Présentation alternative des règles pour $\wedge$ et $\vee$

On peut donner des présentations alternatives de règles d'inférence pour $\wedge$ et $\vee$ (dans le calcul original ou dans le calcul à un côté) qui est équivalente en terme de prouvabilité à la première présentation :

$$\frac{\vdash' A, \Delta_1 \quad \vdash' B, \Delta_2}{\vdash' A \wedge B, \Delta_1, \Delta_2}\ \wedge_m \qquad\qquad \frac{\vdash' A, B, \Delta}{\vdash' A \vee B, \Delta}\ \vee_m$$

Les nouvelles règles sont appelées ***règles multiplicatives*** tandis que les règles de la présentation originale sont appelées ***additives***, le calcul des séquents original sera appelé calcul des séquents additif et noté LK_a tandis que le calcul où les règles pour la disjonction et la conjonction ont été remplacées par les règles multiplicatives sera appelé calcul des séquents multiplicatif et noté LK_m. Pour établir l'équivalence entre ces deux systèmes, l'utilisation des règles structurelles est essentielle. Dans la suite de ce paragraphe, on notera $\wedge_a$, $\vee_a 1$ et $\vee_a 2$ les formes originales des règles d'inférence.

Proposition 1.25 (*Équivalence de LK_a et LK_m*)

Soit $\vdash' \Gamma$ un séquent. On a l'équivalence suivante :

$$\vdash'_{LK_a} \Gamma \qquad \Longleftrightarrow \qquad \vdash'_{LK_m} \Gamma.$$

Démonstration : $\boxed{\Rightarrow}$ Soit $\mathcal{D}$ une preuve de $\vdash' \Gamma$ dans LK_a. On obtient une preuve de $\vdash' \Gamma$ dans LK_m en remplaçant les occurrences de $\wedge_a$, $\vee_a 1$ et $\vee_a 2$ par des suites d'inférences de LK_m ($C^\star$ représente plusieurs applications de la règle C) comme suit :

$$\frac{\vdash' \Gamma, F \quad \vdash' \Gamma, G}{\vdash' \Gamma, F \wedge G}\ \wedge_a \qquad \text{devient} \qquad \cfrac{\cfrac{\vdash' \Gamma, F \quad \vdash' \Gamma, G}{\vdash' \Gamma, \Gamma, F \wedge G}\ \wedge_m}{\vdash' \Gamma, F \wedge G}\ \vdots\, C^\star$$

$$\frac{\vdash' \Gamma, F_i}{\vdash' \Gamma, F_1 \vee F_2} \vee_a i \qquad \text{devient} \qquad \cfrac{\cfrac{\vdash' \Gamma, F_i}{\vdash' \Gamma, F_1, F_2} W}{\vdash' \Gamma, F_1 \vee F_2} \vee_m$$

$\boxed{\Leftarrow}$ Réciproquement, soit $\mathcal{D}$ une preuve de $\vdash' \Gamma$ dans LK_m. On obtient une preuve de $\vdash' \Gamma$ dans LK_a en remplaçant les occurrences de $\wedge_m$ et $\vee_m$ par des suites d'inférences de LK_a ($W^\star$ représente plusieurs applications de la règle W) comme suit :

$$\frac{\vdash' \Gamma_1, F \quad \vdash' \Gamma_2, G}{\vdash' \Gamma_1, \Gamma_2, F \wedge G} \wedge_m \qquad \text{devient} \qquad \frac{\begin{array}{cc} \vdash' \Gamma_1, F & \vdash' \Gamma_2, G \\ \vdots\, W^\star & \vdots\, W^\star \\ \vdash' \Gamma_1, \Gamma_2, F & \vdash' \Gamma_1, \Gamma_2, G \end{array}}{\vdash' \Gamma_1, \Gamma_2, F \wedge G} \wedge_a$$

$$\frac{\vdash' \Gamma, F_1, F_2}{\vdash' \Gamma, F_1 \vee F_2} \vee_m \qquad \text{devient} \qquad \cfrac{\cfrac{\cfrac{\vdash' \Gamma, F_1, F_2}{\vdash' \Gamma, F_1, F_1 \vee F_2} \vee_a 2}{\vdash' \Gamma, F_1 \vee F_2, F_1 \vee F_2} \vee_a 1}{\vdash' \Gamma, F_1 \vee F_2} C$$

□

L'élément essentiel de la preuve précédente est la possibilité d'appliquer les règles d'affaiblissement et de contraction qui sont toutes deux nécessaires dans chaque direction. On remarque par ailleurs que la preuve indique en fait que l'on peut prendre n'importe quelle paire de règles pour $\wedge$ et de règles pour $\vee$.

On remarquera par ailleurs que si l'on prend les règles $\vee_m$ et $\wedge_a$, alors pour chaque formule de la forme $A \wedge B$ ou $A \vee B$ dans un séquent, il y a exactement une règle logique qui peut être appliquée, alors qu'avec les règles $\vee_a$ et $\wedge_m$ on avait plusieurs règles disponibles ($\vee_a 1$ ou $\vee_a 2$ d'un côté, et le choix de la partition du contexte Γ_1, Γ_2 de l'autre).

1.1.6 Permutabilité des règles d'inférence à la Kleene-Curry

Le calcul des séquents considère des preuves formelles qui sont des arbres de séquents et de règles d'inférence. Une preuve fournit donc un ordre partiel sur des occurrences de règles d'inférence. Cet ordre partiel est bien sûr compatible avec la structure de sous-formules[1], mais il y a bien plus de séquentialité encore dans une preuve en calcul des séquents du fait de la structure arborescente de ces objets.

Il est naturel de se demander dans quels cas on peut intervertir des règles d'inférence apparaissant dans une preuve de LK : à quelles conditions, lorsqu'un séquent contenant des formules A et B est prouvable en appliquant d'abord (de bas en haut) une inférence ayant A pour formule principale puis une inférence ayant B pour formule principale, peut-on trouver une autre preuve du même séquent qui est égale à la première à l'ordre près des deux inférences sus-mentionnées ?

Cette question est à l'origine de nombreuses questions en théorie de la démonstration, tant du point de vue de la construction de preuves que du point de vue de la question de l'identité de preuves. Les premiers à avoir mené une étude approfondie de cette question de la permutabilité des inférences en calcul des séquents sont certainement Kleene [Kle52] et Curry [Cur52][2].

[1] Puisqu'on ne fait pas d'inférence profonde comme en calcul des structures [Gug07] par exemple.

[2] Si Curry semble avoir présenté ses premiers résultats antérieurement à Kleene (il a donné une conférence en décembre 1948 lors d'une réunion de l'*Association for Symbolic Logic* et de l'*American Mathematical Society*), leurs travaux sont indépendants et les résultats de Kleene sur le sujet sont plus développés et systématiquement présentés que ceux de Curry. Nous ferons dans la suite référence aux travaux de Kleene-Curry sur la permutabilité.

Nous donnons ici quelques uns des résultats présentés par Kleene [Kle52] qui ont influencé les travaux sur la prouvabilité uniforme [MNPS91] ; nous étudierons plus loin le lien entre ces résultats et les résultats de focalisation en logique linéaire d'Andreoli.

Nous nous plaçons dans le calcul des séquents classique avec la règle de disjonction multiplicative à la place de la règle additive (voir la section 1.1.5) et nous considérons des séquents à un seul côté (voir la section 1.1.4) ce qui ne change pas les résultats mais diminue le nombre de cas à traiter puisqu'il y a moins de règles d'inférence. Nous commençons par définir la notion de règles d'inférence adjacentes, qui sont les inférences susceptibles d'être interverties :

Définition 1.26 (*Inférences adjacentes*)

Soit $\mathcal{D}$ une preuve en calcul des séquents. Deux règles d'inférence ayant une occurrence dans $\mathcal{D}$ sont dites **adjacentes** *lorsqu'elles ne sont séparées que par des règles structurelles et qu'elles s'appliquent à des formules qui sont descendantes de deux formules distinctes d'un séquent de $\mathcal{D}$ (c'est-à-dire qu'elles ne sont pas sous-formules l'une de l'autre).*

Exemple 1.27

Les deux occurences étoilées des règles d'inférence $\wedge$ et $\vee$ sont adjacentes dans la preuve ci-dessous :

$$\mathcal{D} = \cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\mathcal{E}_1}{\vdash \Gamma, A, C, D} \quad \cfrac{\mathcal{E}_2}{\vdash \Gamma, B, C, D}}{\vdash \Gamma, A \wedge B, C, D}\ \wedge^\star}{\vdash \Gamma, A \wedge B, C, D, E}\ W}{\vdash \Gamma, A \wedge B, C \vee D, E}\ \vee^\star}{\vdash \Gamma, (A \wedge B) \vee (C \vee D), E}\ \vee$$

L'***interversion*** ou la ***permutation*** de deux règles d'inférences adjacentes, r_1 et r_2, consiste, à partir d'une preuve $\mathcal{D}$ de séquent conclusion $\vdash \Gamma$ à laquelle s'applique une inférence r_1 dont l'une des prémisses est au-dessous de la conclusion d'une inférence r_2 (les deux règles étant adjacentes), à construire une nouvelle preuve $\mathcal{D}'$ dans laquelle l'ordre des deux inférences a été échangé, c'est-à-dire que r_2 est maintenant au-dessous de r_1 et que le reste de la preuve est inchangé :

$$\dfrac{\dfrac{\dfrac{\mathcal{E}'}{\vdash\Gamma'}\quad\dfrac{\mathcal{E}''}{\vdash\Gamma''}}{\vdash\Gamma_1}\,r_2}{\vdash\Gamma}\,r_1 \longrightarrow \dfrac{\dfrac{\dfrac{\mathcal{E}'}{\vdash\Gamma'}}{\vdash\Gamma'_1}\,r_1\quad\dfrac{\mathcal{E}''}{\vdash\Gamma''}}{\vdash\Gamma}\,r_2$$

$$\dfrac{\dfrac{\dfrac{\mathcal{E}'}{\vdash\Gamma'}\quad\dfrac{\mathcal{E}''}{\vdash\Gamma''}}{\vdash\Gamma_1}\,r_2}{\vdash\Gamma}\,r_1 \longrightarrow \dfrac{\dfrac{\dfrac{\mathcal{E}'}{\vdash\Gamma'}}{\vdash\Gamma'_1}\,r_1\quad\dfrac{\dfrac{\mathcal{E}''}{\vdash\Gamma''}}{\vdash\Gamma''_1}\,r_1}{\vdash\Gamma}\,r_2$$

$$\dfrac{\dfrac{\dfrac{\mathcal{E}}{\vdash\Gamma'}}{\vdash\Gamma_1}\,r_2}{\vdash\Gamma}\,r_1 \longrightarrow \dfrac{\dfrac{\dfrac{\mathcal{E}}{\vdash\Gamma'}}{\vdash\Gamma_2}\,r_1}{\vdash\Gamma}\,r_2$$

$$\dfrac{\dfrac{\dfrac{\mathcal{E}'_1}{\vdash\Gamma'_1}\quad\dfrac{\mathcal{E}'_2}{\vdash\Gamma'_2}}{\vdash\Gamma_3}\,r_1\quad\dfrac{\mathcal{E}''}{\vdash\Gamma''}}{\vdash\Gamma}\,r_2 \longrightarrow \dfrac{\dfrac{\dfrac{\mathcal{E}'_1}{\vdash\Gamma'_1}\quad\dfrac{\mathcal{E}''}{\vdash\Gamma''}}{\vdash\Gamma_1}\,r_2\quad\dfrac{\dfrac{\mathcal{E}'_2}{\vdash\Gamma'_2}\quad\dfrac{\mathcal{E}''}{\vdash\Gamma''}}{\vdash\Gamma_2}\,r_2}{\vdash\Gamma}\,r_1$$

FIG. 1.6 – Divers schémas de permutation.

Définition 1.28 (*Permutabilité d'inférences logiques*)

On dit que deux règles (du fragment logique) R_1 et R_2 sont **permutables** *ou peuvent être* **interverties** *si pour toute preuve $\mathcal{D}$ et toutes occurrences r_1 de R_1 et r_2 de R_2 de conclusions respectives $\mathcal{S}_1$ et $\mathcal{S}_2$, telles que :*

- *r_1 et r_2 sont adjacentes dans $\mathcal{D}$, r_1 étant au-dessus de r_2 ;*
- *la formule principale de r_1 n'est pas une sous-occurrence de la formule principale de r_2 ;*
- *les prémisses de r_1 forment un ensemble de séquents $\mathcal{R}_1$ et les prémisses de r_2 un ensemble de séquents $\mathcal{S} \cup \mathcal{R}_2$ où $\mathcal{S}$ est telle que $\mathcal{S}_1$ est dans la sous-preuve enracinée en $\mathcal{S}$ (autrement dit, la formule principale de r_1 est sous-occurrence d'une formule de $\mathcal{S}$) ;*

il existe une preuve $\mathcal{D}'$ qui ne diffère de $\mathcal{D}$ qu'en remplaçant le sous-arbre de $\mathcal{D}$ enraciné en $\mathcal{S}_2$ dont les feuilles sont les séquents $\mathcal{R}_1 \cup \mathcal{R}_2$ par une déduction dans laquelle $\mathcal{S}_2$ est conclusion d'un certain nombre de règles structurelles puis d'une règle R_1 au-dessus de laquelle sont éventuellement appliquées des règles structurelles au-dessus desquelles on trouve des occurrences des règles R_2 (zéro ou plusieurs règles).

Remarque 1.29

Cette notion devrait être précisée dans certains cas :

- *lorsqu'il y a plusieurs prémisses à la règle r_2 car cela peut conduire à une duplication de la règle r_1 voire à une duplication d'une partie de la preuve ;*
- *lorsque r_1 a plusieurs prémisses la permutation peut éventuellement conduire à identifier deux instances de la règle r_2, voire deux portions de preuve ;*
- *si r_2 est sans prémisse, cela peut entraîner l'effacement de r_1.*

Nous avons volontairement donné une définition relativement imprécise de la permutation de règles d'inférence (et en tout cas relativement abstraite) car nous voudrons l'adapter au calcul des séquents de la logique linéaire au chapitre 9.

Divers schémas de permutation (qui ne correspondent pas forcément aux permutations de LK) sont représentés en figure 1.6.

Ainsi, l'interversion des deux connecteurs adjacents dans la preuve de l'exemple 1.27 produit la preuve :

$$\mathcal{D}' = \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{E}_1}{\vdash \Gamma, A, C, D}}{\vdash \Gamma, A, C \vee D}\,\vee^\star}{\vdash \Gamma, A, C \vee D, E}\,W \qquad \dfrac{\dfrac{\dfrac{\mathcal{E}_2}{\vdash \Gamma, B, C, D}}{\vdash \Gamma, B, C \vee D}\,\vee^\star}{\vdash \Gamma, B, C \vee D, E}\,W}{\vdash \Gamma, A \wedge B, C \vee D, E}\,\wedge^\star}{\vdash \Gamma, (A \wedge B) \vee (C \vee D), E}\,\vee$$

où l'on constate qu'on a dupliqué la règle $\vee^\star$.

Le théorème suivant est une adaptation d'un théorème dû à Kleene [Kle52] au calcul des séquents à un seul côté :

Théorème 1.30 (*Permutabilité des règles d'inférence dans LK [Kle52]*)

Soit $\Gamma \vdash \Delta$ un séquent dans lequel aucune variable n'apparaît à la fois libre et liée. Considérons une classification des occurrences de connecteurs logiques apparaissant dans $\Gamma \vdash \Delta$ en k ensembles $\mathcal{C}_1, \ldots \mathcal{C}_k$ vérifiant :

- *si $c_i \in \mathcal{C}_i$ est un connecteur dans la portée de $c_j \in \mathcal{C}_j$, alors $i \leq j$;*
- *si $c_i = \exists \in \mathcal{C}_i$ et $c_j = \forall \in \mathcal{C}_j$, alors $i \leq j$, sauf si c_j est dans la portée de c_i.*

Dans ce cas, toute preuve de $\Gamma \vdash \Delta$ peut être réarrangée par permutation de règles d'inférence (et sans changer son séquent conclusion) de telle sorte que le long de n'importe qu'elle branche de la preuve, les inférences correspondant à des connecteurs de $\mathcal{C}_i$ apparaissent au-dessous des inférences correspondant à des connecteurs de $\mathcal{C}_j$ si $i < j$.

Les deux conditions sur les connecteurs dans l'énoncé du théorème sont nécessaires. Pour la première condition, il est bien évident qu'on ne peut pas s'en passer. Pour la seconde condition, son caractère nécessaire est moins évident, mais la preuve du théorème du buveur donnée en figure 1.4 est un exemple où la permutation entre le $\forall$ et le $\exists$ est impossible.

Le théorème 1.30 a pour conséquence immédiate le théorème du séquent médian de Gentzen [Gen69] qui affirme que tout séquent constitué de formules prénexes a une preuve où toutes les règles sur les quantificateurs sont appliquées au-dessous des règles sur les connecteurs propositionnels[3].

1.2. Complétude de *LK*

> « Whitehead and Russell, as is well-known, constructed logic and mathematics by initially taking certain evident propositions as axioms and deriving the theorems of logic and mathematics from these by means of some precisely formulated principles of inference (...). Of course, when such a procedure is followed the question at once arises whether the initially postulated system of axioms and principles of inference is complete, that is, whether it actually suffices for the derivation of *every* logico-mathematical proposition. »

[3] La preuve de Gentzen [Gen69] du théorème du séquent médian utilise déjà une sorte de permutation d'inférence, mais pas dans un sens aussi élaboré que chez Kleene. Gentzen fait plutôt reposer sa preuve sur des propriétés de la contraction et de l'affaiblissement que sur une vraie permutation d'inférence comme défini ci-dessus. En particulier, la notion d'occurrence de formule ne semble pas tout à fait claire dans cette preuve de Gentzen.
On peut ensuite déduire facilement le théorème de Herbrand du théorème du séquent médian [Gir87b].

Ainsi débute l'article de Gödel démontrant le théorème de complétude du calcul des prédicats [Gö30, Hei67]. En effet, pour un système déductif, deux questions sont importantes vis-à-vis d'une sémantique :

- Tous les énoncés qui peuvent être prouvés sont-ils vrais ?
- Tous les énoncés vrais sont-ils prouvables ?

Ces deux questions correspondent aux propriétés de ***correction*** et de ***complétude*** d'un système de démonstration. Il faut à vrai dire être plus précis et parler de validité plutôt que de vérité, la validité d'un énoncé étant sa «vérité dans tout modèle». La correction de LK ne pose pas de problème : grâce à l'admissibilité de la coupure, il suffit de vérifier que chaque règle du système sans coupure préserve la validité. La complétude est en revanche beaucoup plus complexe. Ceci n'est pas surprenant puisque, alors qu'on peut approcher la notion de prouvabilité de manière structurée et inductive, analytique, grâce au résultat d'admissibilité de la coupure, la notion de validité de son côté est moins facilement sujette à l'analyse.

La complétude du calcul des prédicats a été initialement démontrée par Gödel en 1930 [Gö30, Hei67]. Nous allons ici nous intéresser à la complétude de LK, en considérant le calcul sans coupure et dans sa présentation avec seulement des formules à droite du $\vdash$.

On dira d'un séquent $\vdash \Gamma$ qu'il est ***valide*** si, pour tout modèle $\mathcal{M}$, l'une des formules de Γ est interprétée par la valeur de vérité v.

La complétude de LK s'énonce alors comme suit :

Théorème 1.31 (*Complétude de LK*)

Si un séquent $\vdash \Gamma$ est valide alors il est prouvable dans LK sans coupure.

On prouvera en fait la contraposée de l'énoncé du théorème, c'est-à-dire que si un séquent $\vdash \Gamma$ n'est pas prouvable, alors il existe un modèle du langage $\mathcal{L}$ dans lequel aucune formule de Γ n'est vraie.

Nous allons utiliser une technique de recherche de preuve qui va consister à considérer une méthode de construction de preuve qui soit complète dans le sens où, si le séquent est prouvable, notre méthode trouve nécessairement une preuve. On utilisera ensuite le fait que, si $\vdash \Gamma$ est un séquent non-prouvable dans LK, on obtient par notre méthode un objet (qui n'est donc pas une preuve dans LK) permettant de construire un contre-modèle de $\vdash \Gamma$.

Ce type de technique pour établir la complétude a notamment été proposé par Smullyan [Smu68].

Définition 1.32 (*Séquents structurés*)

On considère des séquents de la forme : $L; \Gamma; \Phi$, où

- *L est un ensemble de littéraux (formules atomiques ou négations de formules atomiques),*
- *Γ est une liste ordonnée de formules quelconques et*
- *Φ est une liste ordonnée de formules quantifiées existentiellement.*

On dit que le séquent $L; \Gamma; \Phi$ est **satisfaisable** *si le séquent (de LK) $\vdash L, \Gamma, \Phi$ est satisfaisable (on ne se préoccupe alors plus de l'ordre des formules de Γ et Φ).*

Règles Impropres :

$$\frac{\boxed{\text{Open}}}{L; \Gamma; \Phi}\ Open \qquad \frac{\boxed{\text{Failure}}}{L; \emptyset; \emptyset}\ Failure$$

Règles de Classification :

$$\frac{l, L; \Gamma; \Phi}{L; l, \Gamma; \Phi}\ \mathcal{C}_{litt} \qquad \frac{L; \Gamma; \Phi, \exists x.A}{L; \exists x.A, \Gamma; \Phi}\ \mathcal{C}_{\exists}$$

Règles Logiques :

$$\frac{}{A, \neg A, L; \Gamma; \Phi}\ ini \qquad \frac{}{L; \top, \Gamma; \Phi}\ \top$$

$$\frac{L; A, \Gamma; \Phi \quad L; B, \Gamma; \Phi}{L; A \wedge B, \Gamma; \Phi}\ \wedge \qquad \frac{L; A, B, \Gamma, \Phi}{L; A \vee B, \Gamma, \Phi}\ \vee$$

$$\frac{L; A[y/x], \Gamma; \Phi}{L; \forall x.A, \Gamma; \Phi}\ \forall \qquad \frac{L; A[t/x]; \Phi, \exists x.A}{L; \emptyset; \exists x.A, \Phi}\ \exists$$

FIG. 1.7 – Arbres de dérivation systématiques. *Dans la règle $\forall$, la variable y n'apparaît pas libre dans L, Γ, Φ.*

Définition 1.33 (*Arbre de Dérivation Systématique*)

On considère une énumération ϵ des termes du langage du premier ordre considéré.

Un **arbre de dérivation systématique** *pour le séquent $L; \Gamma; \Phi$ est un arbre de dérivation enraciné en $L; \Gamma; \Phi$ dont les règles d'inférence (à branchement fini) sont données en figure 1.7 et qui est tel que :*

- *une branche d'un arbre de dérivation systématique peut être infiniment longue ;*
- *la règle* **ini** *est appliquée dès que possible : dès qu'un arbre de dérivation systématique contient un séquent de la forme $A, \neg A, L'; \Gamma'; \Phi'$, ce séquent doit être conclusion d'une règle* **open** *ou d'une règle* **ini**
- *quand une règle $\exists$ est utilisée, la variable doit être instanciée par le premier terme de l'énumération ϵ qui n'a pas été utilisé plus bas dans la dérivation pour une instanciation de ce quantificateur.*

Un arbre de dérivation systématique est dit **maximal** *ou* **clos** *lorsqu'il ne contient pas d'occurrence de la règle open.*

Un arbre de dérivation systématique est un **arbre échec** *si il contient une occurrence de la règle* **failure** *ou bien une branche infinie. Dans le cas contraire, on parle d'***arbre succès***.*

Remarque 1.34

Dans le cadre d'une énumération ϵ, on pourrait reformuler comme suit les règles concernant les exponentielles, en supposant que la composante Φ des séquents contient maintenant des formules exponentielles indexées :

$$\frac{L; \Gamma; \Phi, \exists^0 x.A}{L; \exists x.A, \Gamma; \Phi}\ \mathcal{C}_{\exists^i} \qquad \frac{L; A[\epsilon(k)/x]; \Phi, \exists^{k+1} x.A}{L;\ \cdot\ ; \exists^k x.A, \Phi}\ \exists^i$$

$$
\begin{array}{cl}
\hline
\neg P(t_k), P(y_{k-1}), \ldots, \neg P(t_1), P(y_0), \neg P(t_0) \;\;;\;\; P(y_k) \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) & ini \\
\hline
P(y_{k-1}), \ldots, \neg P(t_1), P(y_0), \neg P(t_0) \;\;;\;\; \neg P(t_k), P(y_k) \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) & \mathcal{C}_{litt} \\
\hline
 & \exists \\
\vdots & \\
\hline
P(y_1), \neg P(t_1), P(y_0), \neg P(t_0) \;\;;\;\; \emptyset \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) & \exists \\
\hline
\neg P(t_1), P(y_0), \neg P(t_0) \;\;;\;\; P(y_1) \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) & \mathcal{C}_{litt} \\
\hline
P(y_0), \neg P(t_0) \;\;;\;\; \neg P(t_1), P(y_1) \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) & \mathcal{C}_{litt} \\
\hline
P(y_0), \neg P(t_0) \;\;;\;\; (\neg P(t_1) \vee P(y_1)) \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) & \vee \\
\hline
P(y_0), \neg P(t_0) \;\;;\;\; \forall y.(\neg P(t_1) \vee P(y)) \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) & \forall \\
\hline
P(y_0), \neg P(t_0) \;\;;\;\; \emptyset \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) & \exists \\
\hline
\neg P(t_0) \;\;;\;\; P(y_0) \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) & \mathcal{C}_{litt} \\
\hline
\emptyset \;\;;\;\; \neg P(t_0), P(y_0) \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) & \mathcal{C}_{litt} \\
\hline
\emptyset \;\;;\;\; \neg P(t_0) \vee P(y_0) \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) & \vee \\
\hline
\emptyset \;\;;\;\; \forall y.(\neg P(t_0) \vee P(y)) \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) & \forall \\
\hline
\emptyset \;\;;\;\; \emptyset \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) & \exists \\
\hline
\emptyset \;\;;\;\; \exists x.\forall y.(\neg P(x) \vee P(y)) \;\;;\;\; \emptyset & \mathcal{C}_{\exists}
\end{array}
$$

FIG. 1.8 – Arbre de dérivation systématique pour la formule du buveur. *La règle initiale est appliquée lorsque pour la première fois un terme t_k est égal à l'une des variables y_i avec $i < k$.*

Remarque 1.35

Il y a principalement trois choses importantes à noter concernant les arbres de dérivation systématiques :

- *on a incorporé les règles structurelles de LK dans les règles logiques et dans l'axiome (la règle initiale fait l'affaiblissement, l'existentielle est précédée d'une contraction),*
- *on utilise toujours la version* **réversible** *de la règle logique quand une telle règle existe (c'est-à-dire la disjonction multiplicative et la conjonction additive) et*
- *dans la règle $\exists$, on prend la première formule de la liste de formules existentielles comme formule principale et puis la formule est reclassée à la fin de la liste, c'est-à-dire qu'on effectue une contraction sur une formule existentielle avant de lui appliquer une règle existentielle et que l'utilisation des règles existentielles est* **parcimonieuse** *(ces règles ne sont appliquées que lorsque la partie médiane du séquent est vide) et* **équitable** *(chaque formule existentielle dans le séquent se verra, à son tour, appliquer une règle d'inférence et on parcourra tout l'ensemble des termes du langage grâce à l'énumération).*

Nous prouvons maintenant le théorème de complétude. La preuve est constituée de deux parties : on commence par prouver que si un séquent $\vdash \Gamma$ n'est pas prouvable dans LK alors les arbres de dérivations systématiques clos de $\cdot;\Gamma;\cdot$ sont nécessairement des arbres échec. Nous construisons ensuite, à partir d'un arbre échec, un contre-modèle pour le séquent initialement considéré.

Démonstration : Considérons un séquent $\vdash \Gamma$ qui n'est pas prouvable dans LK (avec les séquents à un côté). Soit $\mathcal{T}$ un arbre de dérivation systématique pour le séquent $\vdash \emptyset;\Gamma;\emptyset$ avec l'énumération ϵ qui soit maximal (c'est-à-dire qu'il ne contient aucun nœud *open*).

Il s'agit forcément d'un arbre échec car dans le cas contraire on pourrait construire à partir de $\mathcal{T}$ une preuve de $\vdash \Gamma$ dans LK (ce qui est impossible par hypothèse) : On obtient en effet une preuve de $\vdash \Gamma$ dans LK à partir de $\mathcal{T}$ en transformant tous les $L;\Gamma;\Phi$ en $\vdash L,\Gamma,\Phi$, en effaçant toutes les règles de classification, en conservant les règles $\top$, $\vee$, $\wedge$ et $\forall$ telles qu'elles sont, en ajoutant le nombre approprié de règles d'affaiblissement avant une règle initiale et en effectuant une contraction avant chaque application d'une règle existentielle.

Puisque l'arbre considéré est un arbre-échec, alors il y a nécessairement au moins une branche échec (branche infinie ou se terminant sur un nœud $failure$), on considère une telle branche que l'on appellera β. Cette branche va nous permettre de construire un contre-modèle du séquent conclusion.

Nous allons donner des valeurs de vérité aux formules atomiques du langage d'après β, ce qui nous fournit un modèle, noté $\mathcal{H}_\beta$, comme suit : on peut interpréter les symboles de fonction comme des fonctions des termes dans les termes, la seule partie intéressante étant la valeur de vérité des formules atomiques qui nous indique comment interpréter les symboles de prédicat du langage. L'interprétation des formules atomiques est définie comme suit : si une formule atomique A apparaît dans la composante de gauche d'un séquent de β, elle prend la valeur de vérité f ; si sa négation apparaît dans la composante de gauche, A prend la valeur v. Les formules atomiques qui n'apparaissent dans aucun des séquents de la branche β prennent la valeur v (une valeur arbitraire conviendrait).

Ce procédé est bien défini puisque deux littéraux opposés ne peuvent pas apparaître dans la même branche échec : si A apparaît dans la partie gauche du séquent β_m de β et que $\neg A$ est dans la partie gauche de β_n, alors ils appartiennent tous les deux à la composante de gauche de $\beta_{max(m,n)}$ puisque l'on ne retire jamais de formule de la composante de gauche des séquents en montant le long d'une branche. Mais dans ce cas, la prochaine règle de β doit être une règle *axiome* par construction de l'arbre de dérivation systématique, ce qui est impossible sur une branche échec.

Un tel modèle est appelé ***modèle de Herbrand***.

On vérifie maintenant que le séquent conclusion de l'arbre est effectivement falsifié par ce modèle, en montrant que toute formule apparaissant dans un séquent de β est interprétée par f dans $\mathcal{H}_\beta$; puisque la racine fait bien sûr partie de cette branche, on pourra conclure. On raisonne par induction sur la structure de la formule :

- Si A est un littéral, alors par définition du modèle, elle est interprétée par f.
- A ne peut pas être $\top$ puisque cela impliquerait qu'une règle $\top$ est appliquée dans β mais alors β n'est pas une branche échec ;
- Si A est $B \wedge C$, alors, si A apparaît dans β, A finira par arriver en première position de la composante du milieu d'un séquent β_m. Le séquent en question est alors l'arc-sortant d'une règle $\wedge$ qui a deux arcs entrants, dont l'un des deux fait bien sûr partie de β. Supposons par exemple qu'il s'agit de l'arc de gauche ; dans ce cas, B est dans β_{m+1} et par hypothèse d'induction B est interprétée par f dans $\mathcal{H}_\beta$ et donc A est aussi interprétée par f dans $\mathcal{H}_\beta$;
- Si A est $B \vee C$ alors il existe β_m tel que A est en première position de la composante du milieu de β_m. La règle suivante qui est appliquée est donc $\vee$ produisant β_{m+1} qui contient à la fois B et C. Par hypothèse d'induction B et C sont fausses dans $\mathcal{H}_\beta$ et leur disjonction l'est aussi ;
- Si A est $\forall x.B$ alors la formule sera en première position d'un certain β_m. Sa prémisse β_{m+1} contient une formule $B[y/x]$ qui est interprétée par f par hypothèse d'induction et donc A est fausse dans $\mathcal{H}_\beta$;
- Si A est $\exists x.B$, alors cette formule sera finalement en première position du séquent du milieu et sera classifiée ensuite dans la composante de droite du séquent ce qui assure que la branche échec β est nécessairement infinie. On va donc, le long de la branche β, appliquer une infinité d'instances de la règle $\exists$ à la formule A et qui, grâce à ϵ, énumèreront tous les termes : pour chaque terme t, il existe un séquent β_t de β ayant $B[t/x]$ dans sa composante médiane et qui est donc, par hypothèse d'induction, interprétée par f dans $\mathcal{H}_\beta$ de telle sorte

$$\frac{}{A \vdash A}\ \textit{axiome} \qquad\qquad \frac{\Gamma_1 \vdash A \quad \Gamma_2, A \vdash \Xi}{\Gamma_1, \Gamma_2 \vdash \Xi}\ \textit{coupure}$$

FIG. 1.9 – Calcul des séquents LJ : Groupe Identité.

que A reçoit également la valeur f.

Nous avons donc prouvé que toute formule apparaissant dans la branche β était interprétée par f dans le modèle $\mathcal{H}_\beta$ de telle sorte que toute formule du séquent conclusion est fausse dans $\mathcal{H}_\beta$; ce séquent ne peut être valide.

□

1.3. Logique intuitionniste : LJ

1.3.1 Non-constructivisme de LK

Le calcul des séquents LK n'est pas constructif dans le sens que d'une preuve de $\vdash A \vee B$ on ne peut pas forcément extraire une preuve de $\vdash A$ ou de $\vdash B$ du fait de l'utilisation possible du tiers exclu comme dans la démonstration suivante :

Proposition 1.36

Il existe deux nombres irrationnels a, b tels que a^b est rationnel.

Démonstration : Considérons le nombre irrationnel $\sqrt{2}$. De deux choses l'une. Soit $\sqrt{2}^{\sqrt{2}}$ est rationnel, soit il est irrationnel.

Dans le premier cas, il suffit de prendre $a = b = \sqrt{2}$ tandis que dans l'autre cas, on pose $a = \sqrt{2}^{\sqrt{2}}$ et $b = \sqrt{2}$ et on obtient $a^b = (\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^2 = 2$ Puisqu'il n'y a pas de troisième possibilité, cela montre le résultat.

□

Cette preuve est particulière : même si la preuve est valide, on ne peut pas en extraire une information sur l'irrationalité de $\sqrt{2}^{\sqrt{2}}$ (qui est en effet irrationnel), ni sur une quelconque paire d'irrationnels vérifiant la propriété : on ne peut pas construire de témoin de ce que le théorème affirme.

Dans le système LK, cette propriété est due à la possibilité d'utiliser la contraction à droite qui permet de prouver le tiers exclu :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A}\ ini}{\vdash A, \neg A}\ R\neg}{\vdash A, A \vee \neg A}\ R\vee 2}{\vdash A \vee \neg A, A \vee \neg A}\ R\vee 1}{\vdash A \vee \neg A}\ RC$$

1.3.2 Le calcul des séquents LJ

Définition 1.37 (*Séquents intuitionnistes*)

Les séquents de LJ sont de la forme $\Gamma \vdash \Xi$ où Ξ contient au plus une formule.

$$\frac{\Gamma \vdash \Xi}{\Gamma, A \vdash \Xi}\ LW \qquad \frac{\Gamma \vdash}{\Gamma \vdash A}\ RW \qquad \frac{\Gamma, A, A \vdash \Xi}{\Gamma, A \vdash \Xi}\ LC$$

FIG. 1.10 – Calcul des séquents LJ : Groupe Structurel.

$$\frac{\Gamma \vdash A}{\Gamma, \neg A \vdash}\ L\neg \qquad \frac{\Gamma, A \vdash}{\Gamma \vdash \neg A}\ R\neg$$

$$\frac{\Gamma_1 \vdash A \quad \Gamma_2, B \vdash \Xi}{\Gamma_1, \Gamma_2, A \Rightarrow B \vdash \Xi}\ L\Rightarrow \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\ R\Rightarrow$$

$$\frac{\Gamma, A \vdash \Xi}{\Gamma, A \wedge B \vdash \Xi}\ L\wedge 1 \qquad \frac{\Gamma, B \vdash \Xi}{\Gamma, A \wedge B \vdash \Xi}\ L\wedge 2 \qquad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ R\wedge$$

$$\frac{\Gamma, A \vdash \Xi \quad \Gamma, B \vdash \Xi}{\Gamma, A \vee B \vdash \Xi}\ L\vee \qquad \frac{\Gamma \vdash A}{\Gamma \vdash A \vee B}\ R\vee 1 \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B}\ R\vee 2$$

$$\frac{\Gamma, A[t/x] \vdash \Xi}{\Gamma, \forall x.A \vdash \Xi}\ L\forall \qquad \frac{\Gamma \vdash A}{\Gamma \vdash \forall x.A}\ R\forall\ (*) \qquad \frac{\Gamma, A \vdash \Xi}{\Gamma, \exists x.A \vdash \Xi}\ L\exists\ (**) \qquad \frac{\Gamma \vdash A[t/x]}{\Gamma \vdash \exists x.A}\ R\exists$$

(*) $x \notin FV(\Gamma)$. (**) $x \notin FV(\Gamma, \Xi)$.

FIG. 1.11 – Calcul des séquents LJ : Groupe Logique.

Définition 1.38 (***Calcul des séquents LJ***)

Le calcul des séquents LJ est le système défini par les figures 1.9, 1.10 et 1.11.

De même que LK, LJ satisfait un théorème d'élimination des coupures et une propriété de la sous-formule mais la structure de LJ nous fournit des propriétés bien plus intéressantes qui témoignent du caractère constructif de LJ.

1.3.3 Propriétés d'existence et de la disjonction

L'élimination des coupures, nous donne :

Proposition 1.39 (***Propriété de la disjonction***)

Si $\vdash_{LJ} A \vee B$, *alors* $\vdash_{LJ} A$ *ou* $\vdash_{LJ} B$.

Remarque 1.40

La propriété de la disjonction nous assure en particulier qu'on ne peut pas prouver le tiers exclus dans LJ : si $\vdash a \vee \neg a$ *était prouvable dans LJ pour un littéral* a*, on pourrait prouver soit* $\vdash a$*, soit* $\vdash \neg a$*, c'est-à-dire* $a \vdash$ *ce qui n'est pas le cas puisqu'aucune règle si ce n'est la coupure et l'affaiblissement ne s'applique au premier séquent et aucune règle sinon la coupure, l'affaiblissement et la contraction ne s'applique au second séquent.*

Proposition 1.41 (***Propriété de l'existence***)

Si $\vdash_{LJ} \exists x.A$*, alors il existe un terme* t *tel que* $\vdash_{LJ} A[t/x]$.

Ces propriétés sont valables lorsque le séquent ne contient pas d'hypothèse, c'est-à-dire dans le cas d'une théorie vide, pas nécessairement quand il y a des formules à gauche comme le montre le cas du séquent $A \vee B \vdash A \vee B$.

Les théories de Harrop sont des ***théories constructives*** :

Définition 1.42 (*Théorie de Harrop*)

- *Une (occurrence d'une) sous-formule B de A est dite* **strictement positive** *si elle n'apparaît ni sous une négation ni à gauche d'une implication ;*
- *Une* **formule de Harrop** *est une formule A dont aucune sous-formule strictement positive n'est de la forme $B \vee C$ ou $\exists x.B$;*
- *Une* **théorie de Harrop** *est un ensemble de formules de Harrop.*

Proposition 1.43 (*Constructivité des théories de Harrop*)

Si Γ est une théorie de Harrop, on a les propriétés suivantes :
- *si $\Gamma \vdash_{LJ} A \vee B$ alors $\Gamma \vdash_{LJ} A$ ou $\Gamma \vdash_{LJ} B$;*
- *si $\Gamma \vdash_{LJ} \exists x A$ alors il existe un terme t tel que $\Gamma \vdash_{LJ} A[t/x]$.*

1.3.4 Comparaison de la prouvabilité en LJ et en LK

La prouvabilité dans LJ est évidemment plus faible que la prouvabilité en LK puisque toute preuve de LJ est a fortiori une preuve de LK. Les relations entre la prouvabilité dans LJ et dans LK sont pourtant plus fines que cette simple observation.

En LJ, la règle de contraction à droite est interdite mais elle est autorisée à gauche. Ainsi, si $\vdash A \vee \neg A$ n'est pas prouvable dans LJ, on peut en revanche prouver $\vdash \neg\neg(A \vee \neg A)$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\overline{A \vdash_{LK} A}\ Ini}{\vdash_{LK} A, \neg A}\ R\neg}{\vdash_{LK} A, A \vee \neg A}\ R\vee}{\vdash_{LK} A \vee \neg A, A \vee \neg A}\ R\vee}{\vdash_{LK} A \vee \neg A}\ RC
\qquad
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\overline{A \vdash_{LJ} A}\ Ini}{A \vdash_{LJ} A \vee \neg A}\ R\vee}{\neg(A \vee \neg A), A \vdash_{LJ}}\ L\neg}{\neg(A \vee \neg A) \vdash_{LJ} \neg A}\ R\neg}{\neg(A \vee \neg A) \vdash_{LJ} A \vee \neg A}\ R\vee}{\neg(A \vee \neg A), \neg(A \vee \neg A) \vdash_{LJ}}\ L\neg}{\neg(A \vee \neg A) \vdash_{LJ}}\ LC}{\vdash_{LJ} \neg\neg(A \vee \neg A)}\ R\neg$$

L'idée d'une preuve intuitionniste de $\neg\neg(A \vee \neg A)$ est de commencer par envoyer la formule à prouver dans la protion gauche du séquent grâce à la règle de la négation ce qui permet d'utiliser la contraction à gauche pour avoir plusieurs copies de la formule et de la replacer dans la portion de droite du séquent grâce au second $\neg$. De cette remarque, il n'y a qu'un pas jusqu'aux $\neg\neg$-traductions :

Définition 1.44 (*$\neg\neg$-traduction de Gödel*)

- *$A^\star = \neg\neg A$ pour A une formule atomique ;*
- *$(A \wedge B)^\star = A^\star \wedge B^\star$;*
- *$(\forall x A)^\star = \forall x A^\star$;*
- *$(\neg A)^\star = \neg A^\star$;*
- *$(A \Rightarrow B)^\star = A^\star \Rightarrow B^\star$;*
- *$(A \vee B)^\star = \neg\neg(A^\star \vee B^\star)$;*
- *$(\exists x A)^\star = \neg\neg\exists x A^\star$.*

Théorème 1.45

$\Gamma \vdash_{LK} A$ si, et seulement si, $\Gamma^\star \vdash_{LJ} A^\star$.

Lemme 1.46

$\vdash_{LK} A \Leftrightarrow A^\star$.

Définition 1.47

Une formule A est dite **stable** *quand* $\vdash_{LJ} \neg\neg A \Rightarrow A$.

Lemme 1.48

Quelle que soit la formule A, $A^\star$ est stable.

Lemme 1.49

Si $\Gamma \vdash_{LK} \Delta$ *alors* $\Gamma^\star, \neg\Delta^\star \vdash_{LJ}$.

Remarque 1.50

On peut donc finalement affirmer que ce qui distingue LJ de LK, plus que le plus grand pouvoir déductif de LK, c'est le fait que la prouvabilité en LJ est une relation plus fine qu'en LK.

Informellement, un logicien intuitionniste ne peut pas nécessairement prouver une formule A qu'un logicien classique peut prouver, mais il peut trouver une autre formule ($A^\star$) qu'il saura prouver et que le logicien classique ne sait pas distinguer de la précédente.

En particulier, en logique intuitionniste, l'utilisation du tiers exclu (ou de la contraction à droite) doit être explicitement mentionnée dans la formule par l'utilisation de la double négation (en logique linéaire, on aura le même phénomène mais avec toutes les règles structurelles à part la règle d'échange, et non pas seulement pour RC).

1.4. La déduction naturelle

La déduction naturelle a également été introduite par Gentzen [Gen69]. Les déductions formelles de la déduction naturelle sont proches du raisonnement mathématique usuel, c'est d'ailleurs ce qui a motivé l'introduction de ce système par Gentzen en 1935 qui voulait mettre au point un formalisme qui reflète de manière aussi précise que possible les raisonnements logiques effectivement à l'œuvre dans les preuves mathématiques[4]

Définition 1.51 (*Séquents de la déduction naturelle*)

Les séquents de la déduction naturelle sont donnés par un ensemble Γ, les hypothèses du séquent, et une formule A, la conclusion du séquent. On notera les séquents $\Gamma \vdash A$.

Définition 1.52 (*Déduction naturelle NJ, NK*)

- *La déduction naturelle intuitionniste, ou NJ, est définie par les règles d'inférence des figures 1.12 et 1.13 ;*
- *La déduction naturelle classique, ou NK, est le résultat de l'ajout de la règle de la figure 1.14 à NJ.*
- *Une* **déduction** *(ou preuve) du séquent $\Gamma \vdash A$ dans NJ ou NK est un arbre fini d'inférences du système considéré dont la racine est $\Gamma \vdash A$.*

[4] Gentzen [Gen69], section 2 : *«we wish to set up a formalism that reflects as accurately as possible the actual logical reasoning involved in mathematical proofs».*

$$\dfrac{}{\Gamma, A \vdash A}\ Ax \qquad \dfrac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ \wedge I \qquad \dfrac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\ \Rightarrow I \qquad \dfrac{\Gamma \vdash A}{\Gamma \vdash \forall x.A}\ \forall I \quad (\star)$$

$$\dfrac{\Gamma \vdash A_1 \wedge A_2}{\Gamma \vdash A_i}\ \wedge Ei \quad (i \in \{1, 2\}) \qquad \dfrac{\Gamma \vdash A \quad \Gamma \vdash A \Rightarrow B}{\Gamma \vdash B}\ \Rightarrow E \qquad \dfrac{\Gamma \vdash \forall x.A}{\Gamma \vdash A[t/x]}\ \forall E$$

$(\star)$ Pour cette règle, $x \notin FV(\Gamma)$.

FIG. 1.12 – Déduction naturelle NJ, fragment $\Rightarrow, \wedge, \forall$.

$$\dfrac{\Gamma \vdash A}{\Gamma \vdash A \vee B}\ \vee I1 \qquad \dfrac{\Gamma \vdash B}{\Gamma \vdash A \vee B}\ \vee I2 \qquad \dfrac{\Gamma \vdash A[t/x]}{\Gamma \vdash \exists x.A}\ \exists I \qquad \dfrac{\Gamma, A \vdash \perp}{\Gamma \vdash \neg A}\ \neg I$$

$$\dfrac{\Gamma \vdash A \vee B \quad \Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma \vdash C}\ \vee E \qquad \dfrac{\Gamma \vdash \exists x.A \quad \Gamma, A \vdash C}{\Gamma \vdash C}\ \exists E\ (\star) \qquad \dfrac{\Gamma \vdash \neg A \quad \Gamma \vdash A}{\Gamma \vdash \perp}\ \neg E$$

$(\star)$ x n'est pas libre dans Γ, C

FIG. 1.13 – Fragment $\vee, \exists, \neg$ de la déduction naturelle.

Remarque 1.53

On donne ci-dessous un exemple de preuve en déduction naturelle :

$$\begin{array}{c}
\dfrac{\dfrac{}{\Gamma \vdash A \wedge B \Rightarrow C}\ Ax \qquad \dfrac{\dfrac{}{\Gamma \vdash A}\ Ax \quad \dfrac{}{\Gamma \vdash B}\ Ax}{\Gamma \vdash A \wedge B}\ \wedge I}{A \wedge B \Rightarrow C, A, B \vdash C}\ \Rightarrow E \\
\vdots\ \Rightarrow I^3 \\
\vdash (A \wedge B \Rightarrow C) \Rightarrow (A \Rightarrow (B \Rightarrow C))
\end{array}$$

où $\Gamma = A \wedge B \Rightarrow C, A, B$.

Parmi les dérivations en déduction naturelle, on met en évidence certaines structures, certaines formes de dérivation. On va en particulier être intéressé par l'étude des relations entre diverses preuves via la notion de coupure :

Définition 1.54 (*Coupure en déduction naturelle*)

- *Une* **coupure** *en déduction naturelle est la succession immédiate de l'introduction d'un connecteur et de son élimination, comme dans l'exemple suivant :*

$$\dfrac{\dfrac{\mathcal{D}_1}{\Gamma \vdash A} \qquad \dfrac{\dfrac{\mathcal{D}_2}{\Gamma, A \vdash B}}{\Gamma \vdash A \Rightarrow B}\ \Rightarrow I}{\Gamma \vdash B}\ \Rightarrow E$$

- *Une* **preuve sans coupure** *est une preuve qui ne contient aucune coupure.*

Une preuve contenant une coupure peut être vue comme une preuve structurée avec

$$\frac{\Gamma, \neg A \vdash \bot}{\Gamma \vdash A} \perp C$$

FIG. 1.14 – Raisonnement par l'absurde.

---

un lemme : dans l'exemple ci-dessus, on démontre le lemme A dans la dérivation $\mathcal{D}_1$ d'une part et on utilise le lemme A dans la dérivation $\mathcal{D}_2$ puisqu'il fait partie de l'ensemble d'hypothèses.

Il est naturel de se demander si on peut obtenir, à partir d'une déduction contenant une coupure, une déduction sans coupure, ce qui correspondrait informellement à une démonstration directe, sans lemme. La déduction naturelle permet de définir un procédé d'élimination des coupures qui a été étudié en détail par Prawitz [Pra65] :

$$\frac{\dfrac{\mathcal{D}_1}{\Gamma \vdash A} \quad \dfrac{\dfrac{\mathcal{D}_2}{\Gamma, A \vdash B}}{\Gamma \vdash A \Rightarrow B} \Rightarrow I}{\Gamma \vdash B} \Rightarrow E \qquad \longrightarrow^{\star} \qquad \frac{\tilde{\mathcal{D}}}{\Gamma \vdash B} \qquad \text{où } \tilde{\mathcal{D}} \text{ est sans coupure.}$$

qui a de bonnes propriétés, à savoir d'être confluent (pour le fragment $\Rightarrow, \wedge, \forall$ de NJ) et de terminer.

Les déductions formelles de la déduction naturelle sont proches du raisonnement mathématique usuel, mais elles manquent de structure :

- La notion de coupure est implicite : il n'y a pas de règle explicite pour la coupure comme on en a pour les connecteurs logiques;
- ND n'est vraiment satisfaisante que pour un fragment de la logique intuitionniste : $\Rightarrow$, $\wedge$, $\forall$.

  De manière paradoxale, les connecteurs qui sont le plus intéressants du point de vue de la logique intuitionniste sont $\vee$ et $\exists$ pour lesquels la déduction naturelle ne se comporte pas bien.

1.4.1 Déduction naturelle à conclusions multiples

La non-confluence de l'élimination des coupures en logique classique est problématique, en particulier si l'on veut utiliser ces systèmes de démonstration pour représenter des programmes et l'élimination des coupures pour représenter la dynamique des calculs. Il est souvent souhaitable, sauf dans des cas bien particuliers, qu'un programme, lancé sur les mêmes arguments, donne toujours le même résultat.

Parigot a introduit une variante de la déduction naturelle NK, dont les conclusions ont plusieurs conclusions, comme dans LK mais dont l'élimination des coupures est confluente. Ce système est à l'origine du $\lambda\mu$-calcul que nous étudierons dans le chapitre suivant. Le système de déduction naturelle classique à conclusion multiple, que l'on notera CND, est un sous-produit de la déduction libre [Par91] (FD, pour Free deduction) introduite par Parigot, système qui emprunte certains aspects à la déduction naturelle et au calcul des séquents et qui peut encoder le calcul des séquents et la déduction naturelle.

Le système CND est présenté en figure 1.15.

$$\frac{}{\Gamma, A \vdash A, \Delta}\ Ax$$

$$\frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \Rightarrow B, \Delta} \Rightarrow I \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma \vdash A \Rightarrow B, \Delta}{\Gamma \vdash B, \Delta} \Rightarrow E$$

$$\frac{\Gamma, A \vdash \Delta}{\Gamma \vdash \neg A, \Delta} \neg I \qquad \frac{\Gamma \vdash \neg A, \Delta \quad \Gamma \vdash A, \Delta}{\Gamma \vdash \Delta} \neg E$$

FIG. 1.15 – Déduction naturelle Classique à plusieurs conclusions, fragment $\Rightarrow, \neg$.

1.5. La logique linéaire

Une origine sémantique. La logique linéaire [Gir87a] trouve son origine dans une simplification des domaines de Scott [Sco76] par Girard, les espaces cohérents qui utilisent la propriété de stabilité[5] de Berry [Ber78].

En particulier, les fonctions linéaires (fonctions stables qui préservent les unions quelconques : $f(a \cup b) = f(a) \cup f(b)$) jouent un rôle essentiel dans les espaces cohérents et conduisent à une décomposition de l'espace de fonctions usuel $X \Rightarrow Y$ en une partie linéaire $\multimap$ et une partie dite exponentielle $!(\_)$:

$$X \Rightarrow Y \quad = \quad !X \multimap Y$$

Si la logique linéaire trouve son origine dans les sémantiques de cohérence, sa théorie de la démonstration est très riche et c'est sur cette théorie, et en particulier son calcul des séquents, que nous allons maintenant nous pencher.

1.5.1 Calcul des séquents linéaire, LL

L'introduction de la linéarité conduit Girard [Gir87a] à remettre en cause les règles structurelles de contraction et d'affaiblissement : ces règles empêchent évidemment tout traitement linéaire, c'est-à-dire dans lequel le nombre d'occurrences d'une formule importerait. Alors que la logique intuitionniste, dans son calcul des séquent LJ, interdit la contraction à droite, la logique linéaire va interdire purement et simplement les règles structurelles.

La logique linéaire va avoir les propriétés de constructivité de LJ tout en ayant la symétrie de LK. On considérera ainsi des séquents à un seul côté comme on l'a vu pour LK : $\vdash \Gamma$ et on considérera des formules littérales, ou atomiques : $a, a^\perp, b, b^\perp, \ldots$ la négation linéaire $(\_)^\perp$ étant définie au niveau des formules atomiques par les dualités de de Morgan.

[5]On appelle ***espace cohérent* X** la donnée d'un ensemble appelé la ***trame*** de **X**, noté $|$ X $|$ et d'une relation réflexive et symétrique, la relation de cohérence, notée $x \frown_X y$. On appelle clique de X un sous-ensemble a de $|$ X $|$ dont les points sont deux à deux cohérents et on note $a \sqsubset X$ pour «a est une clique de X».
Une ***fonction stable*** entre deux espaces cohérents X et Y est une fonction qui préserve les cliques, qui est croissante et continue et qui vérifie la condition de stabilité, c'est-à-dire qui préserve les intersections compatibles (si $a \cup b \sqsubset X, f(a \cap b) = f(a) \cap f(b)$) ce qui mène à considérer l'***ordre stable*** : $f \sqsubseteq g$ si $\forall a \subseteq b \sqsubset X, f(a) = f(b) \cap g(a)$.

Deux conjonctions et deux disjonctions. En l'absence des règles structurelles, les présentations alternatives des inférences pour la conjonction et la disjonction classiques de LK ne sont plus équivalentes. Cela nous amène naturellement à considérer deux connecteurs distincts pour la conjonction et deux connecteurs distincts pour la disjonction : une paire conjonction/disjonction sera dite ***multiplicative*** tandis que l'autre sera dite ***additive*** :

Définition 1.55 (*Connecteurs multiplicatifs et additifs*)

- *Les* **connecteurs multiplicatifs** *seront notés* $\otimes$ *(le* **tenseur**, *la conjonction multiplicative),* $\parr$ *(le* **par**, *la disjonction multiplicative),* $\mathbf{1}$ *(l'unité de la conjonction) et* $\perp$ *(l'unité de la disjonction).*
- *Les* **connecteurs additifs** *seront notés* $\&$ *(le* **avec**, *la conjonction additive),* $\oplus$ *(le* **plus**, *la disjonction additive),* $\top$ *(l'unité de la conjonction) et* $\mathbf{0}$ *(l'unité de la disjonction).*

Définition 1.56 (*Inférences multiplicatives et additives*)

Les inférences associées aux connecteurs multiplicatifs et additifs sont les suivantes :

$$\frac{\vdash \Gamma}{\vdash \Gamma, \perp}\,[\perp] \quad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}\,[\parr] \qquad \frac{\vdash \Gamma, A \quad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B}\,[\otimes] \quad \frac{}{\vdash \mathbf{1}}\,[\mathbf{1}]$$

$$\frac{}{\vdash \Gamma, \top}\,[\top] \quad \frac{\vdash \Gamma, A \quad \vdash \Gamma, B}{\vdash \Gamma, A \,\&\, B}\,[\&] \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, A \oplus B}\,[\oplus_0] \quad \frac{\vdash \Gamma, B}{\vdash \Gamma, A \oplus B}\,[\oplus_1]$$

Remarque 1.57

On a les paires de connecteurs duaux pour les relations de de Morgan : $\otimes$ / $\parr$, $\oplus$ / $\&$, $\mathbf{1}/\perp$ *et* $\mathbf{0}/\top$.

L'implication linéaire est ainsi définie comme : $A \multimap B \triangleq A^\perp \parr B$.

Remarque 1.58

Les unités sont des versions d'arité nulle des connecteurs multiplicatifs et additifs :

- $[\parr]$ *a deux sous-formules,* A *et* B *;* $[\perp]$ *n'en a aucune.*
- $[\otimes]$ *a deux prémisses,* $\vdash \Gamma, A$ *et* $\vdash \Delta, B$, *et une partition du contexte en deux sous-ensembles ;* $[\mathbf{1}]$ *n'a aucune prémisse et le contexte est vide (l'ensemble vide est le seul ensemble à avoir une partition de cardinalité nulle).*
- $[\&]$ *a deux prémisse,* $\vdash \Gamma, A$ *et* $\vdash \Gamma, B$, *qui copient chacune le contexte de la formule principale ;* $[\top]$ *a zéro prémisse et le contexte est libre.*
- $\oplus$ *a deux règles d'inférence ;* $\mathbf{0}$ *n'a aucune règle d'inférence.*

Les exponentiels. L'autre ingrédient des espaces cohérents que nous avons mentionné est la construction ! qui permettait de relier l'implication linéaire et l'implication classique. On considère donc deux nouveaux connecteurs ! et ?, les ***exponentiels***, qui vont permettre de gérer la multiplicité des formules ; ce n'est que sur les formules préfixées d'un ? que l'on pourra effectuer contraction ou affaiblissement :

$$\frac{\vdash ?\,\Gamma, B}{\vdash ?\,\Gamma, !\,B}\,[!] \qquad \frac{\vdash \Gamma}{\vdash \Gamma, ?\,B}\,[?\,w] \qquad \frac{\vdash \Gamma, ?\,B, ?\,B}{\vdash \Gamma, ?\,B}\,[?\,c] \qquad \frac{\vdash \Gamma, B}{\vdash \Gamma, ?\,B}\,[?\,d]$$

Les deux exponentiels sont duaux pour les lois de De Morgan ($(?\,A)^\perp \equiv !\,A^\perp$)

$$\frac{}{\vdash A, A^\perp}\ ini \qquad \frac{\vdash \Gamma, A \quad \vdash \Delta, A^\perp}{\vdash \Gamma, \Delta}\ cut$$

$$\frac{\vdash \Gamma}{\vdash \Gamma, \perp}\ [\perp] \quad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}\ [\parr] \qquad \frac{\vdash \Gamma, A \quad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B}\ [\otimes] \quad \frac{}{\vdash \mathbf{1}}\ [\mathbf{1}]$$

$$\frac{}{\vdash \Gamma, \top}\ [\top] \quad \frac{\vdash \Gamma, A \quad \vdash \Gamma, B}{\vdash \Gamma, A \,\&\, B}\ [\&] \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, A \oplus B}\ [\oplus_0] \quad \frac{\vdash \Gamma, B}{\vdash \Gamma, A \oplus B}\ [\oplus_1]$$

$$\frac{\vdash \Gamma, A[c/x]}{\vdash \Gamma, \forall x.A}\ [\forall] \quad \text{si } c \notin FV(\Gamma, \forall x.A) \qquad \frac{\vdash \Gamma, A[t/x]}{\vdash \Gamma, \exists x.A}\ [\exists]$$

$$\frac{\vdash \Gamma, A}{\vdash \Gamma, \forall^2 X.A}\ [\forall^2] \quad \text{si } X \notin FV(\Gamma) \qquad \frac{\vdash \Gamma, A[F/X]}{\vdash \Gamma, \exists^2 X.A}\ [\exists^2]$$

$$\frac{\vdash ?\,\Gamma, B}{\vdash ?\,\Gamma, !\,B}\ [!] \quad \frac{\vdash \Gamma}{\vdash \Gamma, ?\,B}\ [?\,w] \quad \frac{\vdash \Gamma, ?\,B, ?\,B}{\vdash \Gamma, ?\,B}\ [?\,c] \quad \frac{\vdash \Gamma, B}{\vdash \Gamma, ?\,B}\ [?\,d]$$

FIG. 1.16 – Calcul des séquents pour LL

On complète le tableau en considérant les quantificateurs du premier et du second ordre.

Définition 1.59 (*Calcul des séquents de LL*)

Le calcul des séquents de LL est donné en figure 1.16.

Définition 1.60 (*Fragments particuliers*)

On considèrera les fragments de LL suivants, avec ou sans quantificateurs selon le cas :

- *MALL est le fragment de LL qui ne contient pas les exponentiels ;*
- *MLL est le fragment de LL qui ne contient que les connecteurs multiplicatifs ;*
- *MELL est le fragment de LL contenant les connecteurs multiplicatifs et exponentiels ;*
- *On désignera par MLL^- le fragment multiplicatif sans unité.*

Remarque 1.61

On peut restreindre la règle initiale à ne s'appliquer que sur des formules atomiques :

$$\frac{}{\vdash a, a^\perp}\ ini$$

Exemple 1.62

On donne en figure 1.17 un exemple de preuve en logique linéaire sur laquelle nous reviendrons au chapitre 9.

La logique linéaire satisfait évidemment le ***théorème d'élimination des coupures*** ce qui fournit une propriété de sous-formule (tant que le second ordre n'est pas concerné) et des propriétés similaires à la propriété d'existence et de disjonction de LJ :

Propriété 1.63

- *Si $\vdash_{LL} \exists x.A$ alors il existe un terme t tel que $\vdash_{LL} A[t/x]$;*
- *Si $\vdash_{LL} A \oplus B$ alors soit $\vdash_{LL} A$ soit $\vdash_{LL} B$.*

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\overline{\vdash q, q^{\perp}}\ ini \quad \overline{\vdash r, r^{\perp}}\ ini}{\vdash q \otimes r, q^{\perp}, r^{\perp}}\ \otimes}{\vdash q \otimes r, q^{\perp} \parr r^{\perp}}\ \parr \quad \overline{\vdash s, s^{\perp}}\ ini}{\vdash q \otimes r, s \otimes (q^{\perp} \parr r^{\perp}), s^{\perp}}\ \otimes}{\vdash p \oplus (q \otimes r), s \otimes (q^{\perp} \parr r^{\perp}), s^{\perp}}\ \oplus \quad \overline{\vdash \mathbf{1}}\ \mathbf{1}}{\vdash p \oplus (q \otimes r), s \otimes (q^{\perp} \parr r^{\perp}), s^{\perp} \otimes \mathbf{1}}\ \otimes
$$

FIG. 1.17 – Exemple d'une preuve dans LL.

En outre, les règles exponentielles évitent de retrouver, dans LL, les paires critiques de Lafont que nous avons vu dans LK. En effet, dans le cas d'une coupure :

$$
\dfrac{\vdash \Gamma, !\, A^{\perp} \quad \vdash \Delta, ?\, A}{\vdash \Gamma, \Delta}\ cut
$$

seule la formule $?\, A$ peut être affaiblie.

Si l'élimination des coupures de LL ne présente pas le problème de non-déterminisme révélé par la paire critique de Lafont, l'élimination des coupures de LL n'est pas pour autant confluente à cause des cas de commutation lors de l'élimination des coupures quand la formule de coupure n'est pas la formule principale des inférences de ses prémisses comme le montre l'exemple suivant :

$$
\dfrac{\dfrac{\dfrac{\mathcal{D}}{\vdash \Gamma, A, C}}{\vdash \Gamma, A \oplus A', C}\ [\oplus_0] \quad \dfrac{\dfrac{\mathcal{E}}{\vdash \Delta, B, B', C^{\perp}}}{\vdash \Delta, B \parr B', C^{\perp}}\ [\parr]}{\vdash \Gamma, A \oplus A', \Delta, B \parr B'}\ cut
$$

On peut réduire la coupure de plusieurs manières selon l'ordre dans lequel on effectue la commutation de la coupure avec les règles $[\oplus_0]$ et $[\parr]$, on peut ainsi obtenir :

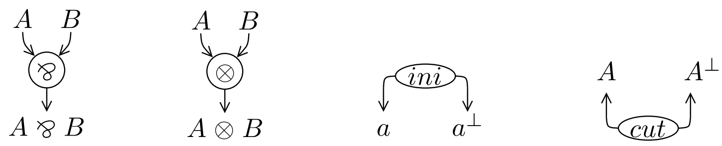
$$
\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{D}}{\vdash \Gamma, A, C} \quad \dfrac{\mathcal{E}}{\vdash \Delta, B, B', C^{\perp}}}{\vdash \Gamma, A, \Delta, B, B'}\ cut}{\vdash \Gamma, A, \Delta, B \parr B'}\ [\parr]}{\vdash \Gamma, A \oplus A', \Delta, B \parr B'}\ [\oplus_0]
\qquad \text{ou} \qquad
\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{D}}{\vdash \Gamma, A, C} \quad \dfrac{\mathcal{E}}{\vdash \Delta, B, B', C^{\perp}}}{\vdash \Gamma, A, \Delta, B, B'}\ cut}{\vdash \Gamma, A \oplus A', \Delta, B, B'}\ [\oplus_0]}{\vdash \Gamma, A \oplus A', \Delta, B \parr B'}\ [\parr]
$$

Cette non-confluence n'est pas du même ordre que celle de la paire critique de Lafont car elle est due à l'ordonnancement des règles d'inférence imposé par le calcul des séquents. Ainsi, les deux preuves obtenues sont équivalentes modulo les permutations d'inférences à la Kleene-Curry étudiées plus haut.

1.5.2 Réseaux de démonstration pour *MLL*

La manière usuelle d'aborder le problème de l'identité de preuve en logique linéaire (et de donner une représentation canonique des preuves) consiste à utiliser les réseaux de démonstration qui ont été initialement introduits par Girard [Gir87a].

Les réseaux de démonstration sont des structures qui ne conservent pas l'information d'ordonnancement des règles non nécessaire dans une preuve de séquents. Une structure de



La cellule *ini* est restreinte aux formules atomiques.

FIG. 1.18 – Cellules des réseaux de démonstration pour MLL^{-}.

démonstration MLL^{-} est ainsi un graphe qui consiste en la forêt des arbres de formules du séquent $\vdash \Gamma$ complétée d'une structure additionnelle représentant les règles initiales :

Définition 1.64 (*Structures de démontration MLL^{-}*)

Une **structure de démonstration** *(ou structure de preuve) de MLL^{-} de conclusions $\vdash \Gamma$ est un graphe constitué de cellules représentées en Figure 1.18 qui sont connectées par des fils portant pour labels des formules de MLL^{-}.*

Pour chaque formule F de $\vdash \Gamma$ il y a un fil pendant avec le label F appelé **conclusion**.

Des conditions supplémentaires sont imposées de manière à assurer qu'une ***structure de preuve*** est bien un objet logique représentant une preuve, c'est-à-dire qu'il peut être ***séquentialisé*** en une preuve de séquent.

Définition 1.65 (*Réseaux de démonstration MLL^{-}*)

Un réseau de démonstration MLL^{-} de conclusions $\vdash \Gamma$ est une structure de preuve qui provient de la déséquentialisation d'une preuve du calcul des séquents $\mathcal{D}$ de $\vdash \Gamma$ en oubliant l'ordonnancement des règles d'inférence : une règle d'inférence de MLL^{-} est transformée comme la cellule correspondante dans la Figure 1.18 et les cellules sont combinées en suivant la trace des occurrences de formules dans la preuve de séquent.

L'élimination des coupures dans les réseaux de démonstration est confluente.

Critères de correction. La définition précédente ne donne pas de critère pratique qui permette de vérifier qu'une structure de preuve donnée est bien un réseau de démonstration, c'est-à-dire qu'elle peut effectivement être séquentialisée en une preuve de LL. Il existe de nombreux critères pour caractériser les réseaux de démonstration qui ont en général la propriété d'être non pas des critères inductifs, mais géométriques. On peut citer par exemple le critère de Danos-Régnier : une structure de preuve π est un réseau de démonstration si tout sous-graphe obtenu en enlevant, pour chaque nœud $⅋$ de π, exactement l'une des deux arêtes prémisses du nœud est connexe et acyclique.

1.6. Programmer en logique

Le principe de la programmation logique est de modéliser le calcul comme une recherche de démonstration. Dans ce modèle de calcul, on identifie un programme avec un séquent et on voit le calcul comme la recherche d'une preuve de ce séquent.

Les différents cadres de programmation logique varient selon les logiques utilisées, les formes de séquents que l'on considère ainsi que les stratégies de démonstration que l'on utilise [MNPS91].

Un exemple historique de programmation logique est celui de la programmation par clauses de Horn sur laquelle est fondé le langage `Prolog`.

1.6.1 Preuves uniformes

Nous allons présenter un cadre formel développé dans un article de Dale Miller, et al. [MNPS91] qui permet de généraliser la programmation logique à des logiques et des ensembles de formules plus généraux. On se place dans le cadre intuitionniste de LJ.

Définition 1.66 (*Preuves uniformes*)

Une **preuve uniforme** *est une preuve en calcul des séquents satisfaisant la condition suivante : pour toute occurrence dans la preuve d'un séquent dont le côté droit n'est pas une formule atomique, ce séquent est conclusion d'une règle d'introduction droite.*

Définition 1.67 (*Prouvabilité uniforme*)

Étant donné un séquent $\Gamma \vdash F$, on dit que $\Gamma \vdash F$ est **prouvable uniformément** *dans une logique donnée s'il existe une preuve de $\Gamma \vdash F$ dans cette logique qui soit uniforme.*

Étant donnée une notion de prouvabilité $\vdash$, on note $\vdash^{unif}$ la prouvabilité uniforme correspondante.

Définition 1.68 (*Langage de programmation logique abstrait*)

Un **langage de programmation logique abstrait** *est la donnée d'un triplet $(\mathcal{D}, \mathcal{G}, \vdash_{\mathcal{L}})$ tel que, pour toute partie finie Γ de $\mathcal{D}$, les* **clauses de programme**, *et pour toute formule $G \in \mathcal{G}$, le* **but**, *la prouvabilité de $\Gamma \vdash G$ (ie. $\Gamma \vdash_{\mathcal{L}} G$) coïncide avec la prouvabilité uniforme de $\Gamma \vdash G$ (ie. $\Gamma \vdash_{\mathcal{L}}^{unif} G$).*

Remarque 1.69

Les propriétés de prouvabilité uniforme sont très liées aux résultats de Kleene-Curry sur la permutabilité des inférences.

1.6.2 Exemples et contre-exemples

On donne ici quelques exemples et contre-exemples de langages de programmation logique abstraits.

Logique intuitionniste. La logique intuitionniste ne fournit pas un langage de programmation logique abstrait : si l'on considère la prouvabilité intuitionniste, alors le séquent $p \vee q \vdash q \vee p$ est prouvable mais n'est pas prouvable uniformément. En effet, la preuve :

$$\dfrac{\dfrac{\dfrac{}{p \vdash p}\,Ax}{p \vdash q \vee p}\,R\vee 1 \qquad \dfrac{\dfrac{}{q \vdash q}\,Ax}{q \vdash q \vee p}\,R\vee 2}{p \vee q \vdash q \vee p}\,L\vee$$

est une preuve intuitionniste, mais il n'est pas possible de fournir une preuve uniforme de ce séquent : supposons en effet qu'on ait une preuve uniforme, on aurait comme dernière règle, soit

$$\dfrac{p \vee q \vdash p}{p \vee q \vdash q \vee p}\,R\vee 1$$

soit

$$\frac{p \vee q \vdash q}{p \vee q \vdash q \vee p}\ R\vee 2$$

mais aucun des deux séquents prémisses n'est prouvable (que ce soit uniformément ou simplement dans LJ).

Logique linéaire. De même, la logique linéaire complète, même dans son fragment intuitionniste, ne donne pas un langage de programmation logique abstrait. On peut considérer les contre-exemples suivants :

1. $a \otimes b \vdash b \otimes a$ (pour la même raison que plus haut...)
2. $!a \vdash\ !a \otimes\ !a$ (pour pouvoir prouver ce séquent, il faut d'abord contracter le $!a$ à gauche puis décomposer le $\otimes$)
3. $!(a \,\&\, b) \vdash\ !a$
4. $b \otimes (b \multimap\ !a) \vdash\ !a$

Les clauses de Horn. Si l'on restreint les ensembles $\mathcal{D}$ et $\mathcal{G}$ aux clauses de Horn et si l'on considère la prouvabilité intuitionniste, on obtient un langage de programmation logique abstrait. Cela revient à avoir les ensembles $\mathcal{D}$ et $\mathcal{G}$ définis par les grammaires respectives suivantes :

$$\boxed{\begin{array}{lcl} G & ::= & \top \mid A \mid G_1 \wedge G_2 \mid G_1 \vee G_2 \mid \exists x.G \\ D & ::= & A \mid G \Rightarrow A \mid D_1 \wedge D_2 \mid \forall x.D \end{array}}$$

où A désigne une formule atomique du 1^er^ ordre.

Formules héréditaires de Harrop. On peut étendre la syntaxe des buts et conserver un langage de programmation logique abstrait, en utilisant les formules héréditaires de Harrop :

$$\boxed{\begin{array}{lcl} G & ::= & \top \mid A \mid G_1 \wedge G_2 \mid G_1 \vee G_2 \mid \exists x.G \mid \forall x.G \mid D \Rightarrow G \\ D & ::= & A \mid G \Rightarrow A \mid D_1 \wedge D_2 \mid \forall x.D \end{array}}$$

1.6.3 Sémantique opérationnelle de la recherche de preuve

Instructions de recherche.

SUCCESS $\mathcal{P} \vdash_{\mathcal{O}} \top$;

AND $\mathcal{P} \vdash_{\mathcal{O}} G_1 \wedge G_2$ si, et seulement si, $\mathcal{P} \vdash_{\mathcal{O}} G_1$ et $\mathcal{P} \vdash_{\mathcal{O}} G_2$;

OR $\mathcal{P} \vdash_{\mathcal{O}} G_1 \vee G_2$ si, et seulement si, $\mathcal{P} \vdash_{\mathcal{O}} G_1$ ou $\mathcal{P} \vdash_{\mathcal{O}} G_2$;

INSTANCE OF $\mathcal{P} \vdash_{\mathcal{O}} \exists_\alpha G$ si, et seulement si, il y a un terme t de type α tel que $\mathcal{P} \vdash_{\mathcal{O}} G[t/x]$;

AUGMENT $\mathcal{P} \vdash_{\mathcal{O}} D \Rightarrow G$ si, et seulement si, $\mathcal{P} \cup \{D\} \vdash_{\mathcal{O}} G$;

GENERIC $\mathcal{P} \vdash_{\mathcal{O}} \forall_\alpha G$ si, et seulement si, $\mathcal{P} \vdash_{\mathcal{O}} G[c/x]$ où c est un paramètre de type α qui n'est libre ni dans $\mathcal{P}$ ni dans G.

Chaînage arrière.

Définition 1.70 (*Backchaining*)

On considère une formule B construite sur les connecteurs $\top$, $\wedge$, $\Rightarrow$ et $\forall$.

On définit $|B|$ comme le plus petit ensemble de paires tel que :

1. $(\emptyset, B) \in |B|$;
2. *si* $(\Delta, B_1 \wedge B_2) \in |B|$, *alors* $(\Delta, B_1) \in |B|$ *et* $(\Delta, B_2) \in |B|$;
3. *si* $(\Delta, \forall x.B') \in |B|$, *alors* $(\Delta, B'[t/x]) \in |B|$ *pour tout terme clos* t ;
4. *si* $(\Delta, G \Rightarrow B') \in |B|$, *alors* $(\Delta \cup \{G\}, B') \in |B|$.

En termes informels, si $(\Delta, A) \in |B|$, alors on peut utiliser B pour établir A si toutes les formules de Δ peuvent être prouvées.

On peut restreindre le backchaining au cas où A est une formule atomique :

$$\frac{\Gamma \longrightarrow G_1 \quad \ldots \quad \Gamma \longrightarrow G_n}{\Gamma \longrightarrow A}\ BC$$

si A est atomique, $B \in \Gamma$ et $(\{G_1, \ldots G_n\}, A) \in |B|$.

Remarque 1.71

Avec la logique linéaire, le backchaining devient : $||B||$ est le plus petit ensemble de paires tel que :

1. $(\emptyset, \emptyset, B) \in ||B||$;
2. *si* $(\Gamma, \Delta, B_1 \,\&\, B_2) \in ||B||$, *alors* $(\Gamma, \Delta, B_1) \in ||B||$ *et* $(\Gamma, \Delta, B_2) \in ||B||$;
3. *si* $(\Gamma, \Delta, \forall x.B') \in ||B||$, *alors* $(\Gamma, \Delta, B'[t/x]$ *pour tout terme clos* t ;
4. *si* $(\Gamma, \Delta, B_1 \Rightarrow B_2) \in ||B||$, *alors* $(\Gamma \cup \{B_1\}, \Delta, B_2) \in ||B||$;
5. *si* $(\Gamma, \Delta, B_1 \multimap B_2) \in ||B||$, *alors* $(\Gamma, \Delta \cup \{B_1\}, B_2) \in ||B||$.

et on a la règle de chaînage arrière :

$$\frac{\Gamma; \emptyset \longrightarrow B_1 \quad \ldots \quad \Gamma; \emptyset \longrightarrow B_n \quad \Gamma; \Delta_1 \longrightarrow C_1 \quad \ldots \quad \Gamma; \Delta_m \longrightarrow C_m}{\Gamma; \Delta_1, \ldots, \Delta_m, B \longrightarrow A}\ BCLL$$

avec $m, n \geq 0$, *A atomique et* $(\{B_1 \ldots B_n\}, \{C_1 \ldots C_m\}, A) \in ||B||$

Chapitre 2

Notions de λ-calcul et de $\lambda\mu$-calcul

Résumé:

Nous rappelons dans ce chapitre les principales définitions concernant le λ-calcul [Chu41] et introduisons les propriétés notables de ce calcul. Nous énonçons et discutons plus particulièrement le théorème de Böhm [Bö68], ou théorème de séparation, qui motivera l'étude du $\lambda\mu$-calcul que nous mènerons dans la suite de cette thèse. Nous introduisons ensuite le $\lambda\mu$-calcul de Michel Parigot [Par92] qui est une extension simple du λ-calcul permettant d'étendre la correspondance de Curry Howard de la logique intuitionniste à la logique classique ainsi que de coder directement en $\lambda\mu$-calcul des opérateurs de contrôle usuels des langages fonctionnels. Nous rappelons également les propriétés principales du $\lambda\mu$-calcul.

Sommaire

Le λ-calcul a été inventé et développé par Alonzo Church [Chu40, Chu41] dès les années 30. Si le λ-calcul avait à l'origine pour objectif de servir de fondement aux mathématiques à partir des notions de fonction et d'application[1] – projet qui s'est révélé être un échec – le λ-calcul a en revanche eu un développement formidable dans les recherches logiques et informatiques puisqu'il s'est révélé pertinent tant pour étudier les notions de calculabilité que celles de déduction. L'isomorphisme de Curry-Howard [How80], établissant une correspondance entre λ-termes et preuves en déduction naturelle ainsi qu'entre calculs des λ-termes et réduction des coupures, a en particulier permis de construire un pont entre théorie du calcul et théorie de la démonstration.

Réferences. Ce chapitre vise à mettre en place les principales définitions utiles pour la suite de cette thèse et ne vise aucunement à l'exhaustivité, ce qui serait vain si l'on considère les quelques références auxquelles le lecteur désireux d'en lire plus – ou autrement – pourra se référer [Bar84, Kri90, GLT89, AC98].

2.1. Le λ-calcul de Church

2.1.1 Définition du λ-calcul pur

Définition 2.1 (*Syntaxe du λ-calcul*)

On se donne un ensemble infini dénombrable $\mathcal{V}$ de variables (notées $x, y, z \ldots$). On désignera par Σ_λ l'ensemble des termes du λ-calcul, ou λ-**termes**, *qui sont définis par la syntaxe suivante :*

$$t ::= \; x \mid \lambda x.t \mid (t)u$$

Remarque 2.2

λ est un lieur de variable ; on ne rappelle pas les définitions des variables libres et liées d'un terme, ni de l'α-équivalence, non plus que de la substitution (sans capture) d'une variable par un terme que l'on notera ici $t\{u/x\}$.

Définition 2.3 (*Contexte du λ-calcul*)

Les **contextes** *(à un trou) du λ-calcul sont définis par la syntaxe suivante :*

$$\mathcal{C} ::= [] \mid \lambda x.\mathcal{C} \mid (\mathcal{C})t \mid (t)\mathcal{C}$$

Si t est un terme et $\mathcal{C}$ un contexte, $\mathcal{C}[t]$ sera le résultat du remplacement de $[]$ par t dans $\mathcal{C}$.

Définition 2.4 (*Règles de réduction du λ-calcul*)

Les règles de réduction du λ-calcul sont au nombre de deux :

$$\begin{array}{llll} \text{règle } \beta : & (\lambda x.t)u & \longrightarrow_\beta & t\{u/x\} \\ \text{règle } \eta : & \lambda x.(t)x & \longrightarrow_\eta & t \qquad \text{si } x \notin FV(t) \end{array}$$

[1] À la manière dont la théorie de Zermelo-Fraenkel a permis de fonder les mathématiques sur la notion d'ensemble et d'appartenance.

Remarque 2.5

Ces règles de réduction passent au contexte, c'est-à-dire qu'on peut les appliquer partout dans le terme où apparaît un sous-terme de l'une des deux formes indiquées. Plus précisément, si $\mathcal{C}$ est un contexte, on a :

$$\begin{array}{lcll} \mathcal{C}[(\lambda x.t)u] & \longrightarrow_{\beta} & \mathcal{C}[t\,\{u/x\}] & \\ \mathcal{C}[\lambda x.(t)x] & \longrightarrow_{\eta} & \mathcal{C}[t] & \text{si } x \notin FV(t) \end{array}$$

On considèrera toujours la β-réduction dans notre calcul, et on n'inclura l'η-réduction que lorsqu'on le précisera. Un terme de la forme $(\lambda x.t)u$ (resp. $\lambda x.(t)x$ avec $x \notin FV(t)$) est appelé un **β-redex** *(resp.* **η-redex***) ; le résultat d'une application d'une réduction est le* **réduit**.

Définition 2.6 (*Formes normales*)

On appelle **forme β-normale** *du λ-calcul (ou, plus simplement, forme normale) tout terme qui ne contient pas de β-redex. On parlera également de forme $\beta\eta$-normale.*

Une définition alternative de la syntaxe du λ-calcul est la suivante :

Définition 2.7

Σ_λ est également défini par la syntaxe suivante :

$$\begin{array}{rll} t ::= & \lambda x_1 \ldots x_n.(x)t_1 \ldots t_k & (n,k \geq 0) \\ \mid & \lambda x_1 \ldots x_n.(\lambda x.u)vt_1 \ldots t_k & (n,k \geq 0) \end{array}$$

Dans le premier cas, la variable x est appelée **variable de tête** *tandis que dans le second cas, le redex $(\lambda x.u)v$ est appelé* **redex de tête**.

Proposition 2.8 (*Caractérisation des formes normales*)

Les formes β-normales du λ-calcul sont les éléments du sous-ensemble des λ-termes définis par la syntaxe suivante :

$$n ::= \lambda x_1 . \ldots \lambda x_k.(x)n_1 \ldots n_k \quad \text{avec } k,l \geq 0$$

La β-réduction est a priori une réduction non-déterministe puisqu'il peut exister un grand nombre de β-redex dans un λ-terme. On a pourtant une propriété de confluence :

Théorème 2.9 (*Confluence du λ-calcul*)

$$\begin{tikzcd} t \arrow[r, "\star", "\beta"'] \arrow[d, "\beta"', "\star"] & t' \arrow[d, dotted, "\beta"', "\star"] \\ t'' \arrow[r, dotted, "\star", "\beta"'] & u \end{tikzcd}$$

Le diagramme ci-dessus doit se lire : si $t, t', t'' \in \Sigma_\lambda$ sont tels que $t \longrightarrow^{\star}_{\beta} t', t''$ alors il existe $u \in \Sigma_\lambda$ tel que $t', t'' \longrightarrow^{\star}_{\beta} u$. Le résultat est également vrai de la réduction $\beta\eta$. Cette propriété entraîne que si t et u sont $\beta\eta$-équivalents, ils ont un réduit commun et que si t possède une forme normale, elle est unique.

2.1.2 Stratégies de réduction

Malgré la confluence, on peut être intéressé par considérer des ***stratégies de réduction*** particulières, c'est-à-dire des disciplines selon lesquelles on applique les règles de réduction.

Définition 2.10 (*Réduction de tête*)

La **réduction de tête** *consiste à réduire le redex de tête, s'il existe.*

La réduction de tête conduit à définir une notion de formes normales restreintes, les formes normales de tête :

Définition 2.11 (*Formes normales de tête*)

Les **formes normales de tête** *sont les termes de la forme* $t = \lambda x_1 . \ldots \lambda x_m . (x) t_1 \ldots t_n$ *pour* $m, n \in \mathbb{N}$*, avec les* t_i *des* λ*-termes quelconques.*

Si l'on considère la β-réduction, on constate qu'elle n'agit qu'à l'intérieur du redex qu'elle est en train de réduire. En particulier, si un terme t est une abstraction, de la forme $\lambda x.u$, cette abstraction ne sera jamais modifiée, quelles que soient les réductions que l'on fait. De la même manière, si t est de la forme $(x)u$, les réductions potentielles se feront toutes à l'intérieur de u et un réduit de t aura nécessairement la forme : $t' = (x)u'$, avec u' un réduit de u. Cela nous amène à la propriété suivante des formes normales de tête :

Proposition 2.12 (*Réduits d'une forme normale de tête*)

Soit $t = \lambda x_1 . \ldots \lambda x_m . (x) t_1 \ldots t_n$ *un terme en forme normale de tête, et soit* u *tel que* $t \longrightarrow_\beta^\star u$*. Alors il existe* $u_1, \ldots, u_n$ *tels que* $u = \lambda x_1 . \ldots \lambda x_m . (x) u_1 \ldots u_n$ *et* $t_i \longrightarrow_\beta^\star u_i$ *pour tout* $1 \leq i \leq k$*.*

Ainsi, ni le nombre de λ préfixant le terme, ni la variable de tête, ni le nombre d'arguments de la variable de tête ne varient au cours du calcul à partir du moment où l'on a atteint une forme normale de tête. Il s'agit ainsi d'une portion de terme définitivement calculée. Si on considère en plus la réduction η, cette propriété n'est évidemment plus valide.

Il n'est pas vrai que tout λ-terme ait une forme normale, pas plus qu'il n'est vrai qu'un λ-terme a nécessairement une forme normale de tête, comme le montre l'exemple suivant :

Définition 2.13 ($(\boldsymbol{\Delta})\boldsymbol{\Delta}$)

On considère le terme $\Delta = \lambda x.(x)x$*. On a* $(\Delta)\Delta \longrightarrow_\beta (\Delta)\Delta$ *et ce terme contient un seul* β*-redex. Il ne peut donc se réduire sur une forme normale :*

$$(\Delta)\Delta \longrightarrow_\beta (\Delta)\Delta \longrightarrow_\beta \ldots$$

La notion de formes normales de tête est cruciale pour définir les ***termes résolubles***, ou ***solvables*** :

Définition 2.14 (*Résolubilité*)

Un λ*-terme* t *est dit* **résoluble** *s'il se réduit sur une forme normale de tête.*

La terminologie résoluble vient de la question suivante : pour quels λ-termes clos t, peut-on trouver une suite de termes $u_1 \ldots u_n$ telle que $(t) u_1 \ldots u_n =_\beta \lambda x.x$, c'est-à-dire pour quels termes t, l'équation $\mathcal{C}[t] = u$ a-t-elle une solution quel que soit u ?

On définit également la stratégie de réduction gauche :

Définition 2.15

La stratégie de **réduction gauche** *est définie comme suit :*
- *si t n'est pas en forme normale de tête, $t \longrightarrow_g u$ si t se réduit sur u par une réduction de tête ;*
- *si $t = \lambda x_1 . \dots \lambda x_m .(x)t_1 \dots t_n$ est en forme normale de tête, soit i minimal tel que t_i ne soit pas en forme β-normale, si $t_i \longrightarrow_g u_i$ alors*

$$t \longrightarrow_g \lambda x_1 . \dots \lambda x_m .(x)t_1 \dots t_{i-1} u_i t_{i+1} \dots t_n .$$

La réduction gauche consiste à réduire toujours le β-rédex le plus à gauche dans le terme t (celui dont la «(» est la plus à gauche). Si t possède une forme normale, la réduction gauche atteindra cette forme normale.

On parlera par ailleurs de ***réduction standard*** $t_1 \longrightarrow_\beta t_2 \longrightarrow_\beta \dots \longrightarrow_\beta t_n$ s'il n'arrive jamais qu'on réduise dans t_i un redex qui soit à gauche du redex réduit[2] dans t_j pour $j < i$. Le ***théorème de standardisation*** du λ-calcul assure qu'il existe toujours une réduction standard d'un terme t à l'un quelconque de ses β-réduits u.

Définition 2.16 (0, 1)

On considère deux termes clos particuliers du λ-calcul qui serviront pour la séparation :
- $0 = \lambda x.\lambda y.y$;
- $1 = \lambda x.\lambda y.x$.

On code le couple par $(t, u) = \lambda z.(z)tu$ (on a alors $(t, u)0 \longrightarrow^\star u$ et $(t, u)1 \longrightarrow^\star t$) et on définit aussi des couples paramétrés :

$$\langle t, u \rangle_k = \lambda x_1 \dots x_k . \lambda z.((z)(t)x_1 \dots x_k)(u)x_1 \dots x_k$$

qui permettent de stocker un certain nombre d'arguments avant de déterminer s'ils retournent la composante de gauche ou bien celle de droite et on a bien sûr :
- $\langle t, u \rangle_0 = (t, u)$.
- $(\langle t, u \rangle_k)v_1 \dots v_k 1 \longrightarrow^\star (t)v_1 \dots v_k$ et
- $(\langle t, u \rangle_k)v_1 \dots v_k 0 \longrightarrow^\star (u)v_1 \dots v_k$.

2.1.3 Machine abstraite de Krivine

On peut présenter la réduction du λ-calcul via une machine abstraite, la machine abstraite de Krivine [Kri92]. Il s'agit en fait d'une variante de la réduction de tête, la réduction de tête faible, qui s'arrête dès qu'on rencontre le premier λ de la forme normale de tête. On peut construire une variante de la machine qui calcule effectivement la réduction de tête en allant regarder sous les λ mais qui est plus complexe à présenter.

[2] Il faudrait introduire la notion de résidu pour avoir une définition précise de ceci.

Définition 2.17 (*Machine Abstraite de Krivine*)

> *Les* **états** *de la* **machine abstraite de Krivine** *sont constitués de triplets* $\langle t, [e], K\rangle$ *où* t *est un* λ*-terme,* e *est un environnement qui associe des clôtures (paire d'un terme et d'un environnement, de la forme* $t[e]$*) à des* λ*-variables et où* K *est une pile de clôtures. Un* **état initial** *de la machine est de la forme* $\langle t, [\emptyset], \bot\rangle$ *où* t *est un* λ*-terme (clos), l'environnement n'est défini nulle part et où la pile est vide (représentée par* $\bot$*).*
>
> *La machine de Krivine est composée de trois transitions, dont l'applicabilité dépend de la structure du terme exécuté :*
>
> $$\begin{array}{llrcllrl} \langle x & [e] & K\rangle & \longrightarrow & \langle t & [e'] & K\rangle & \text{si } e(x) = t[e'] \\ \langle \lambda x.t & [e] & u[e']\cdot K\rangle & \longrightarrow & \langle t & [x = u[e']; e] & K\rangle & \\ \langle (t)u & [e] & K\rangle & \longrightarrow & \langle t & [e] & u[e]\cdot K\rangle & \end{array}$$

On peut interpréter les transitions de la KAM comme les «trois instructions» du λ-calcul : déréférencer une variable, empiler et dépiler.

La machine s'arrête lorsqu'elle atteint une forme de tête faible : soit un λ, soit une variable (si on admet des termes ouverts dans les états initiaux, ce qui est nécessaire pour étendre la machine au-delà de la réduction de tête faible) :
- On ne peut jamais être bloqué lorsque l'on est dans la troisième configuration ;
- On peut être bloqué lorsque l'on est dans le seconde configuration et que $K = \bot$, c'est le cas où l'on a trouvé le λ de tête ;
- On peut être bloqué dans la première configuration si l'environnement e ne lie pas la variable x à une clôture, mais cela suppose que la variable x était libre dans le terme de départ.

2.1.4 Types simples

Définition 2.18 (***λ-calcul simplement typé***)

> *On définit un système de types simples pour le* λ*-calcul. Les types se conforment à la syntaxe suivante :*
>
> $$A, B, ... ::= \ o \mid A \rightarrow B$$
>
> *On définit comme suit un système de typage pour le calcul :*
>
> $$\frac{}{\Gamma, x : A \vdash x : A}\ Var \qquad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \rightarrow B}\ Abs \qquad \frac{\Gamma \vdash t : A \rightarrow B \quad \Gamma \vdash u : A}{\Gamma \vdash (t)u : B}\ App$$

La présentation qu'on vient de faire des types simples est appelé λ-calcul simplement typé ***à la Curry***. On considérera aussi dans la suite λ-calculs typés ***à la Church***, c'est-à-dire avec annotation de type explicite sur les variables de type, les termes ayant alors la forme $\lambda x^A.t$.

Correspondance de Curry-Howard. Le λ-calcul simplement typé peut être mis en correspondance [How80] avec le fragment implicatif de la déduction naturelle intuitionniste

NJ, de la manière suivante :

Types	$\longleftrightarrow$	Formules
Jugements de typage	$\longleftrightarrow$	Séquents de NJ
λ-termes typés	$\longleftrightarrow$	Déductions de NJ
β-réduction	$\longleftrightarrow$	Réduction des coupures

Propriétés des types simples. Le λ-calcul simplement typé a de bonnes propriétés vis-à-vis de la réduction :

Théorème 2.19 (*Préservation du type*)

Si t est typable de type A et si $t \longrightarrow^\star_\beta u$, alors u est typable de type A.

Théorème 2.20 (*Normalisation forte*)

Les λ-termes typés sont fortement normalisants ; il n'existe aucune suite de réductions infinie à partir d'un λ-terme typé t.

On notera que ces propriétés restent vraies en ajoutant la η-réduction.

Formes β-normales η-longues. Un λ-terme simplement typé t de type $A_1 \to \cdots \to A_m \to A$ où A est atomique (ne contient pas de $\to$) peut être préfixé de n λ-abstractions au maximum, respectivement de type $A_1, \ldots, A_n$: $\lambda x_1 . \ldots \lambda x_n . u$ où u a le type A. D'un autre côté, rien n'oblige à ce que t soit effectivement préfixé par ces n abstractions et non pas seulement par une fraction d'entre elles. De même, si u est un sous-terme de type $B_1 \to \cdots \to B_n \to B$ (avec B atomique) de t, alors il peut lui être appliqué au maximum n termes en arguments : $(u)v_1 \ldots v_n$ où chaque v_i est de type B_i, mais rien n'oblige à ce qu'il soit effectivement appliqué à n arguments et non pas simplement aux k premiers.

On peut donc, en regardant le type d'un terme, connaître une information sur sa structure syntaxique ou sur la forme du contexte dans lequel il est placé.

Dans le cas des formes normales, la situation est encore plus intéressante, puisqu'on sait qu'un terme en forme normale est de la forme $t = \lambda x_1 . \ldots \lambda x_k .(y)u_1 \ldots u_l$ et que les considérations précédentes nous assurent que si t est de type $A_1 \to \cdots \to A_m \to A$ et y de type $B_1 \to \cdots \to B_n \to B$, avec A et B atomiques, alors on a $k \leq m$ et $l \leq n$.

La notion de ***forme β-normale η-longue*** permet de considérer des termes qui réalisent les égalités, c'est-à-dire des termes qui ont systématiquement *«le bon nombre de λ»* et *«le bon nombre d'arguments»* :

Définition 2.21 (*Forme η-longue d'un λ-terme typé*)

On définit la traduction $[\_]^{\eta l}$ d'un λ-terme simplement typé en forme normale $t : A$ en un λ-terme simplement typé $[t]^{\eta l} : A$, la **forme η-longue** *de t, comme suit :*

Soit t une forme normale de type $A_1 \to \cdots \to A_n \to A'$, où A' est atomique. t est de la forme $\lambda x_1 \ldots \lambda x_p .((y)u_1 \ldots u_k)$ pour $0 \leq p \leq n$ et $k \geq 0$; on pose alors

$$[t]^{\eta l} = \lambda x_1 \ldots \lambda x_n .(y)[u_1]^{\eta l} \ldots [u_k]^{\eta l}[x_{p+1}]^{\eta l} \ldots [x_n]^{\eta l}$$

où les variables $(x_i)_{p<i\leq n}$ sont choisies de manière à ce qu'il n'y ait pas de conflit de variable et sont de type $(A_i)_{p<i\leq n}$.

Remarque 2.22

La traduction $[\_]^{\eta l}$ est bien définie par induction sur le couple (taille du terme, taille du type) muni de l'ordre lexicographique car

- *la taille des termes considérés diminue strictement pour les u_i (mais le type peut devenir plus compliqué, d'où l'utilisation de l'ordre lexicographique) et*
- *pour les variables x_i, soit la taille diminue si le terme t n'était pas une variable, soit, au cas où t était une variable, la taille du type diminue puisque le type de x_i est l'un des A_i qui apparaît dans le type de t.*

Le fait que la forme η-longue possède «le bon nombre de λ» se comprend aisément d'après la définition de la traduction. Le fait que la variable de tête y soit, dans le terme $[t]^{\eta l}$, «appliquée au bon nombre d'arguments» se voit comme suit : dans t, $(y)u_1 \dots u_k$ était de type $A_{p+1} \to \cdots \to A_n \to A'$, c'est-à-dire que y est de type $B_1 \to \cdots \to B_k \to A_{p+1} \to \cdots \to A_n \to A'$, «attendant» $k + n - p$ arguments, précisément le nombre d'arguments auxquels y est appliquée dans le terme $[t]^{\eta l}$.

Dans la section suivante, on présente le théorème de séparation du λ-calcul non-typé.

2.2. Séparation

Cette section est consacrée à l'étude de la propriété de séparation dans le λ-calcul, également appelée théorème de Böhm du nom de Corrado Böhm qui la démontra en 1968 [Bö68].

On commence par énoncer le théorème, on étudiera ensuite quelques unes des conséquences de ce théorème et on terminera en donnant une autre caractérisation de la séparation. La section 3.3 du chapitre 3, page 73 nous donnera l'occasion de discuter plus en détail le cas de non-séparation et complètera la présente section.

Théorème 2.23 (*de séparation de Böhm [Bö68]*)

Soient t et t' deux formes normales distinctes du λ-calcul, non η-équivalentes, et ayant leurs variables libres parmi $x_1, \dots, x_n$.

Alors il existe des termes clos $u_1, \dots, u_n$ et $\vec{v} = v_1, \dots, v_k$ tels que

- *$t[\vec{u}/\vec{x}]\vec{v} \longrightarrow^\star 1$ et*
- *$t'[\vec{u}/\vec{x}]\vec{v} \longrightarrow^\star 0$.*

Cette propriété d'un calcul est très générale, on peut d'ailleurs en donner un énoncé qui s'affranchit un peu plus de la syntaxe du λ-calcul en utilisant des contextes :

Théorème 2.24

Soient t et t' deux formes normales distinctes du λ-calcul non η-équivalentes. Pour tous termes u, v, il existe un contexte $\mathcal{C}_{u,v}$ tel que $\mathcal{C}_{u,v}(t) \longrightarrow^\star u$ et $\mathcal{C}_{u,v}(t') \longrightarrow^\star v$.

2.2.1 Conséquences et portée du théorème

La séparation est une propriété importante d'un formalisme comme le λ-calcul, et ce pour plusieurs raisons dont on va maintenant donner un bref aperçu.

Conséquences sémantiques. Une première conséquence du théorème de séparation est qu'il donne des conditions fortes pour avoir un modèle du λ-calcul. En effet, deux termes normalisables ne peuvent être identifiés que s'ils sont $\beta\eta$-équivalents.

Les libertés laissées par le théorème concernent donc essentiellement le fait d'être ***extensionnel*** (deux termes η-équivalents sont, ou non, identifiés dans la sémantique) et dans l'interprétation des ***termes qui ne sont pas normalisables*** (on identifie plus ou moins de termes qui n'ont pas de forme normale – de tête). Précisons un peu plus ces notions :

Définition 2.25 (*Sémantique du λ-calcul*)

Une **sémantique** *du λ-calcul est une relation d'équivalence $\equiv$ sur les λ-termes vérifiant :*
- *Si $t \longrightarrow^\star u$, alors $t \equiv u$*
- *Si $t \equiv u$, alors pour tout contexte $\mathcal{C}$, $\mathcal{C}[t] \equiv \mathcal{C}[u]$*

Autrement dit, il s'agit d'une relation d'équivalence compatible avec la β-réduction et qui passe aux contextes.

Définition 2.26 (*Sémantique incohérente*)

Une sémantique **incohérente** *est une sémantique qui identifie tous les λ-termes, il s'agit d'une sémantique triviale.*

On a alors comme corollaire du théorème de séparation :

Corollaire 2.27

Tout sémantique $\equiv$ qui identifie deux termes normaux non η-équivalents est incohérente.

Démonstration : Soient t et u deux termes normaux non η-équivalents. Il s'agit simplement de trouver un contexte séparant grâce au théorème de Böhm. On a ensuite $\mathcal{C}[t] \longrightarrow^\star 1$ et $\mathcal{C}[u] \longrightarrow^\star 0$.

Soient alors a et b deux termes quelconques, avec $\mathcal{D} = (\mathcal{C})ab$, on a $\mathcal{D}[t] \longrightarrow^\star a$ et $\mathcal{D}[u] \longrightarrow^\star b$ et donc $a \equiv b$.

□

Comme nous l'avons noté, la liberté reste du côté de l'identification de termes sans forme normale, mais là encore, il faut être prudent : on obtient en effet une sémantique incohérente si on identifie tous les termes non normalisables. En effet, $\lambda x.\lambda y.(x)(\Delta)\Delta$ et $\lambda x.\lambda y.(y)(\Delta)\Delta$ n'ont pas de forme normale mais ne peuvent être identifiés sans obtenir une sémantique incohérente puisque dans le contexte $(([])\lambda x.u)\lambda x.v$ ils se réduisent sur u et v qui peuvent être deux termes quelconques.

Point de vue purement syntaxique. On peut également considérer le résultat comme un critère purement syntaxique que l'on veut voir rempli par un langage : le théorème (ou plus exactement la preuve du théorème) dit en substance que l'on peut totalement explorer un terme en forme normale, que l'on peut en faire remonter n'importe quelle partie en position de tête. La démonstration du théorème, par la méthode appelée de *Böhm out*, donne en fait une manière de construire une exploration des arbres des deux termes qui atteint une différence.

C'est donc un résultat qui parle autant de la syntaxe des termes que des règles du calcul, de sa sémantique. Il s'agit ainsi d'une propriété d'adéquation de notre syntaxe au calcul, et réciproquement. C'est un critère pour savoir que la syntaxe de nos termes est bonne vis-à-vis de nos règles de calcul.

2.2.2 Une autre présentation de la séparation

On peut également voir le problème de la séparation sous un autre angle, par la définition d'une relation d'orthogonalité. Cette présentation permet de rapprocher le résultat de séparation de Böhm pour le λ-calcul du théorème de Séparation de la Ludique de Jean-Yves Girard [Gir01].

Définition 2.28 (*Orthogonalité $\boldsymbol{\lambda}$-termes/contextes*)

On considère $\mathbb{N}$ l'ensemble des λ-termes en forme normale et $\mathbb{C}$ l'ensemble des contextes, c'est-à-dire des termes avec un trou.

La relation binaire $\perp \subset \mathbb{N} \times \mathbb{C}$ est définie par $(t \perp \mathcal{C}) \iff \mathcal{C}(t)$ est solvable.

On peut alors définir l'orthogonal d'un ensemble de termes en forme normale et d'un ensemble de contextes :

Définition 2.29 (*Othogonal d'un ensemble de termes*)

Soit $T \subset \mathbb{N}$ un ensemble de formes normales de Σ_λ. On définit $T^\perp$ par :

$$T^\perp = \{\mathcal{C} \in \mathbb{C} / \forall t \in T, t \perp \mathcal{C}\}$$

Définition 2.30 (*Orthogonal d'un ensemble de contextes*)

Soit $D \subset \mathbb{C}$ un ensemble de contextes. On définit $D^\perp$ par :

$$D^\perp = \{t \in \mathbb{N} / \forall \mathcal{C} \in D, t \perp \mathcal{C}\}$$

La relation précédente vérifie les propriétés classiques d'une relation d'orthogonalité :

Proposition 2.31

Soient X et Y des sous-ensembles de $\mathbb{N}$ (resp. de $\mathbb{C}$), on a :

1. $X \subset Y \Rightarrow Y^\perp \subset X^\perp$
2. $X \subset X^{\perp\perp}$
3. $X^\perp = X^{\perp\perp\perp}$

Démonstration : Ces propriétés sont presque immédiates mais nous en donnons tout de même rapidement les preuves (on fait les preuves pour les ensembles de termes, mais en fait on n'utilise pas cette hypothèse) :

1. Soient X et Y tels que $X \subset Y$. Soit alors $\mathcal{C}$ un élément de $Y^\perp$, on a par définition : $\forall t \in Y, t \perp \mathcal{C}$ et donc par inclusion de X dans Y, $\forall t \in X, t \perp \mathcal{C}$, c'est-à-dire que $\mathcal{C} \in X^\perp$.
2. Soit X un ensemble de termes et soit $t \in X$. On a par définition de $X^\perp$, $\forall \mathcal{C} \in X^\perp, t \perp \mathcal{C}$, donc $t \in (X^\perp)^\perp = X^{\perp\perp}$. D'où $X \subset X^{\perp\perp}$.
3. On a par la propriété précédente : $X \subset X^{\perp\perp}$ donc en appliquant la propriété 1 à cette inclusion, on obtient $X^\perp \supset X^{\perp\perp\perp}$. Par ailleurs, en appliquant à nouveau 2, on a $X^\perp \subset (X^\perp)^{\perp\perp} = X^{\perp\perp\perp}$, d'où l'égalité recherchée.

□

On peut alors énoncer la séparation comme suit :

Théorème 2.32 (*Séparation*)

Le double orthogonal d'un terme en forme normale est sa classe d'η-équivalence.

On peut également reformuler les choses comme suit :

Définition 2.33 (*préordre d'orthogonalité*)

On définit le préordre $\leq_\perp$ sur les formes normales de la manière suivante :

$$x \leq_\perp y \iff \{x\}^{\perp\perp} \subset \{y\}^{\perp\perp}$$

On peut alors exprimer la séparation de la manière suivante :

Théorème 2.34 (*Séparation, formulation en termes d'ordre*)

Le préordre d'orthogonalité, quotienté par η, est anti-symétrique : c'est un ordre.

2.3. Le $\lambda\mu$-calcul de Parigot.

Étendre la correspondance de Curry-Howard à la logique classique. Pendant longtemps, la correspondance de Curry-Howard établissait un lien entre preuves en logique intuitionniste et programmes, mais les tentatives pour relier preuves en logique classique et programme se sont longtemps révélées infructueuses ; on ne parvenait par à déterminer le contenu calculatoire des preuves de la logique classique.

Le $\lambda\mu$-calcul de Parigot [Par92] est le résultat de cette volonté d'étendre la correspondance de Curry-Howard à la logique classique de manière à comprendre le contenu calculatoire des preuves de la logique classique en développant des termes de preuve pour une présentation de la déduction naturelle classique minimale.

Plusieurs propositions préexistaient (par exemple le $\lambda\mathcal{C}$-calcul de Felleisen [FH92] conçu pour fournir une théorie syntaxique du contrôle dans les langages fonctionnels et plus tard relié à la logique classique par Griffin [Gri90a], ou le système LC de Girard [Gir91] qui donne une présentation de la logique classique à partir d'une notion de polarité pour laquelle l'élimination des coupures devient confluente), mais l'approche de Parigot diffère des deux théories mentionnées précédemment. En effet, il s'agit véritablement d'un calcul (LC n'a pas vraiment de syntaxe de termes, il s'agit encore vraiment d'une logique, il ne fournit pas de calcul non-typé), et le pouvoir déductif de la logique classique n'est pas retrouvé en ajoutant un axiome supplémentaire ($\neg\neg A \Rightarrow A$ dans le cas du $\lambda\mathcal{C}$-calcul) mais en étendant les règles de déduction (et donc en définissant une version classique de la déduction naturelle). À chaque règle de la logique correspond une construction syntaxique du côté calculatoire de la correspondance et en conséquence le $\lambda\mu$-calcul a plus de constructions syntaxiques que le λ-calcul : la μ-abstraction et le nommage des termes.

En outre, le $\lambda\mu$-calcul présente l'avantage d'être une extension simple du λ-calcul et d'en préserver un grand nombre de propriétés.

2.3.1 Définition du $\lambda\mu$-calcul

Le $\lambda\mu$-calcul de Parigot est une extension du λ-calcul qui permet de faire une correspondance entre les termes du calcul typé et les preuves de la déduction naturelle classique qui soit compatible avec l'élimination des coupures. C'est donc une manière de donner un contenu calculatoire aux preuves de la logique classique. Le calcul est obtenu en étendant les λ-termes avec de nouvelles constructions.

Définition 2.35 ($\Sigma_{\lambda\mu}$)

On considère un ensemble dénombrable $\mathcal{V}$ de λ-variables (ou **variables de terme** *dénotées par $x, y, z \ldots$) et un ensemble dénombrable $\mathcal{V}_c$ de μ-variables (ou* **variables de continuation** *dénotées par $\alpha, \beta, \gamma, \ldots$), disjoints l'un de l'autre.*

Les **$\lambda\mu$-termes** *sont définis inductivement par la syntaxe suivante :*

$$t ::= \; x \mid \lambda x.t \mid (t)u \mid \mu\alpha.(t)\beta$$

On notera $\Sigma_{\lambda\mu}$ l'ensemble des termes du $\lambda\mu$-calcul de Parigot, et $\Sigma^c_{\lambda\mu}$ l'ensemble des $\lambda\mu$-termes clos. Dans le terme $\mu\alpha.(t)\beta$, la variable β est dans la portée de l'abstraction $\mu\alpha$ et peut donc être liée par cette abstraction.

On appellera **termes nommés** *les expressions de la forme $(t)\beta$; on utilisera la notation n pour dénoter des termes nommés.*

Remarque 2.36

Une autre présentation de la syntaxe des $\lambda\mu$-termes est donnée ci-dessous :

$$\begin{array}{lcl} t & ::= & x \mid \lambda x.t \mid (t)u \mid \mu\alpha.n \\ n & ::= & (t)\beta \end{array}$$

La catégorie syntaxique t définit les $\lambda\mu$-termes tandis que la catégorie syntaxique n définit les termes nommés. Cette présentation met l'accent sur les termes nommés et sur la structuration des $\lambda\mu$-termes en deux niveaux.

Remarque 2.37

Dans la définition précédente, nous n'avons pas utilisé la notation de la syntaxe traditionnelle introduite par Parigot dans l'article où il introduisait le $\lambda\mu$-calcul mais une notation légèrement différente bien que strictement équivalente. Nous avons remplacé les termes notés d'habitude $\mu\alpha.[\beta]t$ par des termes notés $\mu\alpha.(t)\beta$. Ainsi, dans le terme $\mu\alpha.(t)\alpha$, la variable de continuation α est bien liée par la μ-abstraction $\mu\alpha$.

Nous avons fait le choix de présenter le $\lambda\mu$-calcul avec la notation que l'on utilise pour le $\Lambda\mu$-calcul et qui sera justifiée plus tard pour faciliter les comparaisons entre les différents calculs étudiés, au premier rang desquels le $\Lambda\mu$-calcul.

Les règles de réduction du $\lambda\mu$-calcul étendent également les règles du λ-calcul :

Définition 2.38 (*Règles de réduction du $\lambda\mu$-calcul*)

La réduction du $\lambda\mu$-calcul, notée $\longrightarrow_{\lambda\mu}$, est la réduction induite par les quatre règles de réduction suivantes :

- $(\lambda x.t)u \longrightarrow_\beta t\{u/x\}$
- $(\mu\alpha.n)u \longrightarrow_\mu \mu\alpha.n\{(v)u\alpha/(v)\alpha\}$
- $(\mu\alpha.n)\beta \longrightarrow_\rho n\{\beta/\alpha\}$
- $\mu\alpha.(t)\alpha \longrightarrow_\theta t$ *si* $\alpha \notin FV(t)$

On remarquera que dans la réduction ρ, $(\mu\alpha.n)\beta$ n'est pas un élément de $\Sigma_{\lambda\mu}$ mais est un terme nommé.

On considère comme d'habitude les termes modulo renommage des variables liées. La substitution de la β-réduction est la substitution sans capture de variable usuelle du λ-calcul. La substitution de la ρ-réduction est définie comme la substitution précédente mais s'applique aux variables de continuation. Enfin, la substitution de la μ-réduction $\{(v)u\alpha/(v)\alpha\}$, appelée substitution structurelle, est la substitution sans capture de variable qui remplace tout sous-terme de la forme $(v)\alpha$ par $(v)u\alpha$ et qui est définie formellement ci-dessous :

Définition 2.39 (*Substitution structurelle*)

La **substitution structurelle** *est définie, sur les termes de $\Sigma_{\lambda\mu}$ et sur les termes nommés, comme suit :*
- $x\{(v)u\alpha/(v)\alpha\} = x$
- $(\lambda x.t)\{(v)u\alpha/(v)\alpha\} = \lambda x.t\{(v)u\alpha/(v)\alpha\}$
- $((t)t')\{(v)u\alpha/(v)\alpha\} = (t\{(v)u\alpha/(v)\alpha\})t'\{(v)u\alpha/(v)\alpha\}$
- $(\mu\alpha.n)\{(v)u\alpha/(v)\alpha\} = \mu\alpha.n$
- $(\mu\beta.n)\{(v)u\alpha/(v)\alpha\} = \mu\beta.n\{(v)u\alpha/(v)\alpha\}$ *(si $\beta \neq \alpha$ et $\beta \notin FV(v)$)*
- $(t)\alpha\{(v)u\alpha/(v)\alpha\} = (t\{(v)u\alpha/(v)\alpha\})v\alpha$
- $(t)\beta\{(v)u\alpha/(v)\alpha\} = (t\{(v)u\alpha/(v)\alpha\})\beta$

2.3.2 Isomorphisme de Curry-Howard avec la logique classique

Le $\lambda\mu$-calcul permet de construire une correspondance à la Curry-Howard avec le système de la déduction nature classique de Parigot présentée en section1.4.1, page 41.

Cette correspondance se fait via un système de typage pour les $\lambda\mu$-termes dont les jugements de typage ont la structure suivante :

Définition 2.40 (*Jugements de typage pour le $\lambda\mu$-calcul*)

Les jugements de typage pour le $\lambda\mu$-calcul sont de la forme :

$$x_1 : A_1, \ldots, x_m : A_m \vdash_{\lambda\mu} t \ : A|\alpha_1 : B_1, \ldots, \alpha_n : B_n$$

où $A, A_1, \ldots, A_m, B_1, \ldots, B_n$ sont des types construits à partir de variables de types, avec le connecteur $\rightarrow$.

Définition 2.41 (*Système de typage pour le $\lambda\mu$-calcul*)

On considère les règles de typage suivantes pour les $\lambda\mu$-termes :
- *typage des variables :*
$$\frac{}{\Gamma, x : A \vdash_{\lambda\mu} x \ : A|\Delta}\ Var$$
- *typage des λ-abstractions :*
$$\frac{\Gamma, x : A \vdash_{\lambda\mu} t \ : B|\Delta}{\Gamma \vdash_{\lambda\mu} \lambda x^A.t \ : A \rightarrow B|\Delta}\ \lambda\text{-}Abs$$
- *typage des applications :*
$$\frac{\Gamma \vdash_{\lambda\mu} t \ : A \rightarrow B|\Delta \qquad \Gamma \vdash_{\lambda\mu} u \ : A|\Delta}{\Gamma \vdash_{\lambda\mu} (t)u \ : B|\Delta}\ \lambda\text{-}App$$
- *typage des μ-termes :*
$$\frac{\Gamma \vdash_{\lambda\mu} t \ : B|\Delta, \alpha : A}{\Gamma \vdash_{\lambda\mu} \mu\alpha^A.(t)\beta \ : A|(\Delta, \beta : B) \setminus \alpha : A}\ \mu$$

Remarque 2.42

La règle de typage des μ-termes contient, du point de vue logique, la règle de contraction à droite, essentielle pour avoir la déduction classique.

On remarquera également que cette règle a une forme légèrement inhabituelle en ce que la portion de droite du contexte conclusion contient une soustraction ensembliste : $(\Delta, \beta : B) \setminus \alpha : A$. Cette notation est nécessaire dans la mesure où on pourrait avoir $\beta = \alpha$, qui est précisément le cas mentionné ci-dessus où on fait une contraction à droite.

Remarque 2.43 (*Second ordre*)

On peut considérer également le système de typage du second ordre en ajoutant des variables du second ordre $(X, Y, Z \dots)$, une nouvelle construction de type $(\forall^2 X.A)$ et deux nouvelles règles de typage :

$$\frac{\Gamma \vdash_{\lambda\mu^2} t \ : A|\Delta \qquad \text{avec } X \notin FV(\Gamma, \Delta)}{\Gamma \vdash_{\lambda\mu^2} t \ : \forall^2 X.A|\Delta} \forall^2 - i \qquad \frac{\Gamma \vdash_{\lambda\mu^2} t \ : \forall^2 X.A|\Delta}{\Gamma \vdash_{\lambda\mu^2} t \ : A\{B/X\}|\Delta} \forall^2 - e$$

Avec un typage à la Church, on ajoute deux constructions aux termes pour refléter ces types : une abstraction sur les types $\lambda X.t$ et une application de type $(t)B$ avec la réduction évidente $(\lambda X.t)B \longrightarrow_{\forall^2} t\{B/X\}$.

2.3.3 Propriétés principales du $\lambda\mu$-calcul

Le $\lambda\mu$-calcul est particulièrement intéressant dans la mesure où il s'agit d'une extension classique du λ-calcul qui conserve bon nombre des propriétés du λ-calcul. Nous rappelons rapidement quelques unes des principales propriétés syntaxiques du $\lambda\mu$-calcul pur et typé.

Théorème 2.44 (*Confluence du $\lambda\mu$-calcul [Par92, Py98]*)

Le $\lambda\mu$-calcul est confluent.

On notera que dans la présentation du $\lambda\mu$-calcul que nous venons de donner, il n'y a pas de réduction η. L'ajout de la réduction η fait en effet perdre la confluence, nous reviendrons sur ce point dans le chapitre 3. La preuve de l'article de Parigot [Par92] n'est pas complète ; on trouvera une preuve complète dans la thèse de Py [Py98].

Py définit également une réduction standard pour le $\lambda\mu$-calcul et démontre le ***théorème de standardisation*** correspondant [Py98].

Le calcul typé satisfait deux propriétés fondamentales : préservation du type et normalisation forte.

Théorème 2.45 (*Préservation du type [Par92]*)

Si on peut dériver $\Gamma \vdash_{\lambda\mu} t \ : A|\Delta$ et si $t \longrightarrow_{\lambda\mu} u$, alors on peut dériver $\Gamma \vdash_{\lambda\mu} u \ : A|\Delta$.

Théorème 2.46 (*Normalisation forte [Par97]*)

Si t est un $\lambda\mu$-terme typé, alors il n'y a pas de réduction infinie à partir de t.

Parigot [Par97] prouve la normalisation forte dans le cas propositionnel ainsi qu'au second ordre. Il démontre dans le même article la terminaison de la réduction μ dans le cadre non-typé.

Le $\lambda\mu$-calcul satisfait donc un grand nombre des propriétés du λ-calcul. Pourtant, le ***théorème de séparation de Böhm*** n'est pas vrai en $\lambda\mu$-calcul [DP01a] comme nous allons le voir dans le chapitre suivant.

Remarque 2.47

Nous ne détaillons pas les questions de sémantique du $\lambda\mu$-calcul car ce n'est pas l'objet de cette thèse, mais nous renvoyons à la littérature sur ce sujet [Ong96, Sel01, HS02, Lau04a, Lau04b].

2.3.4 Machine abstraite pour le $\lambda\mu$-calcul

Dans le dernier paragraphe de son premier article sur le $\lambda\mu$-calcul [Par92], Parigot suggère une machine abstraite pour le $\lambda\mu$-calcul sans la définir en détail. Depuis, plusieurs auteurs ont développé des machines abstraites pour $\lambda\mu$. Les premiers d'entre eux sont de Groote [dG98], Biermann [Bie98], ainsi que Streicher et Reus [SR98] ; on pourra également se référer au manuscrit d'Olivier Laurent [Lau03a].

Définition 2.48 ($\lambda\mu$-*KAM*)

Les états de la $\lambda\mu$-KAM étendent les états de la machine pour le λ-calcul : il s'agit de triplets consitués (i) d'un $\lambda\mu$-terme t ou un terme nommé n, (ii) d'un environnement e qui associe à une λ-variable une clôture $t[e]$ et à une μ-variable une pile de clôtures et (iii) d'une pile de clôtures K. La machine est composée des transitions suivantes :

$$\begin{array}{llrcllrl}
\langle x & [e] & K\rangle & \longrightarrow & \langle t & [e'] & K\rangle & \text{si } e(x) = t[e'] \\
\langle \lambda x.t & [e] & u[e']\cdot K\rangle & \longrightarrow & \langle t & [x = u[e']; e] & K\rangle & \\
\langle (t)u & [e] & K\rangle & \longrightarrow & \langle t & [e] & u[e]\cdot K\rangle & \\
 & & & & & & & \\
\langle \mu\alpha.c & [e] & K\rangle & \longrightarrow & \langle c & [\alpha = K; e] & \bot\rangle & \\
\langle (t)\alpha & [e] & \bot\rangle & \longrightarrow & \langle t & [e] & K\rangle & \text{si } e(\alpha) = K
\end{array}$$

2.3.5 Quelques opérateurs de contrôle en $\lambda\mu$-calcul

La $\lambda\mu$KAM peut sauvegarder des piles et les restaurer, c'est cette particularité du $\lambda\mu$-calcul, comparé au λ-calcul, qui explique que l'on puisse coder directement des opérateurs de contrôle en $\lambda\mu$-calcul.

On a ainsi les codages bien connus :

$$\boxed{\begin{array}{lcl}
\texttt{call/cc} & \triangleq & \lambda y.\mu\alpha.((y)\lambda x.\mu\delta.(x)\alpha)\alpha \\
\mathcal{C} & \triangleq & \lambda y.\mu\alpha.((y)\lambda x.\mu\delta.(x)\alpha)\beta
\end{array}}$$

On a :

$$\begin{array}{rcl}
(\texttt{call/cc})tu_1\dots u_k & \longrightarrow_{\lambda\mu} & (\mu\alpha.((t)\lambda x.\mu\delta.(x)\alpha)\alpha)u_1\dots u_k \\
 & \longrightarrow^{\star}_{\lambda\mu} & \mu\alpha.((t)\lambda x.\mu\delta.(x)u_1\dots u_k\alpha)u_1\dots u_k\alpha
\end{array}$$

et

$$\begin{array}{rcl}
(\mathcal{C})tu_1\dots u_k & \longrightarrow_{\lambda\mu} & (\mu\alpha.((t)\lambda x.\mu\delta.(x)\alpha)\beta)u_1\dots u_k \\
 & \longrightarrow^{\star}_{\lambda\mu} & \mu\alpha.((t)\lambda x.\mu\delta.(x)u_1\dots u_k\alpha)\beta
\end{array}$$

Au cours de l'exécution de $(\texttt{call/cc})t\vec{u}$, l'environnement $\vec{u}$ est sauvegardé dans le sous-terme $t' = \lambda x.\mu\delta.(x)u_1 \dots u_k\alpha$ et le calcul continue dans le même environnement. Lorsque t' arrivera en tête, il placera son premier argument dans l'environnement $\vec{u}$ sauvegardé et effacera l'environnement mis à jour depuis l'appel de `call/cc` grâce à la μ-abstraction $\mu\delta$ sans occurrence libre de δ.

L'exécution de $\mathcal{C}$ sauvegarde également l'environnement courant, mais l'efface par la même occasion, grâce à l'utilisation de la μ-variable libre ; c'est-à-dire que t commence son calcul dans un environnement réinitialisé. Lorsque $t' = \lambda x.\mu\delta.(x)u_1 \dots u_k\alpha$ arrivera à nouveau en tête, l'environnement courant sera effacé tandis que l'environnement sauvegardé sera réinstallé.

On notera, dans l'exemple ci-dessus, le fait que le terme $\mathcal{C}$ contient une variable de continuation libre.

Première partie

$\Lambda\mu$-calcul

Chapitre 3

Du $\lambda\mu$-calcul au $\Lambda\mu$-calcul

Résumé:

David et Py [Py98, DP01a] ont démontré que la propriété de séparation (ou encore théorème de Böhm) n'était pas vérifiée pour le $\lambda\mu$-calcul de Parigot [Par92].

Dans ce chapitre, nous commençons par présenter le résultat de non-séparation du $\lambda\mu$-calcul. L'analyse de ce résultat nous conduit ensuite à proposer une extension du $\lambda\mu$-calcul, le $\Lambda\mu$-calcul, pour laquelle le théorème de Böhm sera vérifié : nous introduisons le $\Lambda\mu$-calcul à la fin de ce chapitre et nous démontrerons le théorème de séparation dans le chapitre suivant.

Références : Ce chapitre reprend en partie les résultats publiés dans l'article *Separation with streams in the $\Lambda\mu$-calculus* [Sau05].

Sommaire

3.1. Introduction

Après les travaux fondateurs de Parigot et avec le développement de l'intérêt pour le $\lambda\mu$-calcul comme système standard pour étudier la correspondance de Curry-Howard avec la logique classique ainsi que pour l'étude des langages fonctionnels avec opérateurs de contrôle, il est devenu important d'explorer les propriétés syntaxiques et sémantiques du $\lambda\mu$-calcul, tant dans sa présentation typée que dans sa version pure, introduite et étudiée également dans l'article original de Parigot.

Le $\lambda\mu$-calcul conserve donc un grand nombre des propriétés du λ-calcul qu'il étend, même si les propriétés de confluence du calcul sont plus délicates. Nous allons pourtant voir qu'une propriété fondamentale du λ-calcul, la Séparation, n'est pas satisfaite par le $\lambda\mu$-calcul.

La propriété de Séparation. En 1968, Corrado Böhm [Bö68] a prouvé une propriété syntaxique essentielle du λ-calcul pur, la propriété de Séparation, plus connue sous le nom de théorème de Böhm, qui assure, comme déjà vu au chapitre précédent, que si t et u sont deux formes $\beta\eta$-normales du λ-calcul qui sont syntaxiquement différentes, il existe un contexte $C[]$ qui les sépare ; c'est-à-dire que les termes $C[t]$ et $C[u]$ se réduisent vers deux variables distinctes. Cette propriété a des conséquences importantes à la fois sur le plan sémantique et sur le plan syntaxique. D'un côté, elle assure qu'un modèle non trivial du λ-calcul ne peut pas identifier deux formes $\beta\eta$-normales différentes. De l'autre, elle signifie qu'il y a une sorte d'adéquation entre les règles de réduction et les constructions syntaxiques du langage : la démonstration du théorème de Böhm, utilisant une méthode dite de *Böhm out*, met en évidence le fait qu'il est possible d'explorer complètement les termes normaux au moyen des règles du calcul, c'est-à-dire qu'aucune partie d'un terme en forme normale n'est inaccessible.

L'échec de la séparation. Lorsqu'un langage ne satisfait pas la propriété de séparation, cela signifie que la théorie équationnelle du langage et l'équivalence observationnelle ne se correspondent pas. On peut se demander comment tâcher de réparer ce problème : peut-on trouver une variante du calcul pour laquelle théorie équationnelle et équivalence observationnelle se correspondent ?

$\lambda\mu$-calcul et séparation. Nous allons voir dans ce chapitre que le $\lambda\mu$-calcul de Parigot ne satisfait pas la séparation, ce qui a été démontré par David et Py [Py98, DP01a]. Nous commencerons par donner les grandes étapes de leur démonstration avant d'analyser ce résultat pour arriver à proposer une variante du $\lambda\mu$-calcul, le $\Lambda\mu$-calcul, qui satisfera la propriété de séparation comme nous le verrons dans le chapitre suivant.

3.2. $\lambda\mu\eta$-calcul et échec de la séparation

Le système de réduction pour le $\lambda\mu$-calcul introduit par Parigot ne contient pas la règle η. Pourtant, en $\lambda\mu$-calcul comme en λ-calcul on ne pourra évidemment pas séparer deux termes η-équivalents comme les termes t et $\lambda x.(t)x$ puisque, quel que soit le contexte dans lequel ils seront placés, ils se comporteront de la même manière.

On peut alors choisir de séparer modulo η-équivalence ou bien d'ajouter la réduction η au calcul. C'est la deuxième solution qui est choisie par David et Py pour établir la non-séparation du $\lambda\mu$-calcul : ils considèrent en fait le calcul que nous appellerons le $\lambda\mu\eta$-calcul.

3.2.1 $\lambda\mu$-calcul et η : problème de confluence

Si Parigot n'a pas considéré la règle η parmi les réduction du $\lambda\mu$-calcul, c'est sans doute en partie parce qu'elle fait échouer la propriété de confluence si on ne prend pas quelques précautions. En effet, il existe une paire critique entre les règles μ et η :

$$\mu\alpha.n \longleftarrow_\eta \lambda x.(\mu\alpha.n)x \longrightarrow_\mu \lambda x.\mu\alpha.n\,\{(v)x\alpha/(v)\alpha\}$$

David et Py résolvent cette paire critique en ajoutant une autre réduction au $\lambda\mu$-calcul, la ν-réduction :

$$\mu\alpha.n \longrightarrow_\nu \lambda x.\mu\alpha.n\,\{(v)x\alpha/(v)\alpha\} \qquad \text{si } x \notin FV(n)$$

qui est obtenue par une η-expansion immédiatement suivie d'une μ-réduction.

Définition 3.1 (*$\lambda\mu\eta$-calcul*)

Le $\lambda\mu\eta$-**calcul** *est le calcul obtenu à partir des termes de* $\Sigma_{\lambda\mu}$ *en ajoutant à la réduction de* $\lambda\mu$ *les règles* η *et* ν *:*

$$\begin{array}{llll} \lambda x.(t)x & \longrightarrow_\eta & t & \textit{si } x \notin FV(t) \\ \mu\alpha.n & \longrightarrow_\nu & \lambda x.\mu\alpha.n\,\{(v)x\alpha/(v)\alpha\} & \textit{si } x \notin FV(n) \end{array}$$

La réduction $\longrightarrow_{\lambda\mu\eta}$ *est donc la réduction* $\longrightarrow_{\beta\mu\rho\theta\eta\nu}$.

Remarque 3.2

$\lambda\mu\eta$ contient donc à la fois une η-réduction et une forme réduite d'η-expansion. On peut se demander s'il est véritablement pertinent de procéder de la sorte plutôt que de considérer directement l'η-expansion qui permettrait également d'avoir la confluence. Pour le moment, on ne fait que suivre la présentation de David et Py [Py98, DP01a], mais nous verrons en introduisant le $\Lambda\mu$-calcul à la fin de ce chapitre que cette approche est satisfaisante et plutôt élégante.

On remarquera également que la μ-réduction peut maintenant être simulée par une ν-réduction suivie d'une β-réduction.

Le $\lambda\mu\eta$-calcul satisfait plusieurs propriétés notables, au premier rang desquelles la confluence démontrée par Py dans sa thèse [Py98].

Théorème 3.3 (*Confluence de $\lambda\mu\eta$*)

Le $\lambda\mu\eta$-calcul est confluent sur les $\lambda\mu$-termes μ-clos.

Remarque 3.4

La restriction de l'énoncé de la confluence aux termes sans variable de continuation libre est nécessaire comme en témoigne la paire critique ρ/ν (si $x \notin FV(n)$) :

$$\mu\alpha.n\,\{\beta/\gamma\} \longleftarrow_\rho \mu\alpha(\mu\gamma.n)\beta \longrightarrow_\nu \mu\alpha(\lambda x.\mu\gamma.n\,\{(v)x\gamma/(v)\gamma\})\beta$$

Les deux réduits ne pouvant pas confluer si β est libre.

3.2.2 Le $\lambda\mu$-calcul n'est pas séparé

La propriété de séparation exprime que sur une partie du langage adaptée (les termes $\beta\eta$-normaux dans le cas du λ-calcul, ou les termes $\beta\eta$-normalisables), la théorie équationnelle du langage coïncide avec l'équivalence opérationnelle. Mettre en évidence l'échec de la séparation, la non-séparation, revient à montrer que pour un ensemble de termes auxquels on voudrait voir la séparation s'appliquer, théorie équationnelle et équivalence opérationnelle diffèrent.

Pour énoncer la non-séparation du $\lambda\mu$-calcul (de $\lambda\mu\eta$ en fait), il faut donc :
- identifier l'ensemble des termes à séparer $\mathcal{S}$,
- construire l'équivalence opérationnelle adaptée $\sim_{\lambda\mu\eta}$ et
- montrer qu'il existe des termes de $\mathcal{S}$ sur lesquels $=_{\lambda\mu\eta}$ et $\sim_{\lambda\mu\eta}$ diffèrent.

Formes normales canoniques. Du fait que le $\lambda\mu\eta$-calcul possède une règle d'expansion, un terme ne possède que très rarement une forme normale. Cela conduit David et Py à considérer une notion affaiblie de formes normales qu'ils appellent les formes normales canoniques :

Définition 3.5 (*Formes normales canoniques de $\lambda\mu\eta$ [Py98, DP01a]*)

Les **formes normales canoniques** *de $\lambda\mu\eta$ sont les $\lambda\mu$-termes t :*
- *en forme normale pour $\beta\eta\mu\rho\theta$;*
- *de la forme $\lambda x_1, \ldots, x_k.(y)t'_1 \ldots t'_l$ ou $\lambda x_1, \ldots, x_k.\mu\alpha((y)t'_1 \ldots t'_l)\beta$;*
- *tels que les $(t'_i)_{1 \leq i \leq l}$ soient eux-mêmes en forme normale canonique.*

L'appellation «formes normales canoniques» (*canonical normal forms* en anglais) provient de l'article de David et Py [DP01a]. Dans la thèse de doctorat de Py [Py98], cette classe de termes est appelée «formes normales étendues». Nous utiliserons la terminologie de l'article publié dans la suite de cette thèse.

Équivalence opérationnelle. Nous étendons de manière naturelle la notion de formes normales de tête et de résolubilité du λ-calcul au $\lambda\mu\eta$-calcul :

Définition 3.6 (*Forme normale de tête*)

Une **forme normale de tête** *pour $\lambda\mu\eta$ est un terme qui vérifie la syntaxe suivante :*

$$\begin{array}{rcl} h & ::= & a \mid \lambda x.h \mid \mu\alpha.(h)\beta \\ a & ::= & x \mid (a)t \end{array}$$

Définition 3.7 (*Terme résoluble*)

Un $\lambda\mu$-terme t est **résoluble** *ou* **solvable** *s'il existe un terme u en forme normale de tête tel que $t \longrightarrow^\star_{\lambda\mu\eta} u$.*

Définition 3.8 (*Équivalence opérationnelle*)

Deux $\lambda\mu\eta$-termes t, u sont **opérationnellement équivalents** *si quel que soit le contexte clos $\mathcal{C}$, $\mathcal{C}(t)$ est solvable si, et seulement si, $\mathcal{C}(u)$ est solvable.*
On notera $t \sim_{\lambda\mu\eta} u$ dans ce cas.

Énoncé de la non-séparation de $\lambda\mu\eta$. On a maintenant les outils pour énoncer l'échec de la séparation en $\lambda\mu\eta$-calcul :
- L'ensemble $\mathcal{S}_{\lambda\mu\eta}$ naturel pour tester la séparabilité du $\lambda\mu\eta$-calcul est l'ensemble des termes clos possédant une forme normale canonique ;
- La théorie équationnelle considérée est $=_{\lambda\mu\eta}$;
- L'équivalence opérationnelle utilisée pour séparer est $\sim_{\lambda\mu\eta}$.

$$W_y = \lambda x.\mu\alpha.(x) \begin{cases} \mu\beta.(x) \begin{cases} \mu\delta.(0)\alpha \\ \boxed{y} \\ \alpha \end{cases} \\ \mu\delta.(0)\alpha \\ \alpha \end{cases} \qquad avec \quad 0 = \lambda a.\lambda b.b$$

FIG. 3.1 – Terme W_y.

David et Py montrent l'échec de la séparation en exhibant un contre-exemple, c'est-à-dire une paire de termes qui sont dans $\mathcal{S}_{\lambda\mu\eta}$, qui ne sont pas $\lambda\mu\eta$-équivalents (non égalisés par la théorie équationnelle), mais qui ne sont pas séparables, c'est-à-dire qu'ils sont opérationnellement équivalents. Pour cela, ils considèrent le terme W_y défini comme $\lambda x.\mu\alpha.((x)\mu\beta.(x)U_0 y\alpha)U_0\alpha$ où $U_0 = \mu\delta.(\lambda a.\lambda b.b)\alpha$, voir également la figure 3.1 où W_y est présenté dans une notation arborescente. En effet, en notant $W_0 = W_y\{\lambda a.\lambda b.b/y\}$ et $W_1 = W_y\{\lambda a.\lambda b.a/y\}$, on a :

Théorème 3.9 (*Échec de la séparation en $\lambda\mu\eta$ [Py98, DP01a]*)

La séparation échoue en $\lambda\mu\eta$. En effet, W_0 et W_1 :
- *sont deux termes clos qui ont des formes normales canoniques ;*
- *ne sont pas $\beta\eta\mu\nu\rho\theta$-équivalents ($W_0 \neq_{\lambda\mu\eta} W_1$) ;*
- *mais sont pourtant opérationnellement équivalents ($W_0 \sim_{\lambda\mu\eta} W_1$).*

Remarque 3.10

On remarquera que, contrairement à ce qui est indiqué dans l'énoncé du théorème 4.4 de [DP01a], W_0 et W_1 ne sont pas en forme normale canonique mais se réduisent sur des formes normales canoniques W_0' et W_1' avec $W_y' = \lambda x.\lambda a.\lambda b.\mu\alpha.((x)\mu\beta.(x)U_b y\alpha)U_b\alpha$, si on note U_t le terme $\mu\delta.(t)\alpha$. Cette remarque de détail ne change rien de fondamental au résultat de [DP01a].

La démonstration du théorème passe par un *lemme du contexte* montrant que l'équivalence opérationnelle restreinte aux contextes applicatifs coïncide avec l'équivalence opérationnelle générale. Nous analyserons cette démonstration en section 3.4.

3.3. Séparation et non-séparation : quelques observations

Une fois qu'on a établi l'échec de la Séparation en $\lambda\mu$-calcul, on peut tenter de réparer ce problème et de retrouver la propriété de séparation. L'objet de cette section est d'élaborer une discussion générale à propos de la Séparation et plus spécifiquement à propos des manières de retrouver la Séparation dans un calcul où elle échoue.

3.3.1 Retrouver la Séparation

L'échec de la Séparation peut, de manière générale, être énoncé comme suit : **(i)** il existe **deux termes de $\mathcal{S}$** *(une partie du langage à laquelle s'applique la séparation)* t et s

(ii) qui **ne sont pas équivalents** *(c'est-à-dire qu'ils ne sont pas identifiés par la théorie équationnelle du langage)* **(iii)** mais qu'**aucun contexte** ne sépare *(c'est-à-dire qu'ils sont opérationnellement équivalents).*

Illustrons ces points par trois exemples très simples issus de la théorie du λ-calcul :

1. Considérons l'ensemble des termes du λ-calcul pur comme valeurs de séparation avec la $\beta\eta$-équivalence. Dans ce cas, la séparation échoue car aucun contexte ne sépare $(\Delta)\Delta$ de $(\Delta)\Delta\Delta$ (où $\Delta = \lambda x.(x)x$). Si on restreint la séparation aux termes normalisables, nous retrouvons la propriété de séparation.
$\Longrightarrow$ *Les termes sans forme normale ont été retirés de $\mathcal{S}$.*
2. Considérons le λ-calcul avec la règle β comme seule règle de réduction. La séparation échoue : aucun contexte ne sépare les deux formes normales non β-équivalentes $\lambda x, y.(x)y$ et $\lambda x.x$. Si nous ajoutons la règle η, nous retrouvons la propriété de séparation : nous avons élargi les classes d'équivalence des règles de réduction de manière à ce que le résultat soit vrai.
$\Longrightarrow$ *Les termes qui n'étaient pas séparables ont été rendus équivalents.*
3. Considérons le λ-calcul simplement typé. La séparation échoue : en effet, nous n'avons pas assez de contextes dans le calcul (des contextes simplement typés) pour explorer toutes les parties des termes et mettre en évidence opérationnellement les différences du niveau équationnel. Mais si on enlève les contraintes imposées par le système de typage (c'est-à-dire qu'on autorise davantage de contextes en permettant des contextes non typables par exemple), on retrouve la propriété de Séparation.
$\Longrightarrow$ *Les termes qui n'étaient pas équivalents ont été rendus séparables.*

Ces trois directions suggèrent des méthodes radicalement différentes pour retrouver la séparation pour un langage qui ne sépare pas (par exemple en $\lambda\mu$-calcul). Dans le cas du $\lambda\mu$-calcul, ces directions amènent les remarques suivantes :

1. Il semble difficile de rejeter le contre-exemple W_0 et W_1 en arguant du fait qu'ils ne devraient pas être pris en compte dans la séparation parce que ce ne sont pas des formes normales. Certes, on pourrait considérer que les sous-termes U_0 sont en quelque sorte des termes informationnellement vides un peu à la manière de $(\Delta)\Delta$ de $(\Delta)\Delta\Delta$ parce que la variable δ n'a pas d'occurrence libre dans le corps de U_0 (et cela joue en effet un rôle dans la preuve de séparation de David et Py), mais cela est inévitable dans tout $\lambda\mu$-terme μ-clos, à moins d'avoir un $\lambda\mu$-terme trivial où la seule occurrence d'une variable de continuation est immédiatement après son lieur (c'est-à-dire de la forme $\mu\alpha.(t)\alpha$), mais dans ce cas on se ramène essentiellement à séparer le λ-calcul dans le $\lambda\mu$-calcul (on serait conduit à considérer un ensemble $\mathcal{S}$ contenant les λ-termes et leurs θ-expansions)... ce qui est un peu faible !
2. Une autre manière serait d'essayer d'élargir la théorie équationnelle de manière à la faire coïncider avec l'équivalence opérationnelle. Cette solution est suggérée à la fin de l'article de David & Py : retrouver la propriété de Séparation en ajoutant de nouvelles règles de réduction, c'est-à-dire en rendant plus de termes équivalents (et tout particulièrement une famille de termes liés à W_y).
3. La dernière direction serait de trouver une manière d'ajouter de nouveaux contextes parmi les contextes discriminants disponibles de façon à avoir plus de possibilités d'exploration des termes et de satisfaire la Séparation de cette manière. Cela consisterait

à affiner l'équivalence opérationnelle pour la faire coïncider avec la théorie équationnelle. C'est cette dernière option que nous suivrons dans la suite, en essayant de trouver les contextes «manquants» au $\lambda\mu$-calcul.

On notera que la discussion précédente met en évidence la délicatesse et la fragilité de la propriété de séparation, ainsi que la mesure dans laquelle cette propriété témoigne d'un équilibre dans le langage :

– en effet, en cherchant à rajouter des contextes au langage, on va également ajouter des termes, qu'il faudra à leur tour séparer ;
– de même, dans la seconde option, l'élargissement de la théorie équationnelle, si elle se fait par ajout de règles de réduction, peut également élargir les classes de l'équivalence opérationnelle puisque deux termes qui n'étaient pas opérationnellement équivalents peuvent le devenir avec une réduction étendue.

3.3.2 L'exemple de la Ludique

Bien sûr, on peut aussi songer à combiner ces trois méthodes. Un exemple éclairant de cette idée peut être trouvé dans le cas des *desseins* de la Ludique [Gir01] que nous étudierons dans la suite de cette thèse. La discussion qui suit anticipe légèrement sur l'introduction de la Ludique du chapitre 11.

Girard avait besoin que la séparation soit satisfaite pour ses para-preuves (ou épreuves), de manière à les utiliser comme objets de base d'une théorie interactive. La manière dont il obtient la Séparation emprunte des éléments aux deux dernières options mentionnées ci-dessus.

À partir des preuves de $MALL$ (logique linéaire multiplicative et additive) avec un axiome généralisé (✠), Girard commence par définir les ***dessins*** qui sont des abstractions des preuves de $MAAL$ (logique multiplicative additive affine) qui sont toujours construites à partir de séquents, mais des séquents d'adresses avec des règles logiques généralisées et non plus des séquents de formules. Cette première étape correspondrait à élargir la classe des contextes/tests disponibles pour la Séparation.

Il est ensuite nécessaire de faire une étape de plus en allant vers les ***desseins*** qui sont des objets non plus centrés sur des séquents, mais sur des règles logiques (qui enregistrent la véritable information d'une preuve[1]) et qui sont des sortes de classes d'équivalence des dessins pour la règle structurelle d'affaiblissement, qui ne peut pas être observée interactivement et qui ferait donc échouer la séparation. Cette seconde étape est plus proche de la première option consistant à enrichir la théorie équationnelle.

On notera à ce point que la propriété de séparation de la Ludique est particulière car elle sépare vraiment tous les objets : deux desseins distincts sont séparables. L'enrichissement de la théorie équationnelle que l'on a mentionné au paragraphe précédent est ainsi caché dans le fait que les objets de la Ludique sont quotientés par cette équivalence.

3.3.3 Séparation typée

Disons un mot à propos des résultats de séparation typée [Sta83, Jol00, DP00, DP01b] avant de chercher une solution à l'échec de la séparation pour le $\lambda\mu$-calcul dans le cas non-typé. Il existe des versions de la propriété de séparation qui peuvent être prouvées dans le cas du λ-calcul typé et qui trouvent leurs origines dans les travaux de Statman [Sta83]. Joly en a donné une formulation dans sa thèse de doctorat [Jol00] : étant donnés deux λ-termes clos, simplement typés, t_1 et t_2 qui ont le même type et qui ne sont pas $\beta\eta$-équivalents,

[1] Au moins lorsqu'il s'agit de considérer la dynamique d'une preuve pour l'élimination des coupures.

alors, si on instancie les variables de type apparissant dans le type de t_1 et t_2 par le type des entiers de Church $(o \to o) \to (o \to o)$ alors on peut trouver un ***contexte applicatif typé*** dans lequel les termes se réduisent vers différents entiers de Church.

On retrouve dans l'énoncé même de la séparation typée des éléments de la discussion précédente : l'instanciation de types revenant à autoriser plus de contextes typés, tout en contrôlant très strictement quels contextes sont autorisés.

Ces résultats typés sont très différents des résultats de séparation habituelle à la Böhm, mais se rapprochent de questions de séparation en logique. La théorème de Böhm typé du λ-calcul peut ainsi être vu comme un théorème de séparation en déduction naturelle intuitionniste pour le fragment de l'implication. Il n'est pas surprenant qu'un résultat de séparation soit à trouver du côté de la déduction naturelle, le calcul des séquents étant pour sa part un mauvais candidat car les preuves ont trop d'information sans importance de ce point de vue. On sera en revanche plus intéressés par les systèmes du type réseaux de preuve qui quotientent beaucoup les preuves des séquents ou par les systèmes focalisés qui contraignent à choisir un représentant parmi les preuves (la séparation de la ludique est proche de cet élément).

3.3.4 Une conséquence paradoxale de la séparation ?

Il y a une conséquence de la séparation en λ-calcul qui peut sembler contradictoire, mais qui n'est en fait que paradoxale. Si on se donne deux λ-termes (en forme normale) F et B qui encodent des algorithmes de tri différents (par exemple le tri fusion et le tri à bulle pour un encodage des listes d'entiers de Church), alors il existe un contexte dans lequel ces deux termes donneront des résultats différents, quand bien même les encodages des tris sont corrects. Le point fondamental est que le contexte dans lequel on obtient cette séparation ne correspond pas à des listes d'entiers de Church.

La version typée du théorème nous dit que, étant donnés deux termes, on peut changer leur interface (en instanciant différemment les variables de type, et donc en obtenant un peu plus d'espace dans les type de données...) et qu'il est alors possible de faire calculer différemment ces algorithmes sur des exemples bien-typés. Après l'instanciation de type les fonctions de tri correctes n'ont aucune raison d'être encore des fonctions de tri correctes (ni même de correspondre à la même spécification algorithmique). Cela n'est pas contradictoire... juste paradoxal.

3.4. Analyse du résultat de David et Py : de $\lambda\mu$ à $\Lambda\mu$

Nous ne donnerons pas la preuve complète du résultat de David et Py mais renvoyons le lecteur à leurs travaux [DP01a] et plus particulièrement à la thèse de Py [Py98].

Cette section a simplement vocation à analyser leur schéma de la preuve pour comprendre ce qui manque au $\lambda\mu\eta$-calcul pour séparer, en gardant à l'esprit la discussion de la section précédente. Dans une optique de Böhm out, on va donc se demander ce qui empêche l'exploration de certaines parties de l'arbre du terme W_y.

3.4.1 Schéma de la preuve de non-séparation du $\lambda\mu\eta$-calcul

La preuve de non-séparation est organisée comme suit :

$$W t\vec{u} \quad \longrightarrow^{\star} \quad \mu\alpha.\ (t) \begin{cases} \mu\beta.\ (t) \begin{cases} \mu\delta.(0)\vec{u}\alpha \\ \boxed{\mathrm{y}} \\ \vec{u} \\ \alpha \end{cases} \\ \mu\delta.(0)\vec{u}\alpha \\ \vec{u} \\ \alpha \end{cases}$$

FIG. 3.2 – Illustration du phénomène de non-séparation du $\lambda\mu\eta$-calcul.

- on commence par prouver un lemme du contexte, à savoir que les contextes applicatifs ont le même pouvoir séparant que les contextes généraux du $\lambda\mu\eta$-calcul, c'est-à-dire que les équivalences opérationnelles définies par ces deux classes de contextes sont identiques ;
- on se restreint donc au cas où le terme W_y est placé dans un contexte applicatif de la forme $\mathcal{C} = []tu_1 \ldots u_k$. Il s'agit ainsi de montrer que si $(W_0)tu_1 \ldots u_k$ est solvable, alors $(W_1)tu_1 \ldots u_k$ l'est également ;
- le reste de la preuve consiste essentiellement à montrer que la variable y ne peut pas être placée en position de tête dans une réduction partant de $(W_y)tu_1 \ldots u_k$, quels que soient l'entier k et les termes $t, u_1, \ldots, u_k$, ce qui assure que la solvabilité de $(W_0)tu_1 \ldots u_k$ entraîne celle de $(W_1)tu_1 \ldots u_k$;
- en posant $U_0' = \mu\delta.(0)u_1 \ldots u_k\alpha$, on a (voir aussi la figure 3.2) :

$$(W_y)tu_1 \ldots u_k \longrightarrow^{\star}_{\lambda\mu\eta} \mu\alpha.((t)\mu\beta.(t)U_0'\mathbf{y}u_1 \ldots u_k\alpha)U_0'u_1 \ldots u_k\alpha$$

  c'est-à-dire que, en posant $Z = \mu\beta.(t)U_0'yu_1 \ldots u_k\alpha$, le terme $(W_y)tu_1 \ldots u_k$ se réduit sur $\mu\alpha.(t)ZU_0'u_1 \ldots u_k\alpha$;
- puisqu'on a supposé que $(W_0)tu_1 \ldots u_k$ était solvable, alors le terme $C_{zu} = (t)zuu_1 \ldots u_k$ (avec z et u des variables) est également solvable ;
- en raisonnant par cas sur la variable de tête d'une forme normale de tête de C_{zu} on constate que y ne peut être placée en position de tête dans un quelconque réduit de $(W_y)tu_1 \ldots u_k$. En effet, la variable de tête d'une forme normale de tête de C_{zu} est soit une variable liée, soit z, soit u. Dans le premier cas, cette variable est aussi la variable de tête des formes normales de tête associées à $(W_y)tu_1 \ldots u_k$ et dans les deux autres cas, on constate que $(W_y)tu_1 \ldots u_k$ se réduit également sur un terme qui n'a pas y pour variable de tête, ce qui est dû au fait que $(W_y)tu_1 \ldots u_k$ se réduit sur $\mu\alpha.(C_{zu}[Z/z, U_0'/u])\alpha$ où $Z = \mu\beta.(C_{zu}[U_0'/z, y/u])\alpha$ et donc que dans chacun des deux cas, le terme U_0' arrive en position de tête.

3.4.2 Analyse de la preuve : quels sont les contextes manquants ?

C'est le dernier cas présenté dans la section précédente qui diffère radicalement de la situation du λ-calcul. En effet, on a $\mu\alpha.(t)ZU_0'u_1 \ldots u_k\alpha = \mu\alpha.C_{zu}\{Z/z\}\{U_0'/u\}\alpha$ et $Z = \mu\beta.C_{zu}\{U_0'/z\}\{y/u\}\alpha$, c'est-à-dire que les deux occurrences du terme t en position de tête sont placées ***strictement dans le même contexte*** (à la différence de substitution des variables z et u près) de telle sorte que soit les deux occurrences de t mettent en position de tête leur premier argument, soit elles mettent toutes les deux en tête leur second argument,

$$\frac{}{\Gamma, x : A \vdash_{\lambda\mu\perp} x \ : A|\Delta}\,Var \qquad \frac{\Gamma, x : A \vdash_{\lambda\mu\perp} t \ : B|\Delta}{\Gamma \vdash_{\lambda\mu\perp} \lambda x^A.t \ : A \to B|\Delta}\,\lambda\text{-Abs}$$

$$\frac{\Gamma \vdash_{\lambda\mu\perp} t \ : A \to B|\Delta \qquad \Gamma \vdash_{\lambda\mu\perp} u \ : A|\Delta}{\Gamma \vdash_{\lambda\mu\perp} (t)u \ : B|\Delta}\,\lambda\text{-App}$$

$$\frac{\Gamma \vdash_{\lambda\mu\perp} n \ : \perp|\Delta, \alpha : A}{\Gamma \vdash_{\lambda\mu\perp} \mu\alpha^A.n \ : A|\Delta}\,\mu\text{-Abs} \qquad \frac{\Gamma \vdash_{\lambda\mu\perp} t \ : A|\Delta}{\Gamma \vdash_{\lambda\mu\perp} (t)\beta \ : \perp|\Delta, \alpha : A}\,\mu\text{-App}$$

FIG. 3.3 – Système de typage pour $\lambda\mu$ avec $\perp$ explicite.

mais il est impossible qu'une occurrence sélectionne son premier argument et l'autre son second argument, ce qui serait pourtant nécessaire pour amener y en position de tête.

Cette situation correspondrait au cas délicat de la preuve de séparation du λ-calcul où la même variable se trouve plusieurs fois en position de tête dans un chemin de Böhm out, mais dans le cas du λ-calcul, le contexte séparant a nécessairement été partiellement modifié (puisque certains des arguments sont consommés) de telle sorte qu'il est possible de faire adopter un comportement calculatoire différent aux deux occurrences de t.

C'est à ce niveau qu'il faut analyser la faiblesse du $\lambda\mu\eta$-calcul à séparer : alors qu'en λ-calcul, la λ-abstraction consomme un argument et disparaît, en $\lambda\mu\eta$-calcul, la μ-abstraction est persistante et consomme tous ses arguments, aussi nombreux soient-ils, tout en étant encore prête à en consommer autant.

Il n'est ainsi pas vraiment possible de faire disparaître une μ-abstraction : soit qu'il s'agisse de la réduction μ qui laisse intacte l'abstraction, soit qu'il s'agisse de la réduction ρ, qui certes élimine une μ-abstraction, mais lui substitue une autre abstraction à la même position : $\mu\alpha.(\mu\gamma.t)\beta \longrightarrow_\rho \mu\alpha.t\,\{\beta/\gamma\}$. La θ-réduction fait certes disparaître une μ-abstraction, mais à la condition que la variable liée n'ait qu'une occurrence dans le terme, immédiatement après l'abstraction, ce qui n'est justement pas le cas qui nous préoccupe ici.

La raison de ce mécanisme est à chercher dans le fait que la μ-abstraction et le nommage sont des constructions syntaxiquement liées dans le $\lambda\mu$-calcul de Parigot : c'est la présence du $\mu\alpha$ dans la réduction ρ à partir de $\mu\alpha.(\mu\gamma.t)\beta$ qui empêche de «saturer» une μ-abstraction d'arguments comme on peut saturer une λ-abstraction d'arguments.

Les contextes manquants. Les contextes manquants au $\lambda\mu$-calcul sont ainsi de la forme $[]xy\alpha z$, qui permettraient de faire que les deux occurrences de tête de t dans la réduction de $(W_y)tu_1 \ldots u_k$ s'exécuteraient dans des environnements différents. Nous allons donc considérer une classe de contextes applicatifs étendue comme indiqué dans le précédent paragraphe :

$$\mathcal{C} ::= [] \mid (\mathcal{C})t \mid (\mathcal{C})\alpha$$

mais également un langage qui possède plus de termes et dans lequel les constructions de μ-abstraction et de nommage ne seront plus contraintes à être concomitantes. Nous introduisons ce calcul, le $\Lambda\mu$-calcul, dans la section suivante.

Une contrainte sur les termes, réminiscente d'une contrainte de typage. On peut voir la contrainte sur la structure des termes de $\Sigma_{\lambda\mu}$ en une structuration à deux niveaux comme une réminiscence d'une contrainte de typage.

Considérons la variante présentée en figure 3.3, du système de typage proposé par Parigot que nous avions introduit au chapitre 2, définition 2.41. On notera que dans le système de la figure 3.3, la prémisse de la règle μ-Abs et la conclusion de la règle μ-App sont des jugements sur des termes nommés et non pas sur des $\lambda\mu$-termes. Dans ce système, la règle d'échange de la formule principale qui était opérée par la règle μ de la définition 2.41 est scindée en une règle d'introduction de l'absurdité et une règle d'élimination de l'absurde, la règle de contraction étant réalisée par l'introduction du $\perp$.

La structuration en deux niveaux du $\lambda\mu$-calcul avec des $\lambda\mu$-termes et des termes nommés (voir aussi la remarque 2.36) et le fait que les constructions de μ-abstraction et de nommage soient regroupées en une seule construction syntaxique peuvent ainsi être considérés comme une contrainte de typage sur l'utilisation du type $\perp$: lorsqu'on a introduit le type $\perp$ par la règle d'introduction de l'absurde, la seule chose qu'on puisse faire est de l'éliminer. Ceci sera différent en $\Lambda\mu$-calcul ou dans la version de Philippe de Groote avec règle ϵ que nous considérerons plus loin.

3.5. Le $\Lambda\mu$-calcul

Le $\Lambda\mu$-calcul est une extension du $\lambda\mu$-calcul avec plus de termes et plus de réductions.

Définition 3.11 ($\Sigma_{\Lambda\mu}$)

On considère un ensemble dénombrable $\mathcal{V}_t$ de λ-variables (ou **variables de termes** *dénotées par $x, y, z \dots$) et un ensemble dénombrable $\mathcal{V}_s$ de μ-variables (ou* **variables de stream** *dénotées par $\alpha, \beta, \gamma, \dots$), disjoints l'un de l'autre.*

Les **$\Lambda\mu$-termes** *sont définis inductivement par la syntaxe suivante :*

$$t ::= \ x \mid \lambda x.t \mid (t)u \mid \mu\alpha.t \mid (t)\alpha$$

On notera $\Sigma_{\Lambda\mu}$ l'ensemble des termes du $\Lambda\mu$-calcul et $\Sigma^c_{\Lambda\mu}$ l'ensemble des termes clos.

On notera que, puisque $\alpha \notin \Sigma_{\Lambda\mu}$, la notation $(t)\alpha$ n'est pas ambiguë avec la notation $(t)u$. On remarquera également que $\Sigma_{\lambda\mu} \subsetneq \Sigma_{\Lambda\mu}$ et que les termes nommés de la définition 2.35 sont des éléments de $\Sigma_{\Lambda\mu}$. En outre, des termes comme $\mu\alpha.\mu\beta.t$ ou $\lambda x.(t)\alpha y$ sont dans $\Sigma_{\Lambda\mu}$.

Définition 3.12 (*Règles de réduction du $\Lambda\mu$-calcul*)

La réduction du $\Lambda\mu$-calcul, notée $\longrightarrow_{\Lambda\mu}$, est la réduction induite par les cinq règles de réduction suivantes :

$$\begin{array}{llll}
(\lambda x.t)u & \longrightarrow_{\beta_T} & t\,\{u/x\} & \\
\lambda x.(t)x & \longrightarrow_{\eta_T} & t & \text{si } x \notin FV(t) \\
(\mu\alpha.t)\beta & \longrightarrow_{\beta_S} & t\,\{\beta/\alpha\} & \\
\mu\alpha.(t)\alpha & \longrightarrow_{\eta_S} & t & \text{si } \alpha \notin FV(t) \\
\mu\alpha.t & \longrightarrow_{fst} & \lambda x.\mu\alpha.t\,\{(v)x\alpha/(v)\alpha\} & \text{si } x \notin FV(t)
\end{array}$$

Remarque 3.13

Les règles β_T, η_T, β_S, η_S, fst correspondent respectivement à $\beta, \eta, \rho, \theta$ et ν.

Comme déjà remarqué en section 3.1, la μ-réduction peut être simulée par une fst-réduction suivie d'une β_T-réduction :

$$(\mu\alpha.t)u \longrightarrow_{fst} (\lambda x.\mu\alpha.t\,\{(v)x\alpha/(v)\alpha\})u \longrightarrow_{\beta_T} \mu\alpha.t\,\{(v)u\alpha/(v)\alpha\}\,.$$

Si David et Py conservaient la μ-réduction dans leur analyse du $\lambda\mu$-calcul avec extensionnalité, nous n'inclurons pas cette règle parmi les réductions du $\Lambda\mu$-calcul.

On considère les notations suivantes pour désigner divers sous-systèmes de réductions ou équivalences concernant le $\Lambda\mu$-calcul :

Définition 3.14 ($\boldsymbol{\beta}$, $\boldsymbol{\beta^{var}}$, $\boldsymbol{\eta}$)

On considère les sous-systèmes suivants de la $\Lambda\mu$-réduction ou de la $\Lambda\mu$-équivalence :
- *$\boldsymbol{\beta}$ est le sous-système constitué des réductions β_T et β_S ;*
- *$\boldsymbol{\eta}$ est le sous-système constitué des réductions η_T et η_S ;*
- *$\boldsymbol{\beta}$fst est le sous-système $\beta_T\beta_S$fst et $\boldsymbol{\beta\eta}$fst désigne la $\Lambda\mu$-réduction complète ;*
- *$\boldsymbol{\beta}^{var}$ est le sous-système de $\boldsymbol{\beta}$ qui ne réduit un $\boldsymbol{\beta}$-redex que lorsque l'argument de celui-ci est une variable : $(\lambda x.\mu\alpha.t)y\beta \longrightarrow^{\star}_{\boldsymbol{\beta}^{var}} t\,\{y/x\}\,\{\beta/\alpha\}$;*
- *$=_{\Lambda\mu}$ est l'équivalence associée à $\longrightarrow_{\Lambda\mu}$.*

Plus de termes, plus de place pour η-expanser. Avec le $\Lambda\mu$-calcul, on peut construire de nouveaux termes (qui n'avaient pas d'équivalent dans le $\lambda\mu$-calcul de Parigot) comme par exemple $\mu\alpha.\mu\beta.(t)$ ou $(t)\alpha x\beta$. En effet, de manière intuitive, l'η-expansion et la θ-expansion étaient interdites entre une μ-abstraction et une construction de nommage dans le $\lambda\mu$-calcul, puisqu'il s'agissait d'une seule construction syntaxique dans le calcul de Parigot. Avec le $\Lambda\mu$-calcul, on a la liberté de faire des η-expansions (avec η_T ou η_S) n'importe où dans le terme : $\mu\alpha.\mu\beta.(t)\beta$ ou $\mu\alpha.\lambda x.(t)x$ sont des termes parfaitement autorisés tandis que ni $\mu\alpha.\mu\beta.(n)\beta$ ni $\mu\alpha.\lambda x.(n)x$ n'étaient des termes du $\lambda\mu$-calcul de Parigot.

Ce changement est extrêmement important pour la Séparation : pendant le processus de Séparation, l'η-expansion joue un rôle essentiel (ou, pour le dire différemment, le fait de pouvoir saturer un terme d'arguments est essentiel pour satisfaire la séparation).

3.5.1 Interprétation comme un calcul de stream

Nous donnons maintenant quelques explications pour interpréter intuitivement les réductions de $\Lambda\mu$. η_S est évidemment une règle d'extensionnalité tandis que β_S a, en un certain sens, un contenu plus calculatoire (même si elle ne correspond qu'à une variante très faible de la β_T-réduction puisqu'il ne s'agit que de renommage de variable). En outre, la règle *fst* relie les variables de terme et les variables de stream et c'est cette connection qui rend le calcul intéressant ; sans cette règle, le $\Lambda\mu$-calcul serait juste un calcul avec deux types d'abstraction dont l'une d'elles serait particulièrement faible. Avec la règle *fst*, les deux types de construction du langage interagissent et on peut interpréter les éléments de $\mathcal{V}_s$ comme prenant leurs valeurs dans des streams (ou listes infinies) de termes et la règle *fst* peut être vue comme une manière d'accéder au premier terme de la stream (d'où son nom), une sorte d'opération `pop`. De manière à rendre ce point tout à fait clair, considérons des dérivations *fst* pour un terme μ-abstrait :

$$\begin{array}{rll} \mu\alpha_0.t & \longrightarrow_{fst} & \lambda x_1.\mu\alpha_1.t\,\{(u)x_1\alpha_1/(u)\alpha_0\} \\ & \longrightarrow^{\star}_{fst} & \lambda x_1\ldots\lambda x_n.\mu\alpha_n.t\,\{(u)x_1\ldots x_n\alpha_n/(u)\alpha_0\} \end{array}$$

Ces dérivations peuvent être arbitrairement longues de telle sorte que $\mu\alpha$ peut être vu comme une sorte de λ-abstraction infinie. Il ne s'agit pas exactement de cela, mais plutôt d'une abstraction sur des streams de termes (ou des piles paresseuses...). Ceci avait déjà été remarqué par Parigot dans son article original sur le $\lambda\mu$-calcul [Par92] : *«The operator μ looks like a λ having potentially infinite number of arguments»*. Cette analogie est déjà valide avec le $\lambda\mu$-calcul, mais elle ne prend de l'importance qu'avec le $\Lambda\mu$-calcul dans lequel l'abstraction de streams et l'application de streams sont deux constructions bien identifiées et indépendantes l'une de l'autre.

Une notation pour les streams. On mettra en valeur cette interprétation grâce à la notation suivante :

Définition 3.15 (*Notation de Stream*)

Pour raccourcir l'écriture des $\Lambda\mu$-termes, nous allons adopter les conventions suivantes : quand cela n'introduit pas d'ambiguïté, nous noterons une séquence d'abstractions de la forme : $\lambda x_1\ldots\lambda x_n.\mu\alpha.t$ de manière abrégée $\Lambda\mathcal{S}.t$ et une séquence d'applications de la forme $(t)u_1\ldots u_m\alpha$ en $(t)\mathcal{S}$. On appellera ces deux notations **Abstraction de Stream** *et* **Application de Stream**.

3.5.2 Formes normales de tête et contextes applicatifs de $\Lambda\mu$

On définit, pour $\Lambda\mu$, les formes normales de tête ainsi que l'analogue des formes normales canoniques de la définition 3.5 :

Définition 3.16 (*Formes normales de tête*)

Une **forme normale de tête** *pour $\Lambda\mu$ est un terme qui vérifie la syntaxe suivante :*

$$\begin{array}{rcl} h & ::= & a \mid \lambda x.h \mid \mu\alpha.h \\ a & ::= & x \mid (a)t \mid (a)\alpha \end{array}$$

Un $\Lambda\mu$-terme t est **résoluble** *ou* **solvable** *s'il existe un terme u en forme normale de tête tel que $t \longrightarrow^{\star}_{\Lambda\mu} u$.*

En utilisant la notation de stream, les $\Lambda\mu$-formes normales de tête sont les termes de la forme

$$\Lambda\mathcal{S}_1\ldots\Lambda\mathcal{S}_k.\lambda x_1\ldots\lambda x_l.(x)\mathcal{S}'_1\ldots\mathcal{S}'_{k'}t_1\ldots t_{l'}$$

où les $\mathcal{S}'_1,\ldots\mathcal{S}'_{k'},t_1,\ldots t_{l'}$ sont quelconques.

Il nous sera utile, dans la suite, de pouvoir parler de réduction de tête et de réduction gauche en $\Lambda\mu$. On introduit pour cela la notion de ***pré-redex*** (ou encore $\Lambda\mu$-***coupure***).

Définition 3.17 (*pré-redex, $\Lambda\mu$-coupure*)

Soit t un $\Lambda\mu$-terme. Un **pré-redex** *(ou $\Lambda\mu$-coupure) de t est un sous-terme de t de l'une des formes suivantes :*

- $(\mu\alpha.s)\beta$;
- $(\lambda x.s)u$;
- $(\mu\alpha.s)u$;
- $(\lambda x.s)\alpha$.

Lorsque la variable de stream intervenant dans le pré-redex du quatrième type est libre dans t, on parlera de **pré-redex bloqué**.

Pour les quatre types de pré-redex, on parlera respectivement de **pré-redex (ou $\Lambda\mu$-coupure) de type $(\mathcal{S})\mathcal{S}$, $(\mathcal{T})\mathcal{T}$, $(\mathcal{S})\mathcal{T}$ et $(\mathcal{T})\mathcal{S}$.**

Si t est un $\Lambda\mu$-terme contenant un pré-redex $\mathfrak{r}$ et si $\mathfrak{r}$ n'est pas bloqué, alors soit on peut appliquer une β_T ou une β_S-réduction, soit on peut appliquer une *fst*-réduction qui créera un β_T-redex. Un $\Lambda\mu$-terme μ-clos ne contient jamais de pré-redex bloqué.

Définition 3.18 (*Réduction de tête*)

Tout $\Lambda\mu$-terme est de l'une des deux formes suivantes :

- *soit t est en forme normale de tête ;*
- *soit t est de la forme $\Lambda\mathcal{S}_1 \dots \Lambda\mathcal{S}_k.\lambda x_1 \dots \lambda x_l.(r)\mathcal{S}'_1 \dots \mathcal{S}'_{k'}t_1 \dots t_{l'}$, où r est un prérédex, bloqué ou non. On parlera de* **pré-redex de tête** *pour désigner r.*

La **réduction de tête** *consiste à toujours réduire le pré-redex de tête s'il existe et n'est pas bloqué :*

- *Si le pré-redex de tête est un $\boldsymbol{\beta}$-redex, on applique une $\boldsymbol{\beta}$-réduction ;*
- *Si le pré-rédex de tête n'est pas un $\boldsymbol{\beta}$-redex mais n'est pas bloqué, on applique une fst-réduction au lieur de la variable de stream impliquée dans le pré-redex ;*
- *Si le pré-redex de tête est bloqué ou s'il n'y a pas de pré-redex de tête, on s'arrête.*

Remarque 3.19

La réduction de tête consiste en une sous-réduction de $\longrightarrow_{\beta fst}$ et ne fait pas intervenir les règles d'extensionalité η_T et η_S.

Proposition 3.20

Si t est un $\Lambda\mu$-terme μ-clos, la réduction de tête, si elle termine, termine sur une forme normale de tête.

Démonstration : En effet, d'après les hypothèses de la proposition, un pré-redex n'est jamais bloqué dans le cas considéré, le seul cas où la réduction de tête termine est donc celui où il n'y a pas de pré-redex en tête, c'est-à-dire au cas où on a trouvé une forme normale de tête.

$\square$

On notera que, réciproquement, si un terme t possède une forme de tête, la réduction de tête termine.

Définition 3.21 (*Réduction gauche*)

La réduction gauche consiste à toujours réduire le pré-redex non-bloqué le plus à gauche. On notera $\longrightarrow_g$ la réduction gauche.

Plus de contextes. Bien sûr, comme nous le remarquions à la fin de la section précédente, le $\Lambda\mu$-calcul dispose aussi de contextes étendus. On s'intéressera en particulier aux contextes applicatifs de $\Lambda\mu$ quand nous démontrerons le théorème de Böhm pour le $\Lambda\mu$-calcul au chapitre 4 :

Définition 3.22 (*Contextes applicatifs*)

Les contextes applicatifs du $\Lambda\mu$-calcul sont définis comme suit :

$$\mathcal{C} ::= [] \mid (\mathcal{C})t \mid (\mathcal{C})\alpha$$

En utilisant la notation de stream, les contextes applicatifs de $\Lambda\mu$ sont de la forme :

$$[]\mathcal{S}_1 \dots \mathcal{S}_k t_1 \dots t_l$$

3.5.3 Formes normales canoniques de $\Lambda\mu$

Les formes normales canoniques sont des formes normales de tête particulières :

Définition 3.23 (*Formes normales canoniques*)

Un $\Lambda\mu$-terme t est en **forme normale canonique** *(aussi noté FNC) s'il est $\boldsymbol{\beta\eta}$-normal et s'il ne contient pas de sous-terme de la forme $(\lambda x.u)\alpha$ ou $(\mu\alpha.u)v$.*

C'est ainsi un terme qui ne contient ni pré-redex ni $\boldsymbol{\eta}$-redex.

Remarque 3.24

Dans le cas μ-clos, les formes normales canoniques sont aussi les $\Lambda\mu$-termes $\boldsymbol{\beta\eta}$-normaux tels qu'aucune fst-réduction ne crée de $\boldsymbol{\beta\eta}$-redex.

Dans le cas non μ-clos, cela reviendrait à demander qu'aucune fst-réduction appliquée à une μ-clôture du terme t (c'est-à-dire un terme de la forme $\mu\alpha_1 \dots \mu\alpha_k.t$, où $\alpha_1 \dots \alpha_k$ sont les variables de stream libres de t.) ne crée de $\boldsymbol{\beta\eta}$-redex.

Proposition 3.25

Soit $t \in \Sigma_{\Lambda\mu}$. La réduction gauche à partir de t, si elle termine, s'arrête sur un terme qui se $\boldsymbol{\eta}$-réduit sur une forme normale canonique.

On considère une restriction de la règle *fst* :

Définition 3.26 (*fst*$^-$)

fst$^-$ est la restriction de fst aux fst-redex $t = \mu\alpha.t'$ qui sont appliqués à un terme u (qui sont des pré-redex de type $(\mathcal{S})\mathcal{T}$) ou bien tels que t' contienne au moins un sous-terme $(\lambda x.u)\alpha$ (c'est-à-dire un pré-redex de type $(\mathcal{T})\mathcal{S}$ impliquant la variable α).

On notera $\longrightarrow_{\Lambda\mu^-}$ pour $\longrightarrow_{\boldsymbol{\beta\eta} fst^-}$ et $=_{\Lambda\mu^-}$ pour l'équivalence associée à $\longrightarrow_{\Lambda\mu^-}$.

Proposition 3.27

Les équivalences $=_{\Lambda\mu}$ et $=_{\Lambda\mu^-}$ sont égales.

Démonstration : L'inclusion $=_{\Lambda\mu^-} \subset =_{\Lambda\mu}$ est évidente.

L'inclusion $=_{\Lambda\mu} \subset =_{\Lambda\mu^-}$ est obtenue en transformant une dérivation de $t =_{\Lambda\mu} t'$ en une dérivation de $t =_{\Lambda\mu^-} t'$ en utilisant si besoin des η_T-équivalences pour passer de $u \longrightarrow_{fst} v$ à $u \longrightarrow_{\eta_T^{exp}} u' \longrightarrow_{fst^-} v' \longrightarrow_{\eta_T} v$. Supposons qu'on ait $t \longrightarrow_{fst} u$ avec

$t \not\longrightarrow_{fst^-} u$. En notant $t = C[\mu\alpha.t']$ et $u = C[\lambda y.\mu\beta.t' \{(v)y\beta/(v)\alpha\}]$, on a :

$$\begin{array}{rcl} t & = & C[\mu\alpha.t'] \\ & =_{\eta_T} & C[\lambda x.(\mu\alpha.t')x] \\ & =_{fst^-} & C[\lambda x.(\lambda y.\mu\beta.t' \{(v)y\beta/(v)\alpha\})x] \\ & =_{\eta_T} & C[\lambda y.\mu\beta.t' \{(v)y\beta/(v)\alpha\}] \qquad = \quad u. \end{array}$$

□

Proposition 3.28

Les $\Lambda\mu$-termes μ-clos en forme normale canonique sont exactement les formes normales de $\Lambda\mu^-$.

Démonstration : Immédiat d'après la définition des formes normales canoniques.

□

On établira la propriété suivante lorsque l'on aura la confluence de $\Lambda\mu$. Il s'agit d'une propriété analogue à l'unicité de la forme $\beta\eta$-normale dans le λ-calcul :

Proposition 3.29

Deux termes μ-clos en forme normale canonique sont $\Lambda\mu$-équivalents si, et seulement si, ils sont fst-équivalents :

$$\text{Si } t, u \in \Sigma^c_{\Lambda\mu} \text{ sont en FNC, alors } t =_{\Lambda\mu} u \Leftrightarrow t =_{fst} u.$$

Démonstration : Voir le chapitre sur la confluence du $\Lambda\mu$-calcul, proposition 5.27.

□

Bien que la réduction $\longrightarrow_{\Lambda\mu^-}$ semble plus satisfaisante que $\longrightarrow_{\Lambda\mu}$ du point de vue des formes normales canoniques, on n'utilisera pas cette réduction car elle n'est pas confluente.

En effet, il existe plusieurs formes normales canoniques $=_{\Lambda\mu}$-équivalentes et donc $=_{\Lambda\mu^-}$-équivalentes par la proposition 3.27 or nous avons vu qu'il s'agissait de formes normales pour $\longrightarrow_{\Lambda\mu^-}$: il n'y a donc pas la propriété d'unicité de la forme normale dans $\longrightarrow_{\Lambda\mu^-}$. Plus précisément, la non-confluence de $\Lambda\mu^-$ est illustrée par l'exemple :

$$\begin{array}{ccc} \mu\alpha.\mu\beta.(\lambda x.(y)x)\alpha & \xrightarrow[\eta_T]{} & \mu\alpha.\mu\beta.(y)\alpha \\ \big\downarrow{\scriptstyle fst^-} & & \\ \lambda z.\mu\alpha.\mu\beta.(\lambda x.(y)x)z\alpha & & \\ \big\downarrow{\scriptstyle \beta_T} & & \\ \lambda z.\mu\alpha.\mu\beta.(y)z\alpha & & \end{array}$$

où les deux termes obtenus, formes normales canoniques, ne peuvent confluer dans le système $\Lambda\mu^-$ puisqu'ils sont en forme normale.

Remarque 3.30

Dans un article où il étudie les traductions CPS du $\lambda\mu$-calcul [dG94], Philippe de Groote introduit une extension du $\lambda\mu$-calcul, que nous appellerons $\lambda\mu_{dG}$ et qui est très proche du $\Lambda\mu$-calcul excepté qu'il ne comporte ni η_T ni fst ; $\lambda\mu_{dG}$ peut être considéré comme un fragment de $\Lambda\mu$. Par ailleurs, dans son article suivant sur le sujet [dG98], concernant les machines abstraites pour le $\lambda\mu$-calcul, il considère également des termes de $\Sigma_{\Lambda\mu}$ mais ajoute une nouvelle réduction, $\longrightarrow_\epsilon$. Nous reviendrons sur ce calcul, $\lambda\mu\epsilon$, par la suite.

Luke Ong [Ong96] considère également une variante du $\lambda\mu$-calcul construite sur $\Sigma_{\Lambda\mu}$ qui est proche de $\Lambda\mu$ mais est présentée comme une théorie équationnelle. Ong n'autorise pas les variables de stream à prendre le type $\perp$, en conséquence de quoi il n'y a pas de terme typé de la forme $\mu\alpha.\mu\beta.t$, mais des termes de la forme $\mu\alpha.(\lambda x.(t)\beta)t'$.

3.6. Conclusion

Dans ce chapitre, nous avons présenté le $\lambda\mu\eta$-calcul et le résultat de non-séparation du $\lambda\mu$-calcul, établi par David et Py via la définition de formes normales canoniques et l'étude de l'équivalence opérationnelle de $\lambda\mu\eta$. En analysant ce résultat de non-séparation, nous avons proposé une extension du $\lambda\mu$-calcul de Parigot, le $\Lambda\mu$-calcul, dont nous avons donné les définitions principales dans la dernière partie de ce chapitre.

Le reste de cette partie de la thèse sera consacré à l'étude des propriétés de ce calcul :

- Nous commencerons par établir au chapitre 4 un théorème de Böhm pour le $\Lambda\mu$-calcul ;
- Nous montrerons la confluence du calcul au chapitre 5, ce qui nous fournira comme corollaire une preuve de confluence de $\lambda\mu\eta$;
- Au chapitre 6, nous proposerons ensuite un nouveau système de types pour le $\Lambda\mu$-calcul, $\Lambda_{\mathcal{S}}$, qui met en avant l'interprétation de streams déjà évoquée dans le présent chapitre ;
- Le chapitre 7 sera consacré à une analyse du $\Lambda\mu$-calcul à l'aide de réseaux adaptés des réseaux de preuve de MELL, les SANE ;
- Nous établirons enfin au chapitre 8 des résultats comparant les diverses versions du $\lambda\mu$-calcul usuellement considérées dans la littérature et divers résultats renforçant l'interprétation de stream du $\Lambda\mu$-calcul déjà discutée dans le présent chapitre ainsi que des connections avec le contrôle délimité.

Chapitre 4

Théorème de Böhm pour le $\Lambda\mu$-calcul

Résumé:

Dans ce chapitre, nous démontrons le théorème de Böhm pour le $\Lambda\mu$-calcul. La preuve que nous présentons est inspirée de la technique de Joly [Jol00] pour le λ-calcul pur plutôt que des preuves classiques [Kri90].

Références : Les résultats de ce chapitre ont été publiés dans l'article *Separation with streams in the $\Lambda\mu$-calculus* [Sau05]

Sommaire

L'objectif de ce chapitre est de démontrer le théorème suivant :

Théorème 4.1 (***Théorème de séparation pour le $\Lambda\mu$-calcul***)

Soient t et u deux $\Lambda\mu$-termes clos en forme normale canonique. Alors si $t \neq_{\Lambda\mu} u$, il existe un contexte $\mathcal{C}$ tel que $\mathcal{C}[t] \longrightarrow^{\star}_{\Lambda\mu} 1$ tandis que $\mathcal{C}[u] \longrightarrow^{\star}_{\Lambda\mu} 0$.

De plus, le contexte $\mathcal{C}$ peut être choisi applicatif, de la forme $\mathcal{C} = []\mathcal{S}_1 \ldots \mathcal{S}_k w_1 \ldots w_l$.

Il existe plusieurs preuves du théorème de Böhm en λ-calcul pur. Outre l'article de Böhm [Bö68], on pourra consulter les preuves qu'en donnent Krivine [Kri90], Ronchi della Rocca [dR94] ou plus récemment la preuve proposée par Joly dans son doctorat [Jol00].

Le théorème de Böhm a ceci d'intéressant qu'il permet de construire effectivement un contexte séparant ; il ne fait pas que dire que la théorie équationnelle est maximale. Nous illustrerons ainsi notre résultat en séparant les termes W_0 et W_1 de David et Py à la fin du chapitre.

La preuve que nous donnons dans cette section est inspirée de la preuve de Thierry Joly [Jol00] pour le λ-calcul. Avant d'entrer dans les détails techniques de la preuve de Séparation, nous décrivons les grands traits de cette preuve en termes informels.

4.1. Organisation de la preuve

Induction sur la taille des termes en FNCA. Étant donnés deux termes clos t et u que nous souhaitons séparer, nous allons raisonner par induction sur la ***taille*** des termes et par cas sur leur ***structure***. Nous devons donc considérer le cas où t et u ont des variables libres de manière à pouvoir appliquer l'hypothèse d'induction. Le cas principal de la preuve est celui où t et u sont de la forme : $(x)\mathcal{T}_1 \ldots \mathcal{T}_m t^1_{m+1} \ldots t^k_{m+1}$ et $(y)\mathcal{U}_1 \ldots \mathcal{U}_n u^1_{n+1} \ldots u^l_{n+1}$, on parle de ***forme normale canonique applicative (FNCA)***.

Cas de base. Si $x \neq y$ ou $(m,k) \neq (n,l)$, nous sommes dans le cas où il est facile de construire un contexte séparant et où il n'est pas nécessaire d'appliquer une induction aux sous-termes : nous sommes dans un cas où la différence entre les deux termes est au niveau de la tête des termes et où il n'y a donc pas besoin de descendre dans la structure arborescente des termes pour la trouver.

Cas inductif. La séparation devient plus complexe dans le cas où $x = y$, $m = n$ et $k = l$: il faudra appliquer une hypothèse d'induction pour pouvoir séparer. Dans ce cas, en effet, nous devons utiliser le fait que $t \neq_{\Lambda\mu} u$ pour trouver des sous-termes t' et u' qui ont la même ***position*** dans les arbres de terme de t et u et qui ne sont pas équivalents puis séparer t' et u' à l'aide d'un contexte $\mathcal{C}'$ et montrer que l'on peut utiliser la séparation de t' et u' pour séparer t et u.

Trouver des sous-termes non-équivalents. Dans le cas du λ-calcul, il est évident que l'on peut trouver de tels termes, mais en $\Lambda\mu$-calcul, les choses se compliquent légèrement. En effet, il pourrait arriver que les termes soient de la forme $(x)\alpha$ et $(x)y\beta$ (ou même de la forme $(x)\alpha$ et $(x)\beta$). Dans ce cas, bien que les termes ne soient pas équivalents, il n'y a pas de position satisfaisant la condition précédente et donc pas de sous-termes de t' et u' à sélectionner, puisque α et β (et plus généralement $\mathcal{T}_i$ et $\mathcal{U}_i$) ne sont pas des termes du $\Lambda\mu$-calcul.

Nous aurions dû effectuer une *fst*-réduction sur les abstractions $\mu\alpha$ et $\mu\beta$ lorsque les termes étaient encore clos (c'est-à-dire avant d'entamer la séparation) de telle sorte que nous nous serions trouvés en présence des termes $(x)v_\alpha\alpha$ et $(x)yv_\beta\beta$ pour lesquels il n'y a pas de problème pour trouver des sous-termes séparables. Ce point est formalisé dans le ***lemme du sous-terme*** (Lemme 4.25) qui donne une manière de prévenir le problème mentionné ci-dessus.

Sélection des sous-termes. Une fois que deux tels termes t' et u' ont été trouvés, on doit les sélectionner pour les amener en position de tête et finalement les séparer grâce à l'hypothèse d'induction. Cette sélection est réalisée en affectant un terme P à la variable de tête commune x, où P serait un ***sélecteur de sous-terme*** (voir la définition 4.6), noté $\Phi^{\mathcal{P}}_{(i,j)}$, pour des termes t et u possédant des sous-termes aux positions $\mathcal{P}$ et devant sélectionner les sous-termes t' et u' qui sont à la position (i, j).

Surcharge de la substitution de la variable de tête. Néanmoins, il n'est pas si facile de séparer les termes t et u puisque en plus de la sélection des sous-termes t' et u' par le terme P assigné à la variable de tête x, nous devons également séparer t' et u' or la variable x pourrait avoir une occurrence dans les sous-termes t' et u' et donc se voir déjà attribuer un terme P' pour une partie auxiliaire du travail de séparation.

Nous aurons besoin d'un lemme auxiliaire pour assurer que l'on peut assigner à x un terme qui jouera à la fois *(i)* le rôle de sélection des sous-termes particuliers t' et u' et *(ii)* de séparation de t' et u' ; il s'agira du ***lemme de la paire paramétrique*** (Lemme 4.19). Le but de ce lemme est de garantir qu'il est possible d'assigner un terme plus structuré à la variable x (une sorte de paire de termes) sans perturber le procédé de séparation qui doit avoir lieu dans les sous-termes t' et u'.

Mise en forme applicative. Enfin, il nous faut traiter le cas où les termes ne sont pas de la forme $(z)\mathcal{S}_1 \dots \mathcal{S}_m s_1 \dots s_k$. Nous devrons les réduire au cas précédent d'une manière qui soit compatible avec l'induction, c'est-à-dire qui se fasse à taille constante ou en diminuant la taille, et avec le lemme du sous-terme. Il faut donc définir une mesure sur les formes normales canoniques du $\Lambda\mu$-calcul qui prenne en compte cette réduction de manière à ce que les termes ne voient pas leur taille augmenter pendant cette étape. Le point qui sera crucial dans cette étape est la stratification des formes normales de tête en deux niveaux : un niveau d'abstractions (λ-abstractions ou μ-abstractions) et un niveau d'applications (à la fois de termes et de streams). La non-croissance par mise en forme applicative sera l'objet du lemme 4.12.

4.2. Définitions préliminaires

4.2.1 Taille, positions et sélecteur de sous-terme

Dans cette section, nous introduisons quelques définitions et termes particuliers et démontrons des lemmes dont nous avons besoin pour la preuve du théorème de séparation. Nous commençons par définir la taille d'un $\Lambda\mu$-terme que nous utiliserons dans notre raisonnement par induction.

Définition 4.2 (*Taille des $\Lambda\mu$-termes, taille des streams*)

Nous définissons inductivement une taille $\mathcal{T}$ sur les formes normales canoniques du $\Lambda\mu$-calcul et une taille $\mathcal{T}'$ sur les streams de formes normales canoniques :

- $\mathcal{T}(x) = 0$
- $\mathcal{T}(\Lambda\mathcal{S}_1 \ldots \Lambda\mathcal{S}_k.\lambda x_1 \ldots \lambda x_l.t) = 1 + \mathcal{T}(t)$ *si t n'est pas une abstraction et* $k + l > 0$
- $\mathcal{T}((x)\mathcal{S}_1 \ldots \mathcal{S}_m u_1 \ldots u_n) = 1 + \sum_{i=1}^{m} \mathcal{T}'(\mathcal{S}_i) + \sum_{i=1}^{n} \mathcal{T}(u_i) \quad$ *si* $(m, n) \neq (0, 0)$
- $\mathcal{T}'(\mathcal{S}) = \sum_{i=1}^{m} \mathcal{T}(t_i)$ *si $\mathcal{S}$ est la stream* $\mathcal{S} = t_1 \ldots t_m\alpha$.

En plus de la taille d'un terme, nous aurons besoin de parler précisément des sous-termes de certains termes (puisque la séparation, par Böhm Out, est essentiellement une exploration des termes). Pour cela, la longueur d'une stream, les positions des sous-termes et les contextes applicatifs minimaux seront des notions d'une grande importance.

Définition 4.3 (*Longueur d'une Stream, longueur d'un contexte applicatif*)

- *La* **longueur d'une stream** $\mathcal{S} = t_1 \ldots t_n\alpha$ *est* $\sharp\mathcal{S} = n$. *Si* $\mathcal{S} = \alpha$, $\sharp\mathcal{S} = 0$;
- *La* **longueur d'un contexte applicatif** $\mathcal{C} = []\mathcal{S}_1 \ldots \mathcal{S}_m t_1 \ldots t_n$ *est* $\sharp\mathcal{C} = n + \sum_{i=1}^{m} \sharp\mathcal{S}_i$.

Définition 4.4 (*Position d'un sous-terme*)

Étant donné un terme t de la forme $\Lambda\mathcal{S}_1 \ldots \Lambda\mathcal{S}_k.\lambda x_1 \ldots \lambda x_{k'}.(x)\mathcal{S}'_1 \ldots \mathcal{S}'_l t_1 \ldots t_{l'}$, *son* **ensemble de positions** $\mathcal{P}(t)$ *est défini comme* $\{(i, j), 1 \leq i \leq l, 1 \leq j \leq \sharp\mathcal{S}'_i\} \cup \{(l + 1, j), 1 \leq j \leq l'\}$ *et le sous-terme de t* **en position** $(i, j) \in \mathcal{P}(t)$ *est le j^{e} élément de $\mathcal{S}'_i$ si $i \leq l$ ou bien t_j si* $i = l + 1$.

Définition 4.5 (*Ensemble de $\Lambda\mu$-positions*)

On dit qu'une partie $\mathcal{P}$ finie de $\mathbb{N}^{\star 2}$ est un ensemble de $\Lambda\mu$-**positions** *si les éléments de $\mathcal{P}$ sont tels que si $(i, j + 1) \in \mathcal{P}$ avec $j \geq 1$, alors* $(i, j) \in \mathcal{P}$.

Définition 4.6 (*Sélecteur de sous-terme, ou projection $\Phi^{\mathcal{P}}_{(i,j)}$*)

Le Sélecteur de sous-terme, ou projection, $\Phi^{\mathcal{P}}_{(i,j)}$ qui doit renvoyer le sous-terme en position (i, j) d'un terme $t = (x)\mathcal{S}_1 \ldots \mathcal{S}_s t_1 \ldots t_m$ est

$$\Phi^{\mathcal{P}}_{(i,j)} = \mu\alpha_1 \ldots \mu\alpha_{i-1}.\lambda x_i^1 \ldots \lambda x_i^j.\mu\alpha_i \ldots \mu\alpha_s.\lambda x_{s+1}^1 \ldots \lambda x_{s+1}^m.x_i^j$$

où (i, j) est dans l'ensemble de positions $\mathcal{P}$ du terme t. (Bien sûr, si $i = s + 1$, alors $\Phi^{\mathcal{P}}_{(i,j)} = \mu\alpha_1 \ldots \mu\alpha_s.\lambda x_{s+1}^1 \ldots \lambda x_{s+1}^m.x_{s+1}^j$).

4.2.2 Contextes Applicatifs Minimaux

Définition 4.7 (*Forme Normale Canonique Applicative/Fonctionnelle*)

Soit t un $\Lambda\mu$-terme en forme normale canonique. t est de la forme :

$$\Lambda\mathcal{S}_1 \ldots \Lambda\mathcal{S}_k.\lambda x_1 \ldots \lambda x_l.(x)\mathcal{S}'_1 \ldots \mathcal{S}'_{k'} t_1 \ldots t_{l'}.$$

t est dit en **forme normale canonique applicative (FNCA)** *si* $(k, l) = (0, 0)$ *et en* **forme normale canonique fonctionnelle** *sinon.*

Proposition 4.8 (***Mise en forme normale canonique applicative***)

- *Soit t un $\Lambda\mu$-terme en forme normale canonique. Il existe un contexte applicatif $\mathcal{C}_t$ tel que $\mathcal{C}_t(t)$ se réduit en forme normale canonique applicative ;*
- *Soient t et u deux $\Lambda\mu$-termes en forme normale canonique. Il existe un contexte applicatif $\mathcal{C}_{t,u}$ tel que $\mathcal{C}_{t,u}(t)$ et $\mathcal{C}_{t,u}(u)$ se réduisent en formes normales canoniques applicatives ;*
- *plus généralement, si $t_1, \dots t_k$ sont des $\Lambda\mu$-termes en forme normale canonique, il existe un contexte $\mathcal{C}_{(t_i)_{1\leq i\leq k}}$ tel que pour tout $1 \leq j \leq k$, $\mathcal{C}_{(t_i)_{1\leq i\leq k}}(t_j)$ se réduit en une forme normale canonique applicative.*

Démonstration : Le troisième résultat, qui généralise les deux premiers, se montre par induction sur le nombre maximal de streams abstraites en tête de l'un des t_i puis sur la longueur maximale du premier préfixe de λ en tête des t_i.

□

Remarque 4.9

On notera que cette mise en forme applicative peut se faire grâce à la réduction gauche.

Dans chacun des cas de la proposition précédente, il existe bien sûr un contexte applicatif de longueur minimale (voir la définition 4.3). Par ailleurs, ces contextes de longueur minimale ont la même structure de stream (et les streams apparaissant à une même position de deux contextes de longueur minimale ont elles-mêmes même longueur). Il existe un contexte applicatif de longueur minimale ***«plus général»*** que les autres dans le sens où tous les autres contextes en sont des instances par substitution de terme ou de variable de stream.

On appelle ce contexte le plus général le contexte applicatif minimal associé à t et u (et on ne le définit que pour deux termes, mais sa généralisation ne présente pas de difficulté) :

Définition 4.10 (***Contexte Applicatif Minimal ; CAM***)

Étant données deux $\Lambda\mu$-formes normales canoniques t et u, un **Contexte Applicatif Minimal** *(CAM) est un contexte $\mathcal{C} = []x_{11}\dots x_{1k_1}\ \alpha_1\dots\alpha_n x_{(n+1)1}\dots x_{(n+1)k_{n+1}}$ de longueur minimale tel que $\mathcal{C}[t]$ et $\mathcal{C}[u]$ se réduisent vers des formes normales canoniques applicatives, t' et u', et tel que les variables de $\mathcal{C}$ soient deux à deux distinctes et qu'elles n'interfèrent pas avec les variables libres de t et u, c'est-à-dire que les variables libres de $\mathcal{C}$ sont différentes de toutes les variables libres de t et u.*

t' et u' sont appelés les **formes réduites** *de $\mathcal{C}[t]$ et $\mathcal{C}[u]$.*

Remarque 4.11

Plus précisément, si t, u sont de la forme $t = \Lambda\mathcal{S}_1\dots\Lambda\mathcal{S}_p.\lambda x_1\dots\lambda x_{p'}.(x)\mathcal{S}'_1\dots\mathcal{S}'_q t_1\dots t_{q'}$ et $u = \Lambda\mathcal{P}_1\dots\Lambda\mathcal{P}_r.\lambda y_1\dots\lambda y_{r'}.(y)\mathcal{P}'_1\dots\mathcal{P}'_s u_1\dots u_{s'}$, leur **contexte applicatif minimal** *est un contexte applicatif de la forme : $[]x_{11}\dots x_{1k_1}\alpha_1\dots\alpha_n\ x_{(n+1)1}\dots x_{(n+1)k_{n+1}}$ tel que $n = max(p, r)$ vérifiant les conditions suivantes :*

- *$x_{ij} \neq x_{kl}$ dès que $(i, j) \neq (k, l)$;*
- *$\alpha_i \neq \alpha_j$ dès que $i \neq j$;*
- *$x_{ij} \notin FV_t((t)u)$ et $\alpha_i \notin FV_s((t)u)$;*
- *si $p = r = n$, on a pour tout $i \leq n$, $k_i = max(\sharp\mathcal{S}_i, \sharp\mathcal{P}_i)$ et $k_{n+1} = max(p', r')$;*
- *si $p > r$, on a pour tout $i \leq r$, $k_i = max(\sharp\mathcal{S}_i, \sharp\mathcal{P}_i)$, $k_{r+1} = max(\sharp\mathcal{S}_{r+1}, r')$ et pour tout $r + 1 < i \leq n$, $k_i = \sharp\mathcal{S}_i$ et $k_{n+1} = p'$. Le cas où $p < r$ est symétrique.*

Pour assurer que l'induction peut être appliquée dans la preuve du théorème principal, il nous faut vérifier que la mise en FNCA ne fait pas croître la taille des termes considérés.

Proposition 4.12 (*Décroissance de la taille par mise en FNCA*)

Considérons deux $\Lambda\mu$-termes t et u en forme normale canonique, $\mathcal{C}$ un CAM pour cette paire de termes et t' et u' les formes réduites de $\mathcal{C}[t]$ et $\mathcal{C}[u]$ respectivement.

Alors on a $\mathcal{T}(t') + \mathcal{T}(u') \leq \mathcal{T}(t) + \mathcal{T}(u)$.

Démonstration : On donne simplement une idée de la preuve qui ne présente pas de difficulté, en reprenant les notations de la remarque 4.11.

La preuve se fait par induction sur le nombre de streams composant le contexte applicatif minimal et la longueur de ce contexte, en distinguant le cas où $(p, p') = (r, r')$ du cas où $(p, p') \neq (r, r')$. Selon le cas, la taille décroît strictement ou bien elle reste constante, voir l'exemple qui suit à ce sujet.

□

Remarque 4.13

On remarquera que cette inégalité n'est pas stricte en général, même si le contexte applicatif minimal n'est pas trivial.

Par exemple, les termes $t = \mu\alpha.\lambda x.y$ et $u = z$ admettent comme CAM le contexte $\mathcal{C} = []\alpha x$ qui conduit aux termes $t' = y$ et $u' = (z)\alpha x$ et la somme des tailles n'a pas diminué.

Proposition 4.14

Soient t et u deux formes normales canoniques et soit $\mathcal{C}$ un contexte applicatif minimal pour t, u. Soient t' et u' les formes réduites de $\mathcal{C}[t]$ et $\mathcal{C}[u]$. On a :
- $FV((t)u) \cap FV(\mathcal{C}) = \emptyset$;
- $FV((t')u') = FV((t)u) \cup FV(\mathcal{C})$.

Démonstration : La vérification est immédiate.

□

4.3. Lemmes péliminaires

4.3.1 Paires paramétriques

Dans la partie clé de la preuve, nous devrons remplacer un terme par un autre terme qui transporte plus d'information : nous devons donc stocker deux programmes faisant respectivement les tâches décrites en sous-section 4.1. La manière habituelle de faire cela en λ-calcul est d'utiliser des paires mais nous avons besoin que l'information ne soit pas récupérée immédiatement ; nous avons besoin que la «paire» accepte de stocker des arguments avant de choisir quelle composante elle devra retourner. Dans ce but, une structure de paires paramétriques est nécessaire (il s'agit simplement d'une extension de l'encodage habituel des paires en λ-calcul) :

Définition 4.15 (*Paires Paramétriques :* $\langle \_ , \_ \rangle_k$)

Soit $k \in \mathbb{N}$ et u, v des $\Lambda\mu$-termes. On définit $\langle u, v\rangle_k$ par induction sur k comme suit :
- $\langle u, v\rangle_0 = \lambda z.(z)uv \qquad z \notin FV_t((u)v)$
- $\langle u, v\rangle_{k+1} = \mu\alpha.\langle (u)\alpha, (v)\alpha\rangle_k \qquad \alpha \notin FV_s((u)v)$

Cela peut être reformulé en : $\langle u, v\rangle_k = \mu\alpha_1 \dots \mu\alpha_k.\lambda z.((z)(u)\alpha_1 \dots \alpha_k)(v)\alpha_1 \dots \alpha_k$

Remarque 4.16

Il serait également naturel de définir des paires à double indice : $\langle \_, \_\rangle_{(k,l)}$ *par :*

- $\langle u,v\rangle_{(0,0)} = \langle u,v\rangle_0$;
- $\langle u,v\rangle_{(k+1,l)} = \mu\alpha.\langle (u)\alpha,(v)\alpha\rangle_{(k,l)} \qquad \alpha \notin FV_s((u)v)$
- $\langle u,v\rangle_{(0,l+1)} = \lambda x.\langle (u)x,(v)x\rangle_{(0,l)} \qquad x \notin FV_t((u)v)$
- *ou, de manière plus synthétique :*

$\langle u,v\rangle_{(k,l)} = \mu\alpha_1\dots\mu\alpha_k.\lambda x_1\dots\lambda x_l.\lambda z.((z)(u)\alpha_1\dots\alpha_k x_1\dots x_l)(v)\alpha_1\dots\alpha_k x_1\dots x_l.$

Pourtant, ces paires ne nous seront pas utiles car les paires de la définition 4.15 sont suffisantes.

Proposition 4.17

Les relations suivantes sont vérifiées :

- $\langle u,v\rangle_0 1 \longrightarrow_g^{\star} u$;
- $\langle u,v\rangle_0 0 \longrightarrow_g^{\star} v$;
- $\langle u,v\rangle_{k+1} t \longrightarrow_g^{\star} \langle (u)t,(v)t\rangle_{k+1}$;
- $\langle u,v\rangle_{k+1} \alpha \longrightarrow_g^{\star} \langle (u)\alpha,(v)\alpha\rangle_{k}$;
- *Plus généralement, en utilisant la notation de Stream, on obtient :*
  $\langle u,v\rangle_k \mathcal{S}_1\dots\mathcal{S}_k 1 \longrightarrow_g^{\star} (u)\mathcal{S}_1\dots\mathcal{S}_k \qquad$ *et* $\qquad \langle u,v\rangle_k \mathcal{S}_1\dots\mathcal{S}_k 0 \longrightarrow_g^{\star} (v)\mathcal{S}_1\dots\mathcal{S}_k$

Notation 4.18 (*La Stream* $\mathbb{1}^k$)

On notera $\mathbb{1}^k$ *pour la stream* $[]1\dots1\alpha$ *constituée de* k *occurrences du terme* 1*, que l'on notera encore* $(1^k,\alpha)$ *lorsqu'on veut être précis quant à la variable de stream utilisée.*

La stream $\mathbb{1}^k$ sera constamment utilisée pendant le Böhm Out. On démontre maintenant le lemme des paires paramétriques :

Lemme 4.19 (*Lemme des Paires Paramétriques*)

Soient τ,u *des* $\Lambda\mu$*-termes et* $x,y \in \mathcal{V}_t$ *avec* $x \notin FV_t(u)$.
Si $\tau\{u/x\} \longrightarrow_g^{\star} y$*, alors nous avons :*
$\exists K \in \mathbb{N}$ *tel que* $\forall k \geq K, \forall v \in \Sigma_{\Lambda\mu}$,
$\exists n_0$ *tel que* $\forall (n_1,\dots,n_{k+1})$ *tous supérieurs ou égaux à* n_0,
$\exists (l_1,\dots,l_{k+1}) \in \mathbb{N}^{k+1}$ *tels que :*

$$\tau\{\langle u,v\rangle_k/x\}\,\mathbb{1}^{n_1}\dots\mathbb{1}^{n_{k+1}} \longrightarrow_g^{\star} (y)\mathbb{1}^{l_1}\dots\mathbb{1}^{l_{k+1}}.$$

Le lemme assure que si $\tau\{u/x\}$ se réduit sur une variable y, alors pour tout entier k assez grand et pour tout terme v, la paire paramétrique $\langle u,v\rangle_k$ se comportera de la même manière que u dans le terme τ pourvu qu'on lui passe un nombre suffisamment grand de streams $\mathbb{1}^n$ suffisamment grandes. L'idée intuitive est que l'on peut remplacer le terme u par un terme transportant plus d'information mais qui se comporte tout de même comme u dans le contexte τ.

Démonstration : On démontre le lemme par induction sur la longueur de la réduction gauche $\tau\{u/x\} \longrightarrow_g^{\star} y$ et par cas sur τ (c'est-à-dire par induction sur les paires (longueur de la réduction gauche, numéro du cas sur τ) ordonnées pour l'ordre lexicographique).

Premier cas : τ n'est pas en forme normale de tête. Dans ce cas, nous avons $\tau \longrightarrow_g \tau'$ pour un certain $\tau' \in \Sigma_{\Lambda\mu}$, et en conséquence $\tau\{u/x\} \longrightarrow_g$

$\tau' \{u/x\} \longrightarrow_g^\star y$ puisque τ n'est pas en forme normale de tête. Par hypothèse d'induction sur τ', nous obtenons le résultat.

Second cas : τ est en forme normale de tête. Dans ce cas, τ est de la forme $\tau = (z)\mathcal{S}_1 \dots \mathcal{S}_p t_1 \dots t_q$ et nous avons trois sous-cas : soit $z = y$, soit $z = x$ et $\tau = x$, soit $z = x$ et $(p, q) \neq (0, 0)$.

1. *Si $z = y$*, le résultat est immédiatement vrai puisque $\tau = y$.
2. *Si $z = x$ et $\tau = x$*, alors on a $u \longrightarrow_g^\star y$ et pour tout $(n_1, \dots, n_{k+1})$ tous plus grands que 1, on a :

$$\begin{aligned}\langle u, v\rangle_k \mathbb{1}^{n_1} \dots \mathbb{1}^{n_{k+1}} &\longrightarrow_g^\star \langle (u)\mathbb{1}^{n_1} \dots \mathbb{1}^{n_k}, (v)\mathbb{1}^{n_1} \dots \mathbb{1}^{n_k}\rangle_0 1 \mathbb{1}^{n_{k+1}-1}\\ &\longrightarrow_g^\star (u)\mathbb{1}^{n_1} \dots \mathbb{1}^{n_{k+1}-1}\\ &\longrightarrow_g^\star (y)\mathbb{1}^{n_1} \dots \mathbb{1}^{n_{k+1}-1}\end{aligned}$$

3. *Si $z = x$ et $\tau = (x)\mathcal{S}_1 \dots \mathcal{S}_p t_1 \dots t_q$ avec $(p, q) \neq (0, 0)$*, on a alors bien sûr $\tau \{u/x\} \longrightarrow_g^+ y$.

   Dans la suite, on abrège $t \{u/x\}$ par t' et on étend cette notation de manière évidente aux applications de stream de telle sorte que $\tau \{u/x\}$ sera noté $(u)\mathcal{S}'_1 \dots \mathcal{S}'_p t'_1 \dots t'_q$.

   On remarque que $\tau \{u/x\}$ ne peut pas être en forme normale de tête (puisque le terme doit se réduire vers y en au moins une étape), et on note alors τ' le $\Lambda\mu$-terme $(u)\mathcal{S}_1 \dots \mathcal{S}_p t_1 \dots t_q$ (seul le x en position de tête a été substitué). Puisque x ne fait pas partie des variables libres de u, on a l'égalité suivante : $\tau' \{u/x\} = \tau \{u/x\}$.

   Les longueurs des réductions associées à τ et τ' sont les mêmes (les réductions sont les mêmes...) mais contrairement à τ, τ' n'est pas en forme normale de tête de telle sorte que τ' correspond au premier cas et qu'il est possible d'appliquer l'hypothèse d'induction à τ'.

   Considérons donc un entier K satisfaisant la condition pour τ' et définissons K' comme $max(K, p+1)$. On prend $k \geq K'$ et $v \in \Sigma_{\Lambda\mu}$. Soit n_0 tel que requis par l'énoncé du lemme appliqué à τ' et on définit n'_0 comme $n_0 + 1$. Considérons maintenant $(n_1, \dots, n_{k+1})$ tous plus grands que n'_0. Nous avons une réduction de $\tau[\langle u, v\rangle_k/x]\mathbb{1}^{n_1} \dots \mathbb{1}^{n_{k+1}}$ vers $(y)\mathbb{1}^{l_1} \dots \mathbb{1}^{l_{k+1}}$ pour des entiers $l_1, \dots, l_{k+1}$.

   On montre alors en figure 4.1 une dérivation de $\tau \{\langle u, v\rangle_k/x\} \mathbb{1}^{n_1} \dots \mathbb{1}^{n_{k+1}}$ vers $(y)\mathbb{1}^{l_1} \dots \mathbb{1}^{l_{k+1}}$.

   La composante de droite de la paire n'est pas explicitée puisqu'elle n'affecte pas la dérivation. La dernière étape est justifiée par l'hypothèse d'induction appliquée à τ' et le fait que k soit plus grand que K et que les $n_1, \dots, n_{k-p+1} - 1, \dots, n_{k+1}$ soient tous plus grands que n_0. Les termes notés t'' ou $\mathcal{S}''$ dénotent le fait que la substitution est maintenant $\{\langle u, v\rangle_k/x\}$ au lieu de $\{u/x\}$.

   Ceci conclut la preuve du lemme.

□

4.3.2 Lemme des sous-termes

Définition 4.20 (*Condition de Sous-terme*)

Deux $\Lambda\mu$-termes t et u satisfont la condition de sous-terme si, appliqués à un CAM, ils se réduisent vers $t' = (x)\mathcal{S}_1 \dots \mathcal{S}_s t_1 \dots t_m$ et $u' = (y)\mathcal{P}_1 \dots \mathcal{P}_p u_1 \dots u_n$ qui sont tels que :

- *$x \neq y$ ou $s \neq p$ ou $m \neq n$;*
- *ou bien il existe une paire $(i, j) \in \mathcal{P}(t') \cap \mathcal{P}(u')$ telle que les sous-termes de t' et u' en position (i, j) satisfont la condition de sous-terme.*

$$
\begin{array}{ll}
& \tau\,\{\langle u,v\rangle_k/x\}\,\mathbb{1}^{n_1}\dots\mathbb{1}^{n_{k+1}} \\
= & (\langle u,v\rangle_k)\mathcal{S}''_1\dots\mathcal{S}''_p t''_1\dots t''_q\mathbb{1}^{n_1}\dots\mathbb{1}^{n_{k+1}} \\
\longrightarrow^{\star}_{g} & (\langle (u)\mathcal{S}''_1\dots\mathcal{S}''_p t''_1\dots t''_q,\star\rangle_{k-p})\mathbb{1}^{n_1}\dots\mathbb{1}^{n_{k+1}} \\
\longrightarrow^{\star}_{g} & (\langle (u)\mathcal{S}''_1\dots\mathcal{S}''_p t''_1\dots t''_q\mathbb{1}^{n_1}\dots\mathbb{1}^{n_{k-p}},\star\rangle_0) \\
& 1\mathbb{1}^{n_{k-p+1}-1}\dots\mathbb{1}^{n_{k+1}} \\
\longrightarrow^{\star}_{g} & (u)\mathcal{S}''_1\dots\mathcal{S}''_p t''_1\dots t''_q\mathbb{1}^{n_1}\dots\mathbb{1}^{n_{k-p+1}-1}\dots\mathbb{1}^{n_{k+1}} \\
= & \tau'\,\{\langle u,v\rangle_k/x\}\,\mathbb{1}^{n_1}\dots\mathbb{1}^{n_{k-p+1}-1}\dots\mathbb{1}^{n_{k+1}} \\
\longrightarrow^{\star}_{g} & (y)\mathbb{1}^{l_1}\dots\mathbb{1}^{l_{k+1}}
\end{array}
$$

FIG. 4.1 – Dérivation du lemme des paires paramétriques

On remarquera que la condition de sous-terme implique que les termes ne sont pas équivalents, le contraire n'étant pas vérifié : de manière générale, des termes non-équivalents en CNF ne vérifient pas la condition de sous-terme comme le montrent les termes $(x)y\alpha$ et $(x)\alpha$.

La *fst*-transformation traite ce cas. Il s'agit d'appliquer une étape de la règle *fst* à chaque μ-abstraction du terme considéré. On commence par annoter les μ-abstractions en travaillant dans un langage élargi dont la syntaxe est définie ci-dessous :

Définition 4.21 ($\lambda\overline{\mu}$-*calcul*)

Les $\lambda\overline{\mu}$-termes sont définis par la syntaxe suivante :

$$s,t,u,v ::= \; x \mid \lambda x.t \mid \mu\alpha.t \mid \overline{\mu}\alpha.t \mid (t)\alpha \mid (t)u$$

Définition 4.22 ($\overline{fst}$-*réduction*)

La $\overline{fst}$-réduction est la réduction définie par la règle :

$$\overline{\mu}\alpha.t \longrightarrow_{\overline{fst}} \lambda x.\mu\alpha.t\,\{(u)x\alpha/(u)\alpha\} \qquad x \notin FV_t(t)$$

Il est immédiat que la $\overline{fst}$-réduction termine et est confluente.

Définition 4.23 ($\overline{\mu}$-*transformation*)

Étant donné un $\Lambda\mu$-terme t, la $\overline{\mu}$-transformation appliquée à t produit le terme $\overline{t}$ défini inductivement comme suit :
- $\overline{x} = x$
- $\overline{\lambda x.t} = \lambda x.\overline{t}$
- $\overline{(t)u} = (\overline{t})\overline{u}$
- $\overline{(t)\alpha} = (\overline{t})\alpha$
- $\overline{\mu\alpha.t} = \overline{\mu}\alpha.\overline{t}$

On peut finalement définir la *fst*-transformation d'un $\Lambda\mu$-terme t :

Définition 4.24 (*fst-transformation*)

Étant donné un $\Lambda\mu$-terme t, on définit $\lceil t\rceil$ comme la $\overline{fst}$-forme normale de $\overline{t}$.

La *fst*-transformation d'un $\Lambda\mu$-terme, telle que définie dans la définition 4.24, revient précisément à avoir appliqué une *fst*-réduction à chaque μ-abstraction du $\Lambda\mu$-terme. Par ailleurs, on remarque que $\mathcal{T}(t) = \mathcal{T}(\lceil t\rceil)$.

Lemme 4.25 (***Lemme du sous-terme***)

> *Étant donnés deux $\Lambda\mu$-termes clos t et u en forme normale canonique qui ne sont pas $\Lambda\mu$-équivalents, leurs fst-transformés vérifient la condition de sous-terme.*

Pour démontrer le lemme du sous-terme, on va procéder par induction mais il nous faudra renforcer l'hypothèse d'induction de manière à pouvoir l'appliquer à des sous-termes.

On commence par montrer le lemme suivant :

Lemme 4.26

> *On considère la condition suivante sur des paires de termes t et u, que l'on appellera $(*)$. t et u sont tels que :*
>
> *(i) ils sont en forme normale canonique ;*
>
> *(ii) ils ne sont pas $\Lambda\mu$-équivalents ;*
>
> *(iii) toute occurrence d'une variable de stream α, liée dans t (resp. u), apparaît dans un sous-terme de t (resp. u) de la forme $\lambda v_\alpha.\mu\alpha.\mathcal{C}[(t')t'_1 \dots t'_k v_\alpha \alpha]$ où les occurences de α et v_α sont libres dans $\mathcal{C}[(t')t'_1 \dots t'_k v_\alpha \alpha]$; toutes les occurrences de v_α sont de cette forme.*
>
> *(iv) toute occurrence d'une variable de stream α, libre dans t (resp. u), peut se voir associer une variable de terme v_α, libre dans t (resp. u), telle que les occurrences libres de α et v_α libres dans t (resp. u), sont toujours de la forme $(t')v_\alpha\alpha$.*
>
> *Si t et u satisfont $(*)$, alors les formes réduites t' et u' obtenues en appliquant un CAM $\mathcal{C}_{t,u}$ à t et u vérifient encore $(*)$.*

Démonstration : Soient $t = \Lambda\mathcal{S}_1 \dots \Lambda\mathcal{S}_p.\lambda x_1 \dots \lambda x_{p'}.(x)\mathcal{S}'_1 \dots \mathcal{S}'_q t_1 \dots t_{q'}$ et $u = \Lambda\mathcal{P}_1 \dots \Lambda\mathcal{P}_r.\lambda y_1 \dots \lambda y_{r'}.(y)\mathcal{P}'_1 \dots \mathcal{P}'_s u_1 \dots u_{s'}$. Soit $\mathcal{C}_{t,u}$ un CAM pour ces deux termes et soient t' et u' les formes réduites associées.

- Par définition des formes réduites, t' et u' satisfont la condition (i) de $(*)$.
- La condition (ii) est évidemment vérifiée car $t' =_{\Lambda\mu} u'$ impliquerait $t =_{\Lambda\mu} u$.
- Si α est une variable de stream liée de t' (resp. u'), elle était également liée dans t (resp. u) et donc vérifiait la condition (iii) pour t (resp. u).
- Enfin, une variable de stream libre α de t' peut être de deux types (par le lemme 4.14) :
  - *soit α était libre dans t* auquel cas la condition (iv) était vérifiée dans t et l'application du CAM ne perturbe rien puisque l'ensemble des variables libres de t et u et celui du CAM sont disjoints ;
  - *soit α est une variable libre du CAM* qui termine, par exemple, la i^e stream. Par minimalité de $\mathcal{C}_{t,u}$ la longueur de cette stream doit être égale soit à la longueur de $\Lambda\mathcal{S}_i$, soit à la longueur de $\Lambda\mathcal{P}_i$ (au moins l'une des deux existe, supposons alors qu'il s'agit de $\Lambda\mathcal{S}_i$), c'est-à-dire que, α va être substituée là où se trouvent les occurences de la variable de stream *liée* de t ou u qui termine $\Lambda\mathcal{S}_i$ et que l'on notera par la suite β. Puisque t vérifie la condition $(*)$, et comme β est liée dans t, toutes les occurrences liées de β dans t sont de la forme $\lambda v_\beta.\mu\beta.\mathcal{C}[(t')t'_1 \dots t'_k v_\beta \beta]$ et λv_β reçoit comme argument la variable de terme x_α qui précédait α dans le CAM, ce qui assure que (iv) est encore vérifiée pour la variable libre α de t' (resp. u').

□

On démontre maintenant le lemme du sous-terme :

Démonstration du lemme 4.25 : On montre le lemme par induction sur la somme des tailles des termes concernés :

- si la somme des tailles de t et u vaut 0, on est en présence de deux variables et comme les termes ne sont pas équivalents, les variables sont distinctes et la condition de sous-terme est satisfaite.

– supposons le résultat vrai pour tous les termes dont la somme des tailles est inférieure à un entier n et considérons t et u satisfaisant la condition $(*)$ dont la somme des tailles vaut $n+1$.
Soit $\mathcal{C}$ un CAM pour t et u et soient t' et u' les formes réduites de $\mathcal{C}[t]$ et $\mathcal{C}[u]$. Notons $t' = (x)\mathcal{S}_1 \dots \mathcal{S}_s t_1 \dots t_m$ et $u' = (y)\mathcal{P}_1 \dots \mathcal{P}_p u_1 \dots u_n$. Si $x \neq y$, $s \neq p$ ou $m \neq n$, on a le résultat par définition de la condition de sous-terme. Supposons donc que $x = y$, $s = p$ et $m = n$. Si il existe $i \leq s$ tel que la variable de stream $\mathcal{S}_i$ diffère de la variable de stream de $\mathcal{P}_i$ alors puisque t' et u' vérifient $(*)$, les termes qui précèdent ces variables libres fournissent un sous-terme satisfaisant la condition puisqu'il s'agit de variables distinctes. Si en revanche toutes les variables de stream se correspondent deux à deux, alors de deux choses l'une : soit il existe $i \leq s$ tel que $\mathcal{S}_i$ et $\mathcal{P}_i$ ont des longueurs distinctes $k < l$, auquel cas le sous-terme en position k satisfait la condition (dans la stream de longueur k, il s'agit de la variable associée à la variable de stream qui ne peut apparaître ailleurs qu'en fin de stream et donc pas en position k dans l'autre stream), soit toutes les streams ont même longueur, auquel cas il existe une position (i, j) telle que les termes t_{ij} et u_{ij} en ces positions satisfont la condition $(*)$ (ils sont en forme normale canonique, ne sont pas $\Lambda\mu$-équivalents et les conditions (iii) et (iv) sont satisfaites puisqu'il s'agit de sous-termes de t et u) Dans tous les cas, les termes considérés satisfont $(*)$ et sont de taille strictement inférieure à celles de t et u ce qui permet d'appliquer l'hypothèse de récurrence : on a trouvé des sous-termes satisfaisant la condition de sous-termes et donc t et u satisfont également la condition de sous-terme par la même occasion.
Ceci termine la preuve du lemme.

□

4.4. Preuve du résultat principal

On dispose maintenant des outils pour démontrer le résultat principal :

Théorème 4.27 (***Théorème de Séparation pour $\Lambda\mu$***)

Soient t et u deux $\Lambda\mu$-termes clos en forme normale canonique qui ne sont pas $\Lambda\mu$-équivalents. Il existe un contexte applicatif $\mathcal{C} = []\mathcal{S}_1 \dots \mathcal{S}_k w_1 \dots w_l$ tel que $\mathcal{C}[t] \longrightarrow^\star_{\Lambda\mu} 1$ tandis que $\mathcal{C}[u] \longrightarrow^\star_{\Lambda\mu} 0$.

Démonstration : Nous allons en fait séparer les *fst*-transformés $\lceil t \rceil$ et $\lceil u \rceil$ de t et u respectivement, ce qui sera plus simple puisque par le lemme 4.25 ils vérifient le lemme du sous-terme. Il est facile de vérifier qu'un contexte séparant pour ces termes est également un contexte séparant pour t et u.

Par ailleurs, nous allons montrer que la séparation peut se faire en utilisant simplement la réduction gauche et enfin, nous allons raisonner par induction sur la taille des termes et par cas sur leur structure (dans le cas où les termes peuvent posséder des variables libres) en surchargeant l'induction de manière à imposer trois hypothèses sur les termes t et u auxquels s'appliquent l'induction : *(i)* qu'ils soient en forme normale canonique, *(ii)* qu'ils ne soient pas $\Lambda\mu$-équivalents et enfin *(iii)* qu'ils satisfassent la condition de sous-terme (voir définition 4.20).

Notre hypothèse d'induction sera donc :

HR(n)

Pour toute paire de termes t et u si :

(i) la somme des tailles de t et u est inférieure ou égale à n ;

(ii) $\mathcal{V} = FV_t((t)u)$;

(iii) t et u sont en forme normale canonique ;

(iv) $t \neq_{\Lambda\mu} u$;

(v) t et u satisfont la condition de sous-terme ;

alors il existe une substitution $\sigma = \left\{\overrightarrow{t}/\overrightarrow{x}, x \in \mathcal{V}\right\}$ et un contexte $\mathcal{C} = []\overrightarrow{\mathcal{S}}\,\overrightarrow{w}$ t.q. :

- $\mathcal{C}[t^\sigma] \longrightarrow_g^\star 1$ et
- $\mathcal{C}[u^\sigma] \longrightarrow_g^\star 0$.

Premier cas : Les termes t et u sont de la forme $(x)\mathcal{S}_1 \ldots \mathcal{S}_p w_1 \ldots w_q$.
Soient $t = (x)\mathcal{T}_1 \ldots \mathcal{T}_m t_1 \ldots t_{m'}$ et $u = (y)\mathcal{U}_1 \ldots \mathcal{U}_n u_1 \ldots u_{n'}$.

On raisonne par cas selon les valeurs de (x, m, m') et (y, n, n') :
Si $x \neq y$, alors, en posant $\sigma = \{U_x/x\}\{U_y/y\}$ avec $U_x = \mu\alpha_1 \ldots \mu\alpha_m.\lambda y_1.\ldots\lambda y_{m'}.1$ et $U_y = \mu\alpha_1 \ldots \mu\alpha_n.\lambda y_1.\ldots\lambda y_{n'}.0$, on a $t^\sigma \longrightarrow_g^\star 1$ tandis que $u^\sigma \longrightarrow_g^\star 0$.
Si $x = y = s$ mais $(m, m') \neq (n, n')$, on doit distinguer deux sous-cas :
• $m = n$ et $m' \neq n'$. On considère le cas où $m' \lneq n'$, l'autre cas étant symétrique. En posant $U_s = \mu\alpha_1 \ldots \mu\alpha_m.\lambda z_1.\ldots\lambda z_{n'+1}.z_{n'+1}$ et $\overrightarrow{V}$ le contexte applicatif $\{\lambda z_1.\ldots\lambda z_{n'-m'}.1, 0^{n'-m'}\}$, on a $t\,\{U_s/s\}\,\overrightarrow{V} \longrightarrow_g^\star (\lambda z_1 \ldots \lambda z_{n'-m'}.1)0^{n'-m'} \longrightarrow_g^\star 1$ tandis que $u\,\{U_s/s\}\,\overrightarrow{V} \longrightarrow_g^\star 0$ et on obtient le résultat attendu.
• Si $m \neq n$, par exemple $m \lneq n$. En posant $U_s = \mu\gamma_1 \ldots \mu\gamma_n.\lambda z_1 \ldots \lambda z_{n'+1}.z_{n'+1}$ et $\overrightarrow{V} = \{\mu\gamma_1 \ldots \mu\gamma_{n-m}.\lambda z_1 \ldots \lambda z_{n'+1}.0, \gamma_1, \ldots, \gamma_{n-m}, 1^{n'+1}\}$ on obtient la dérivation suivante :

$$\begin{array}{ll}
 & t\,\{U_s/s\}\,\overrightarrow{V} \\
= & ((U_s)\mathcal{T}_1' \ldots \mathcal{T}_m' t_1' \ldots t_{m'}' \quad \mu\gamma_1 \ldots \mu\gamma_{n-m}.\lambda z_1 \ldots \lambda z_{n'+1}.0)\gamma_1 \ldots \gamma_{n-m} 1^{n'+1} \\
\longrightarrow_g^\star & ((\mu\gamma_{m+1} \ldots \mu\gamma_n.\lambda z_1 \ldots \lambda z_{n'+1}.z_{n'+1})t_1' \ldots t_{m'}' \\
 & \quad \mu\gamma_1 \ldots \mu\gamma_{n-m}.\lambda z_1 \ldots \lambda z_{n'+1}.0)\gamma_1 \ldots \gamma_{n-m} 1^{n'+1} \\
\longrightarrow_g^\star & (\lambda z_1.\ldots\lambda z_{n'+1}.z_{n'+1})1^{n'+1} \\
\longrightarrow_g^\star & 1
\end{array}$$

et de l'autre côté, on a $u\,\{U_s/s\}\,\overrightarrow{V} \longrightarrow_g^\star 0$.
Si $x = y = s$ et $(m, m') = (n, n')$. Il s'agit du seul cas vraiment intéressant. Dans ce cas, puisque la paire (t, u) vérifie la condition de sous-terme, nous avons une paire d'entiers (i, j) telle que $t_{ij} \neq_{\Lambda\mu} u_{ij}$ et telle que t_{ij} et u_{ij} vérifient la condition de sous-terme et sont en forme normale canonique, les tailles des termes t_{ij} et u_{ij} étant strictement inférieures aux tailles de t et u.

Par hypothèse d'induction sur (t_{ij}, u_{ij}), il y a des termes $\overrightarrow{w} = (w_x)_{x \in FV((t_{ij})u_{ij})}$ et $\overrightarrow{V} = \mathcal{V}_1 \ldots \mathcal{V}_v V_1 \ldots V_{v'}$ tels que $t_{ij}\,\{\overrightarrow{w}/\overrightarrow{x}\}\,\overrightarrow{V} \longrightarrow_g^\star 1$ et $u_{ij}\,\{\overrightarrow{w}/\overrightarrow{x}\}\,\overrightarrow{V} \longrightarrow_g^\star 0$ de telle sorte que pour toutes variables z_1, z_2 on a $t_{ij}\,\{\overrightarrow{w}/\overrightarrow{x}\}\,\overrightarrow{V} z_1 z_2 \longrightarrow_g^\star z_1$ et $u_{ij}\,\{\overrightarrow{w}/\overrightarrow{x}\}\,\overrightarrow{V} z_1 z_2 \longrightarrow_g^\star z_2$.

En notant t' et u' les termes $t_{ij}\,\{\overrightarrow{w}/\overrightarrow{x}, x \neq s\}\,\overrightarrow{V} z_1 z_2$ et $u_{ij}\,\{\overrightarrow{w}/\overrightarrow{x}, x \neq s\}\,\overrightarrow{V} z_1 z_2$ et pourvu que $z_1, z_2 \neq s$, on obtient $t'\,\{w_s/s\} \longrightarrow_g^\star z_1$ et $u'\,\{w_s/s\} \longrightarrow_g^\star z_2$.

À ce point, il nous faut utiliser le lemme des paires paramétriques de manière à passer deux termes à la fois à la variable de tête s : d'un côté, on a besoin de sélectionner un terme dont le rôle est de sélectionner les sous-termes en position (i, j) de t and u (qui seront $\Phi_{(i,j)}^{\mathcal{P}(t)}$)[1] et de l'autre côté un terme dont le rôle sera de séparer

[1]Noter que par hypothèse on a $\mathcal{P}(t) = \mathcal{P}(u)$.

$$
\begin{array}{ll}
& t\left\{\overrightarrow{w'}/\overrightarrow{x}\right\}\overrightarrow{V}\psi_1\psi_0\mathbb{1}^{k_1}\dots\mathbb{1}^{k_{k-m-v}}0\mathbb{1}^{k_{k-m-v+1}}\dots\mathbb{1}^{k_Q} \\
\longrightarrow_g^\star & \langle\star,(\Phi)\mathcal{T}_1'\dots\mathcal{T}_m'{t'}_{m+1}^{1}\dots{t'}_{m+1}^{m'}\overrightarrow{V}\psi_1\psi_0\mathbb{1}^{k_1}\dots\mathbb{1}^{k_{k-m-v}}\rangle_0 0\mathbb{1}^{k_{k-m-v+1}}\dots\mathbb{1}^{k_Q} \\
\longrightarrow_g^\star & \Phi\mathcal{T}_1'\dots\mathcal{T}_m'{t'}_{m+1}^{1}\dots{t'}_{m+1}^{m'}\overrightarrow{V}\psi_1\psi_0\mathbb{1}^{k_1}\dots\mathbb{1}^{k_Q} \\
\longrightarrow_g^\star & (t'_{ij})\overrightarrow{V}\psi_1\psi_0\mathbb{1}^{k_1}\dots\mathbb{1}^{k_{k-m-v+1}}\dots\mathbb{1}^{k_Q} \\
\longrightarrow_g^\star & (\psi_1)\mathbb{1}^{l_1}\dots\mathbb{1}^{l_Q} \\
\longrightarrow_g^\star & 1
\end{array}
$$

FIG. 4.2 – Dérivation finale pour le théorème 4.27.

les termes t_{ij} et u_{ij}.

En appliquant le lemme 4.19 (le lemme des paires paramétriques) à t' et à u' avec w_s et s, on trouve un entier K (le maximum des entiers pour t' et u') tel que pour tout $k \geq K$ et pour tout $(k_1, \dots, k_Q)$ suffisamment grands, il existe $(l_1, \dots, l_Q)$ et $(l'_1, \dots, l'_Q)$ tels que :

$$
\begin{array}{lll}
t'\{\langle w_s,\Phi\rangle_k/s\}\mathbb{1}^{k_1}\dots\mathbb{1}^{k_Q} & \longrightarrow_g^\star (z_1)\mathbb{1}^{l_1}\dots\mathbb{1}^{l_Q} & \text{et} \\
u'\{\langle w_s,\Phi\rangle_k/s\}\mathbb{1}^{k_1}\dots\mathbb{1}^{k_Q} & \longrightarrow_g^\star (z_2)\mathbb{1}^{l'_1}\dots\mathbb{1}^{l'_Q} &
\end{array}
$$

Cela peut aussi s'écrire (avec w'_x égal à w_x sauf si $x = s$ auquel cas $w'_s = \langle w_s, \Phi\rangle_k$) :

$$
\begin{array}{lll}
t_{ij}\left\{\overrightarrow{w'}/\overrightarrow{x}\right\}\overrightarrow{V}z_1z_2\mathbb{1}^{k_1}\dots\mathbb{1}^{k_Q} & \longrightarrow_g^\star (z_1)\mathbb{1}^{l_1}\dots\mathbb{1}^{l_Q} & \text{et} \\
u_{ij}\left\{\overrightarrow{w'}/\overrightarrow{x}\right\}\overrightarrow{V}z_1z_2\mathbb{1}^{k_1}\dots\mathbb{1}^{k_Q} & \longrightarrow_g^\star (z_2)\mathbb{1}^{l'_1}\dots\mathbb{1}^{l'_Q} &
\end{array}
$$

On définit maintenant $\psi_1 = \mu\alpha_1 \dots \mu\alpha_Q.1$ et $\psi_0 = \mu\alpha_1 \dots \mu\alpha_Q.0$ et on considère :

$$
\begin{array}{ll}
& t\left\{\overrightarrow{w'}/\overrightarrow{x}\right\}\overrightarrow{V}\psi_1\psi_0\mathbb{1}^{k_1}\dots 0\mathbb{1}^{k_{k-m-v+1}}\dots\mathbb{1}^{k_Q} \\
= & \langle w_s,\Phi\rangle_k\mathcal{T}_1'\dots\mathcal{T}_m't_1'\dots t_{m'}'\overrightarrow{V}\psi_1\psi_0\mathbb{1}^{k_1}\dots 0\mathbb{1}^{k_{k-m-v+1}}\dots\mathbb{1}^{k_Q}
\end{array}
$$

et le terme correspondant pour u.

On peut enfin exhiber la dérivation de $\mathcal{C}(t)$ à 1 dans ce cas, ce que nous faisons en figure 4.2 (la composante de gauche de la paire n'est pas montrée puisqu'elle n'est pas pertinente pour cette partie du calcul). La dérivation de $\mathcal{C}(u)$ vers 0 est similaire.

Second cas : Au moins l'un de t et u n'est pas de la forme $(x)\mathcal{S}_1 \dots \mathcal{S}_p w_1 \dots w_q$.

En plaçant les deux termes dans un contexte applicatif minimal, ils se réduisent sur des termes t' et u' qui sont de la forme souhaitée, qui satisfont la condition du sous-terme et tels que $\mathcal{T}(t) + \mathcal{T}(u) \geq \mathcal{T}(t') + \mathcal{T}(u')$, de telle sorte qu'il est possible d'utiliser l'hypothèse d'induction.

□

4.5. Séparation du contre-exemple de David & Py

Pour illustrer notre résultat, nous exhibons un contexte séparant pour les termes W_0 et W_1 qui étaient des témoins de l'échec de la séparation dans l'article de David & Py. Nous montrerons comment il est possible de faire se réduire le terme $W = \lambda x.\mu\alpha.((x)\mu\gamma.(x)U_0 y\alpha)U_0\alpha$ vers la variable y dans le contexte $\mathcal{C}$, de sorte que W_0 se réduirait sur 0 et W_1 sur 1 dans ce contexte.

On note $\mathcal{P}$ la paire $\langle \lambda z_0, z_1.\mu\alpha.\mu\beta.z_1, \lambda x.\mu\alpha.x\rangle_1$, le contexte séparant est

$$\mathcal{C} = ([])\mathcal{P}x_0x_1\alpha 0\alpha 1\alpha$$

$$
\begin{array}{rcl}
\boxed{\mathcal{C}[W]} & = & (W)\mathcal{P}x_0x_1\alpha 0\alpha 1\alpha \\
& = & (\lambda x.\mu\alpha.((x)\mu\gamma.(x)U_0 y\alpha)U_0\alpha)\mathcal{P}x_0x_1\alpha 0\alpha 1\alpha \\
& \longrightarrow_g^\star & ((\mathcal{P})\ \mu\gamma.(\mathcal{P})\ U_{x_1}yx_0x_1\alpha)\ U_{x_1}x_0x_1\alpha 0\alpha 1\alpha \\
& \longrightarrow_g^\star & \langle\star, ((\lambda x.\mu\alpha.x)\ \mu\gamma.(\mathcal{P})\ U_{x_1}yx_0x_1\alpha)\ U_{x_1}x_0x_1\alpha\rangle_0\ 0\alpha 1\alpha \\
& \longrightarrow_g^\star & ((\lambda x.\mu\alpha.x)\ \mu\gamma.(\mathcal{P})\ U_{x_1}yx_0x_1\alpha)\ U_{x_1}x_0x_1\alpha\alpha 1\alpha \\
& \longrightarrow_g^\star & (\mu\gamma.(\mathcal{P})\ U_{x_1}yx_0x_1\alpha)\ \alpha 1\alpha \\
& \longrightarrow_g^\star & (\mathcal{P})\ U_{x_1}yx_0x_1\alpha 1\alpha \\
& \longrightarrow_g^\star & \langle(\lambda z_0, z_1.\mu\alpha.\mu\beta.z_1)\ U_{x_1}yx_0x_1\alpha, \star\rangle_0\ 1\alpha \\
& \longrightarrow_g^\star & (\lambda z_0, z_1.\mu\alpha.\mu\beta.z_1)\ U_{x_1}yx_0x_1\alpha\alpha \\
& \longrightarrow_g^\star & \boxed{y}
\end{array}
$$

FIG. 4.3 – Séparation du contre-exemple de David et Py

(ce contexte est une version simplifiée du contexte qui peut être extrait de la preuve mais ayant la même forme ; k_0 pourrait être fixé à 0).

Le processus de séparation est décrit dans la dérivation de la figure 4.3.

4.6. Remarques conclusives

Interprétation de streams. Notre intuition opérationnelle, tout au long de la preuve de séparation, s'est appuyée sur l'interprétation de streams, en particulier sur le fait que les streams peuvent stocker des termes en nombre arbitrairement grand.

La notation de stream a par ailleurs grandement allégé les notations de ce chapitre, permettant de conserver la concision sans sacrifier à l'ambiguïté. Pour autant, il ne s'agit que d'une notation informelle et les streams n'ont pas, en $\Lambda\mu$-calcul, de statut formel. En particulier, le passage d'argument par une abstraction de stream, $\mu\alpha$, se fait élément par élément et non pas en un bloc d'une «stream entière». On pourrait au contraire, souhaiter développer un calcul de stream disposant d'une véritable construction de stream dans laquelle on pourrait considérer des réductions du type :

$$(\Lambda\alpha.t)\mathcal{S} \longrightarrow t\,\{\mathcal{S}/\alpha\}$$

Les chapitres suivants illustreront ce point de vue :

- Le système de typage pour $\Lambda\mu$ que nous proposerons au chapitre 6 dispose d'une construction de type pour les streams ;
- Les réseaux SANE, que nous développerons au chapitre 7, permettront d'étudier le $\Lambda\mu$-calcul mais auront suffisamment de flexibilité pour mettre en évidence une véritable construction de stream dans les réseaux ;
- Enfin, nous définirons, au chapitre 8, un véritable calcul avec une construction de stream ; l'étude d'une machine abstraite pour $\Lambda\mu$ que nous développerons aussi au chapitre 8 va également dans ce sens.

Une autre stratégie pour retrouver la séparation du $\lambda\mu$-calcul ? Nous avons analysé au chapitre précédent diverses heuristiques pour chercher à modifier un calcul de manière à valider la propriété de séparation. Nous avons pris le parti d'élargir l'ensemble des contextes de manière à ce que le calcul obtenu ait un pouvoir d'exploration de termes

$$W_{\mathcal{C}} = \lambda x.\mu\alpha.(x) \left\{ \begin{array}{l} \mathcal{C}\left[\mu\beta.(x) \left\{ \begin{array}{l} \mu\delta.(0)\alpha \\ y \\ \alpha \end{array} \right. \right] \\ \\ \mu\delta.(0)\alpha \\ \alpha \end{array} \right.$$

FIG. 4.4 – Généralisation du contre-exemple W.

plus grand. Cette approche présentait le risque d'élargir également l'ensemble des termes à séparer et nécessitait donc d'arriver à un équilibre, un compromis : les contextes rajoutés pour réparer la non-séparation de $\lambda\mu$ doivent pouvoir séparer non seulement les $\lambda\mu$-termes, mais également les termes du nouveau calcul. Nous aurions pu adopter une direction différente consistant à rechercher à ajouter de nouvelles réductions, ou de nouvelles équivalences, sans modifier la syntaxe des termes du langage, de manière à permettre de faire coïncider la théorie équationelle de $=_{\lambda\mu\eta'}$ et l'équivalence opérationnelle. Bien sûr, là aussi nous aurions pris le risque d'élargir l'équivalence opérationnelle en même temps que l'on élargissait la théorie équationnelle mais, plus fondamentalement, il ne s'agissait peut-être pas d'une direction fructueuse. Nous donnons quelques arguments, sans résultat formel toutefois, nous conduisant à penser que cette seconde direction serait infructueuse.

La terme de la figure 4.4 est une version généralisée du contre-exemple W de David et Py. Quel que soit le contexte $\mathcal{C}$ paramétrant $W_{\mathcal{C}}$, si le terme obtenu est en forme normale canonique, on dispose d'un contre-exemple à la séparation, le résultat de David et Py s'adaptant directement. On peut ainsi cacher le facteur de non-séparation n'importe où dans le terme. Cela indique qu'une réduction ou équivalence, qui devrait égaliser les termes $W^0_{\mathcal{C}}$ et $W^1_{\mathcal{C}}$, devrait pouvoir analyser la structure globale des termes. De telles équivalences, et plus encore quand il s'agit de réductions, sont difficiles à construire. On ne voit qu'une réduction du type : $\mu\alpha.\mathcal{C}[(t)\alpha] \longrightarrow t$, mais qui présente, outre le fait de produire des variables libres (certaines variables de t peuvent être liées dans $\mathcal{C}$), le gros inconvénient d'être fortement non-déterministe, au point que la théorie équationnelle en devient aisément triviale ; on pourra à ce sujet considérer le cas du $\lambda\mu^{++}$-calcul [Nou02], mentionné en conclusion de l'article de David et Py [DP01a], avec des équivalences de la forme :

$$\begin{array}{rll} w & =_{S_6} & \mu\alpha.(\alpha((\lambda x, y.y)(\alpha w))) \\ & =_{C_\lambda} & \mu\alpha.(\alpha(\lambda y.y)) \\ & =_{C_\lambda} & \mu\alpha.(\alpha((\lambda x, y.y)(\alpha v))) \\ & =_{S_6} & v \end{array}$$

pour v, w des termes quelconques ne contenant pas de variable α libre.

Nous proposons donc la conjecture suivante :

Conjecture 4.28

Il n'est pas possible de retrouver la séparation en $\lambda\mu$-calcul en conservant la syntaxe de $\Sigma_{\lambda\mu}$ mais en élargissant l'équivalence calculatoire grâce à de nouvelles réductions à moins d'égaliser tous les termes du langage, c'est-à-dire de construire une équivalence triviale.

Une autre preuve de séparation ? La technique de preuve de Joly que nous avons adaptée au $\Lambda\mu$-calcul dans ce chapitre est originale et assez différente de preuves plus classiques et plus proches de la technique originale de Böhm [Kri90]. On pourrait également démontrer la séparation en $\Lambda\mu$-calcul de manière plus classique, en utilisant une notion de transformations de Böhm qui seraient définies comme suit :

Définition 4.29 (*Transformation de Böhm*)

Une **transformation de Böhm** *est une application des $\Lambda\mu$-termes dans les $\Lambda\mu$-termes construite par composition de transformations de Böhm élémentaires de la forme ($u \in \Lambda\mu, x \in \mathcal{V}_t$ et $\alpha, \beta \in \mathcal{V}_s$) :*
- $B^T_u(t) = (t)u$
- $B^S_\alpha(t) = (t)\alpha$
- $B^T_{u/x}(t) = t\,\{u/x\}$
- $B^S_{u\beta/\alpha}(t) = t\,\{(v)u\beta/(v)\alpha\}$

Nous ne développerons pas plus cette direction dans ce chapitre. On notera tout de même que lorsque nous étudierons les réseaux SANE, nous démontrerons un théorème de séparation pour les SANE ; cette démonstration suivra les chemins plus classiques du «Böhm out» plutôt que la preuve de Joly qui ne s'est pas révélée adaptable de manière simple à ce cadre.

Chapitre 5

Confluence du $\Lambda\mu$-calcul

Résumé:

Dans ce chapitre, nous établissons la confluence du $\Lambda\mu$-calcul introduit au chapitre 3. Nous obtenons comme corollaire une preuve simplifiée de la confluence du $\lambda\mu\eta$-calcul.

Références : Les résultats de ce chapitre ont été publiés dans l'article *On the relations between the syntactic theories of $\lambda\mu$-calculi* [Sau08a].

Sommaire

5.1. Le problème de la confluence dans les $\lambda\mu$-calculs

La confluence dans les $\lambda\mu$-calculs a toujours été une propriété délicate. Il ne faut sans doute pas s'en étonner si l'on se rappelle que le $\lambda\mu$-calcul provient originellement d'une extension de la correspondance de Curry-Howard à la logique classique, extension dont la difficulté résidait justement en ce que l'élimination des coupures en logique classique n'est pas confluente (voir le contre-exemple de Lafont en section 1.1.3).

La confluence pour le $\lambda\mu$-calcul de Parigot. Dans son article de 1992 [Par92], M. Parigot introduit les $\lambda\mu$-calculs pur et typé avec les règles β, μ, ρ, θ. Il énonce la confluence du calcul pur et en propose une preuve qui comporte malheureusement quelques lacunes car la méthode des réductions parallèles qui y est présentée n'est pas suffisamment puissante pour assurer la confluence du calcul. En effet, la réduction parallèle introduite dans [Par92] n'est pas suffisament générale pour obtenir une confluence forte et donc assurer la confluence à partir de la seule confluence locale. Une preuve complète est donnée par Walter Py [Py98, DP01a].

La confluence pour des variantes de $\lambda\mu$. Ph. de Groote, quant à lui, propose une extension du $\lambda\mu$-calcul de Parigot [dG98] mais là encore la confluence pose des problèmes et force à restreindre certaines règles (comme la règle ρ dont l'application doit être contrainte en présence de la réduction ϵ).

Les deux calculs mentionnés ci-dessus ne comportent pas de réduction η. Le premier à fournir une étude complète de la confluence en $\lambda\mu$-calcul, avec ou sans règle η est Walter Py dans sa thèse de doctorat [Py98]. Py comble la lacune de la preuve de Parigot en développant une méthode de «réductions parallèles généralisées» et montre des résultats de confluence pour le calcul $\lambda\mu\epsilon$ de Ph. de Groote (avec ϵ mais sans η) ainsi que pour une version du $\lambda\mu$-calcul en appel par valeur. Par ailleurs, intéressé par la question du théorème de Böhm en $\lambda\mu$-calcul de Parigot, Py ajoute la η-réduction au calcul et l'adapte de manière à conserver la confluence. Il fournit en particulier une preuve de confluence pour le $\lambda\mu$-calcul de Parigot avec les règles β, μ, ρ, θ, ν et η. Il étudie aussi le $\lambda\mu$-calcul par valeur.

Baba, Hirokawa et Fujita [BHeF01] montrent également, un peu après Py, un résultat légèrement plus faible que le sien puisqu'ils prouvent la confluence pour un $\lambda\mu$-calcul sans θ. Ils s'intéressent également au cas du $\lambda\mu$-calcul par valeur.

5.2. Rappels sur la confluence : définitions et résultats de base

On rappelle dans cette section quelques définitions de base et lemmes clés qui serviront dans la suite du chapitre. Pour plus de détails concernant les questions de confluence, on renvoie le lecteur à l'ouvrage très complet de Terese [Ter03] ainsi qu'au premier chapitre de la thèse de Walter Py [Py98] qui fournit une bonne introduction à ces questions, claire et synthétique.

Définition 5.1 (*Clôtures d'une réduction*)

Soit $\rightarrow$ une réduction d'un système de réécriture.
On définit les clôtures de $\rightarrow$ ainsi que les réductions dérivées suivantes :

- $\rightarrow^=$ *est la* **clôture réflexive** *de $\rightarrow$: $t \rightarrow^= u$ si, et seulement si $t = u$ ou $t \rightarrow u$;*
- $\rightarrow^+$ *est la* **clôture transitive** *de $\rightarrow$: $t \rightarrow^+ u$ si, et seulement si $t \rightarrow u$ ou s'il existe v tel que $t \rightarrow v \rightarrow^+ u$;*
- $\rightarrow^\star$ *est la* **clôture réflexive-transitive** *de $\rightarrow$ (ou* **$\star$-clôture***) : $t \rightarrow^\star u$ si, et seulement si $t = u$ ou $t \rightarrow^+ u$;*
- $\rightarrow^n$*, pour un entier n, est définie par $t \rightarrow^n u$ s'il existe une suite de longueur $n+1$ de termes : $v_0, v_1, \ldots, v_n$ telle que $v_0 = t$, $v_n = u$ et pour tout $i < n, v_i \rightarrow v_{i+1}$;*
- $\rightarrow_{1,2}$ *est l'***union des deux réductions** *$\rightarrow_1$ et $\rightarrow_2$: $t \rightarrow_{1,2} u$ si, et seulement si, $t \rightarrow_1 u$ ou $t \rightarrow_2 u$.*

5.2.1 Confluence

Définition 5.2 (*Confluence*)

Un sous-système $\rightarrow$ d'un système de réécriture est dit **confluent** *si pour tout triplet (t, t', t'') de termes du système, si $t \rightarrow^\star t'$ et $t \rightarrow^\star t''$ alors il existe un terme u tel que $t' \rightarrow^\star u$ et $t'' \rightarrow^\star u$. On notera souvent cette assertion de la manière diagrammatique suivante qui est commode et généralement non ambiguë :*

$$\begin{tikzcd} t \arrow[r, "\star"] \arrow[d, "\star"] & t' \arrow[d, dotted, "\star"] \\ t'' \arrow[r, dotted, "\star"] & u \end{tikzcd}$$

Définition 5.3 (*Propriété de Church-Rosser*)

Un sous-système $\rightarrow$ d'un système de réécriture est dit **Church-Rosser** *si lorsque deux termes t et u sont équivalents dans la théorie équationnelle associée à $\rightarrow$ alors il existe un terme v tel que $t, u \rightarrow^\star v$.*

La propriété de confluence n'est pas forcément simple à manipuler puisqu'elle concerne des séquences de réduction arbitrairement longues à partir de t avant de chercher à faire confluer vers un même réduit u. Deux notions dérivées, l'une plus faible que la confluence, l'autre plus forte, sont de manipulation plus aisée parce qu'elles sont locales : l'hypothèse de la définition concerne des réductions de longueur 1 à partir de t.

Définition 5.4 (*Confluence locale*)

Un sous-système $\rightarrow$ d'un système de réécriture est dit **localement confluent** *si :*

$$\begin{tikzcd} t \arrow[r] \arrow[d] & t' \arrow[d, dotted, "\star"] \\ t'' \arrow[r, dotted, "\star"] & u \end{tikzcd}$$

Définition 5.5 (***Confluence forte***)

Un sous-système $\rightarrow$ d'un système de réécriture est dit **fortement confluent** *si :*

$$\begin{tikzcd} t \arrow[r] \arrow[d] & t' \arrow[d, dotted] \\ t'' \arrow[r, dotted] & u \end{tikzcd}$$

Propriété 5.6

On a la série d'implications suivante :

$$\rightarrow \text{ fortement confluente} \Rightarrow \quad \rightarrow \text{ confluente} \Rightarrow \quad \rightarrow \text{ faiblement confluente}$$

Aucune autre implication n'est valable dans le cas général.

Il est immédiat qu'un système de réécriture fortement confluent est confluent, mais ce n'est pas vrai de la confluence locale qui n'entraîne la confluence qu'avec des conditions supplémentaires. En particulier, la méthode des réductions parallèles (ou méthode de Tait-Martin Löf) consiste à remplacer un système de réécriture localement confluent par un système fortement confluent qui a la même $\star$-clôture et à utiliser ensuite la première implication de la propriété 5.6.

5.2.2 Commutation

Définition 5.7 (***Commutation***)

Deux sous-systèmes $\rightarrow_1$ et $\rightarrow_2$ d'un système de réécriture **commutent** *si*

$$\begin{tikzcd} t \arrow[r, "\star", "2"'] \arrow[d, "1"', "\star"] & t' \arrow[d, dotted, "1"', "\star"] \\ t'' \arrow[r, dotted, "\star", "2"'] & u \end{tikzcd}$$

On parlera de **commutation forte** *lorsque le diagramme est vérifié sans les $\star$-clôtures.*

Lemme 5.8 (***de Hindley-Rosen***)

Soient deux sous-systèmes $\rightarrow_1$ et $\rightarrow_2$ d'un système de réécriture. Si $\rightarrow_1$ et $\rightarrow_2$ sont tous deux confluents et si ils commutent, alors $\rightarrow_{1,2}$ est confluent.

Pour mémoire, on rappelle la démonstration du lemme :

Démonstration : Soient t, t', t'' tels que $t \rightarrow^\star_{1,2} t', t''$. On raisonne par récurrence sur la somme $n = n' + n''$ du nombre d'alternances entre $\rightarrow_1$ et $\rightarrow_2$ sur $t \rightarrow^\star_{1,2} t'$ et sur $t \rightarrow^\star_{1,2} t''$.

Si $n = 0$, on a le résultat en utilisant soit la commutation des deux systèmes, soit la confluence de $\rightarrow_1$, soit la confluence de $\rightarrow_2$ selon le cas.

Supposons le résultat vrai pour tout entier inférieur ou égal à un certain entier n, on montre qu'il est alors vrai pour le successeur de ce nombre : on a par exemple $t \rightarrow^\star_{1,2}\rightarrow^+_1 t'_1 \rightarrow^+_2 t'$. Par induction on a l'existence d'un u_1 tel que $t'', t'_1 \rightarrow^\star_{1,2} u_1$. Grâce à la confluence de $\rightarrow_2$ et à la commutation des systèmes, on a l'existence d'un u tel que $t' \rightarrow^\star_{1,2} u$ et $u_1 \rightarrow^\star_2 u$, d'où le résultat : $t' \rightarrow^\star_{1,2} u$ et $t'' \rightarrow^\star_{1,2} u$.

On en déduit finalement la confluence de $\rightarrow_{1,2}$.

□

Lemme 5.9 (*Commutation de l'union de deux réductions*)

Si trois sous-systèmes $\to_1$, $\to_2$ et $\to_3$ d'un système de réécriture sont tels que $\to_1$ commute avec $\to_3$ et $\to_2$ commute avec $\to_3$, alors $\to_{1,2}$ commute avec $\to_3$.

Démonstration : Soient t, t', t'' tels que $t \to_3^\star t'$ et $t \to_{1,2}^\star t''$. On cherche u tel que $t' \to_{1,2}^\star u$ et $t'' \to_3^\star u$. Le résultat s'obtient facilement par induction sur le nombre d'alternances entre des séquences de réductions de $\to_1$ et de $\to_2$ de t à t''.

□

Lemme 5.10 (*Variante de la commutation de l'union de deux réductions*)

Si trois sous-systèmes $\to_1$, $\to_2$ et $\to_3$ d'un système de réécriture sont tels que

1. *$\to_1$ commute avec $\to_3$;*
2. $$\begin{tikzcd} t \arrow[r, "\star", "3"'] \arrow[d, "2"', "\star"] & t' \arrow[d, dotted, "1{,}2"', "\star"] \\ t'' \arrow[r, dotted, "\star", "3"'] & u \end{tikzcd}$$

alors $\to_{1,2}$ commute avec $\to_3$.

Démonstration : Soient t, t', t'' tels que :

$$\begin{tikzcd} t \arrow[r, "\star", "1{,}2"'] \arrow[d, "3"', "\star"] & t' \\ t'' & \end{tikzcd}$$

On raisonne par induction sur le nombre n d'alternances entre des réductions dans $\to_1$ et $\to_2$ sur $t \to_{1,2} t'$: si $n = 0$ alors on a le résultat par la propriété 1 si $t \to_1 t'$ et par la propriété 2 sinon.

Supposons maintenant que le résultat soit vrai lorsqu'on passe de t à t' en changeant n fois de système. Soit $t \to_{1,2} t'$ une réduction avec $n+1$ alternances, par exemple : $t \to_{1,2} t_1 \to_1^+ t_2 \to_2^+ t'$ on a alors :

$$\begin{tikzcd} t \arrow[r, "\star", "1{,}2"'] \arrow[d, "3"', "\star"] \arrow[dr, phantom, "HR(n)"] & t_2 \arrow[r, "+", "2"'] \arrow[d, dotted, "3"', "\star"] \arrow[dr, phantom, "Hyp\ 2."] & t' \arrow[d, dotted, "3"', "\star"] \\ t'' \arrow[r, dotted, "\star", "1{,}2"'] & t_2'' \arrow[r, dotted, "\star", "1{,}2"'] & u \end{tikzcd}$$

En effet, on peut appliquer l'hypothèse de récurrence à t, t_2, t'' puisqu'il y a n alternances de t à t_2 ; on obtient ainsi t_2''. On peut enfin appliquer la seconde hypothèse du lemme pour obtenir u et clore le diagramme de commutation (si la réduction de t à t' s'était achevée sur une séquence de $\to_1$, on aurait également pu clore le diagramme grâce à l'hypothèse de commutation de $\to_1$ et $\to_3$).

□

Remarque 5.11

On pourrait évidemment étendre le précédent lemme au cas où aucune des deux réductions ne commute avec $\to_3$ mais que toutes deux commutent «à travers leur union» comme dans la seconde condition du lemme précédent. On ne détaille pas ce lemme plus général car le lemme précédent sera suffisant pour la confluence du $\Lambda\mu$-calcul.

5.3. Confluence en $\Lambda\mu$-calcul

On montre dans cette section la confluence du $\Lambda\mu$-calcul restreint aux termes sans variable de stream libre :

Théorème 5.12 (*Confluence du $\Lambda\mu$-calcul*)

Le $\Lambda\mu$-calcul est confluent pour les termes μ-clos :
Pour tous t, t', t'' $\Lambda\mu$-termes μ-clos, il existe $u \in \Lambda\mu$ tel que

$$\begin{tikzcd} t \arrow[r, "\star", "\Lambda\mu"'] \arrow[d, "\Lambda\mu"', "\star"] & t' \arrow[d, dotted, "\Lambda\mu"', "\star"] \\ t'' \arrow[r, dotted, "\star", "\Lambda\mu"'] & u \end{tikzcd}$$

On va démontrer le théorème 5.12 en considérant des sous-systèmes de réduction de $\Lambda\mu$ qui sont confluents et qui commutent. Plus précisément, la confluence du $\Lambda\mu$-calcul sera obtenue grâce aux deux résultats préliminaires suivants :

(i) la confluence de $\boldsymbol{\beta}$***fst*** (obtenue en section 5.3.2, proposition 5.23),

(ii) la commutation de $\boldsymbol{\beta}$***fst*** et de $\boldsymbol{\eta}$ (obtenue en section 5.3.4, proposition 5.25).

En effet, la confluence de $\boldsymbol{\eta}$ qui s'obtient facilement (proposition 5.24) assure, via le lemme d'Hindley-Rosen la confluence de $\Lambda\mu$.

5.3.1 Restriction au cas μ-clos

Les problèmes de confluence en $\lambda\mu$-calcul proviennent, comme on peut s'y attendre, des réductions qui impliquent à la fois des variables de terme et des variables de stream, c'est-à-dire à la fois des λ-abstractions et des μ-abstractions.

C'est grâce à la règle *fst* que les deux types de variables sont connectées et c'est de cette réduction que naissent les problèmes de confluence. la réduction *fst*, ou plutôt la réduction ν, a été introduite en $\lambda\mu$-calcul pour résoudre une paire critique entre μ et η, mais encore faut-il que le terme ne contienne pas de variable de stream libre.

En effet, on a une paire critique entre la réduction *fst* et la réduction β_S :

$$\begin{tikzcd} (\mu\alpha.t)\beta \arrow[r, "fst"'] \arrow[d, "\beta_S"'] & (\lambda x.\mu\gamma.t\,\{(u)x\gamma/(u)\alpha\})\beta \\ t\,\{\beta/\alpha\} & \end{tikzcd}$$

qui nécessite, pour pouvoir confluer, que la variable β soit liée : pour faire confluer cette paire critique, il faut créer un β_T-redex au niveau du λx de manière à créer le β_S-redex entre $(\mu\gamma.t')\beta'$ tandis que la variable β de $t\,\{\beta/\alpha\}$ se voit également appliquer une règle *fst*. Mais ceci n'est possible que si la variable β est liée...

5.3.2 Confluence de la $\boldsymbol{\beta}$*fst*-réduction

Pour montrer la confluence de la $\boldsymbol{\beta}$*fst*-réduction, on a besoin d'un certain nombre de résultats préliminaires :

Lemme 5.13

> 1. *β_T et $\boldsymbol{\beta}\!fst$ commutent.*
> 2. *fst et $\beta_T fst$ commutent.*

Démonstration :

1. β_T commute avec β_T, avec fst et avec β_S ; en effet, on a :
   - β_T/β_T : car β_T est confluente dans $\Lambda\mu$ (identique au cas du λ-calcul).
   - β_T/fst : il n'y a pas de paire critique entre β_T et fst, seul le rédex fst peut être dupliqué par la réduction β_T et on a :

$$\begin{tikzcd} t \arrow[r, "fst"'] \arrow[d, "\beta_T"'] & t' \arrow[d, dotted, "\beta_T"'] \\ t'' \arrow[r, dotted, "\star", "fst"'] & u \end{tikzcd}$$

     qui assure la commutation ;
   - β_T/β_S : il n'y a pas de paire critique entre β_T et β_S, seul le rédex β_S peut être dupliqué par la réduction β_T et on a donc :

$$\begin{tikzcd} t \arrow[r, "\beta_S"'] \arrow[d, "\beta_T"'] & t' \arrow[d, dotted, "\beta_T"'] \\ t'' \arrow[r, dotted, "\star", "\beta_S"'] & u \end{tikzcd}$$

     qui assure la commutation.

   Le lemme de commutation de l'union de réductions nous permet de conclure que β_T commute avec $\boldsymbol{\beta}\!fst$.
2. On a les faits suivants :
   - fst/fst : fst est fortement confluente ;
   - fst/β_T : on a le diagramme de commutation suivant entre fst et β_T :

$$\begin{tikzcd} t \arrow[r, "\star", "fst"'] \arrow[d, "\beta_T"'] & t' \arrow[d, dotted, "\beta_T"'] \\ t'' \arrow[r, dotted, "\star", "fst"'] & u \end{tikzcd}$$

   D'après le lemme 5.9 sur la commutation de l'union de réductions (page 107), on a donc le résultat.

□

On va maintenant montrer un résultat entre fst et β_S qui nous permettra de déduire la commutation de fst avec $\boldsymbol{\beta}\!fst$. Il ne s'agit pas d'une commutation entre fst et β_S car il y a une paire critique entre ces deux réductions que l'on ne peut pas faire confluer avec ces deux seules règles :

$$\begin{tikzcd} C[(\mu\alpha.t)\beta] \arrow[r, "fst"'] \arrow[d, "\beta_S"'] & C[(\lambda x.\mu\gamma.t\,\{(u)x\gamma/(u)\alpha\})\beta] \\ C[t\,\{\beta/\alpha\}] & \end{tikzcd}$$

On peut faire confluer cette paire critique de la manière suivante : on sait que le terme de départ n'a pas de variable de stream libre, il y a donc un μ qui lie la variable β, $C[] = C_1[\mu\beta.C_2[]]$.

On applique une *fst* réduction à cette abstraction dans $C_1[\mu\beta.C_2[t\,\{\beta/\alpha\}]]$ ce qui nous donne $C_1[\lambda z.\mu\delta.(C_2[t\,\{\beta/\alpha\}]\,\{(u)z\delta/(u)\beta\})] = C_1[\lambda z.\mu\delta.(C_2[t\,\{(u)z\delta/(u)\alpha\}]\,\{(u)z\delta/(u)\beta\})]$. On applique alors une séquence $fst; \beta_T; \beta_S$ à $C_1[\mu\beta.C_2[(\lambda x.\mu\gamma.t\,\{(u)x\gamma/(u)\alpha\})\beta]]$ de manière à obtenir le diagramme suivant :

$$\begin{tikzcd}
C[(\mu\alpha.t)\beta] \arrow[r, "fst"'] \arrow[ddd, "\beta_S"'] & C_1[\mu\beta.C_2[(\lambda x.\mu\gamma.t\,\{(u)x\gamma/(u)\alpha\})\beta]] \arrow[d, dotted, "fst"'] \\
& C_1[\lambda z.\mu\delta.(C_2[(\lambda x.\mu\gamma.t\,\{(u)x\gamma/(u)\alpha\})z\delta])\,\{(u)z\delta/(u)\beta\}] \arrow[d, dotted, "\beta_T"'] \\
& C_1[\lambda z.\mu\delta.(C_2[(\mu\gamma.t\,\{(u)z\gamma/(u)\alpha\})\delta])\,\{(u)z\delta/(u)\beta\}] \arrow[d, dotted, "\beta_S"'] \\
C_1[\mu\beta.C_2[t\,\{\beta/\alpha\}]] \arrow[r, dotted, "fst"'] & C_1[\lambda z.\mu\delta.(C_2[t\,\{(u)z\delta/(u)\alpha\}])\,\{(u)z\delta/(u)\beta\}]
\end{tikzcd}$$

On remarque que la règle β_T qui est appliquée ici est une instance particulièrement simple puisqu'elle substitue une variable par une variable, notée β_T^{var} et on remarquera que, contrairement au cas général de β_T, β_T^{var} ne peut pas dupliquer de rédex. Ce dernier point sera essentiel pour le reste de la preuve.

Lemme 5.14

$$\begin{tikzcd}
t \arrow[r, "\star", "fst"'] \arrow[d, "\beta_S"', "\star"] & t' \arrow[d, dotted, "\boldsymbol{\beta}^{var} fst"', "\star"] \\
t'' \arrow[r, dotted, "\star", "fst"'] & u
\end{tikzcd}$$

Démonstration : Dans le cas $\begin{tikzcd} t \arrow[r, "fst"'] \arrow[d, "\beta_S"'] & t' \\ t'' & \end{tikzcd}$, on a trois cas possibles :

1. dans le cas que l'on a examiné avant l'énoncé du lemme (t contient une μ-abstraction $\mu\alpha$ sur laquelle s'opère à la fois la *fst*-réduction vers t' et la β_S-réduction vers t''), on fait confluer le diagramme en utilisant respectivement une règle *fst* à partir de t'' et une séquence $fst; \beta_T^{var}; \beta_S$ à partir de t'.
2. dans le cas dual du cas précédent (la variable β qui subit la *fst*-réduction vers t' est argument du β_S-rédex menant à t''), on a :

$$\begin{tikzcd}
C_1[\mu\beta.C_2[(\mu\alpha.t)\beta]] \arrow[r, "fst"'] \arrow[d, "\beta_S"'] & C_1[\lambda x.\mu\gamma.(C_2[(\mu\alpha.t)x\gamma])\,\{(u)x\gamma/(u)\beta\}] \\
C_1[\mu\beta.C_2[t\,\{\beta/\alpha\}]] &
\end{tikzcd}$$

On fait alors confluer le diagramme de manière sensiblement similaire au cas précédent :
– on applique une *fst*-réduction en $\mu\beta$ dans t'' et
– une séquence $fst; \beta_T^{var}; \beta_S$ dans t' :

(i) on réduit le *fst*-redex en $\mu\alpha$ ce qui crée un β_T^{var}-rédex en produisant le terme $C_1[\lambda x.\mu\gamma.(C_2[(\lambda z.\mu\delta.t\,\{(u)z\delta/(u)\alpha\})x\gamma])\,\{(u)x\gamma/(u)\beta\}]$;

(ii) on effectue la réduction de ce β_T^{var}-rédex ce qui crée un β_S-rédex et finalement

(iii) on achève la séquence de réduction par une β_S-réduction produisant le terme $C_1[\lambda x.\mu\gamma.(C_2[t\,\{(u)x\gamma/(u)\alpha\}])\,\{(u)x\gamma/(u)\beta\}]$

qui est précisément le terme obtenu par *fst*-réduction à partir de t''.

3. Enfin, le cas où la $\mathcal{V}_S$ variable qui subit la réduction *fst* n'est pas en position d'argument dans le β_S-rédex et où les deux rédex ne partagent pas la même μ-abstraction nous permet d'avoir un diagramme de commutation forte puisqu'il n'y a pas de paire critique et pas de duplication :

$$\begin{tikzcd} t \arrow[r, "fst"'] \arrow[d, "\beta_S"'] & t' \arrow[d, dotted, "\beta_S"'] \\ t'' \arrow[r, dotted, "fst"'] & u \end{tikzcd}$$

On a donc le diagramme suivant :

$$\begin{tikzcd} t \arrow[rrr, "\beta_S"'] \arrow[d, "fst"'] & & & t'' \arrow[d, dotted, "fst"'] \\ t' \arrow[r, dotted, "=", "fst"'] & \cdot \arrow[r, dotted, "=", "\beta_T^{var}"'] & \cdot \arrow[r, dotted, "\beta_S"'] & u \end{tikzcd}$$

La confluence forte de *fst* et la commutation forte[1] de *fst* et β_T^{var} assurent que l'on peut faire commuter une séquence arbitrairement longue de réduction *fst* avec une réduction β_S :

$$\begin{tikzcd} t \arrow[rr, "\beta_S"'] \arrow[d, "fst"', "\star"] & & t'' \arrow[d, dotted, "fst"', "\star"] \\ t' \arrow[r, dotted, "\star", "fst\beta_T^{var}"'] & \cdot \arrow[r, dotted, "\beta_S"'] & u \end{tikzcd}$$

On obtient finalement par une induction triviale sur la longueur de la β_S-réduction de t à t'', la confluence du diagramme suivant :

$$\begin{tikzcd} t \arrow[r, "\star", "\beta_S"'] \arrow[d, "fst"', "\star"] & t'' \arrow[d, dotted, "fst"', "\star"] \\ t' \arrow[r, dotted, "\star", "\boldsymbol{\beta}^{var}fst"'] & u \end{tikzcd}$$

ce qui nous donne le résultat.

□

On a comme conséquence simple de la commutation de β_T^{var} avec $\boldsymbol{\beta}^{var}fst$ et du résultat précédent :

Lemme 5.15

$$\begin{tikzcd} t \arrow[r, "\star", "\beta_T^{var}fst"'] \arrow[d, "\beta_S"', "\star"] & t' \arrow[d, dotted, "\boldsymbol{\beta}^{var}fst"', "\star"] \\ t'' \arrow[r, dotted, "\star", "\beta_T^{var}fst"'] & u \end{tikzcd}$$

[1] la commutation simple de *fst* et β_T (cf. lemme 5.13, page 109) devient commutation forte avec β_T^{var} puisque le *fst*-rédex ne peut plus être dupliqué ou effacé.

Lemme 5.16

1. *fst et $\boldsymbol{\beta}$fst commutent ;*
2. *$\beta_T$fst et $\boldsymbol{\beta}$fst commutent ;*
3. *$\beta_T$fst et $\boldsymbol{\beta}^{var}$fst commutent.*

Démonstration :

1. Il s'agit d'une conséquence immédiate des lemmes 5.13 et 5.14 qui permettent d'appliquer le lemme 5.10 page 107 (variante de la commutation d'une union de réductions) et d'obtenir la commutation souhaitée.
2. Il s'agit une nouvelle fois d'appliquer le lemme de commutation de l'union de deux réductions puisque β_T et *fst* commutent avec $\boldsymbol{\beta}$*fst*, leur union β_T*fst* commute avec $\boldsymbol{\beta}$*fst*.
3. La dernière partie du lemme est obtenue, grâce au résultat sur la commutation de l'union de deux réductions, comme corollaire de la commutation de β_T (resp. *fst*) avec $\boldsymbol{\beta}^{var}$*fst*, qui est elle-même obtenue de manière similaire au lemme 5.13 (resp. au premier point du présent lemme).

□

5.3.3 $\Lambda\mu$-termes indexés

Comme on l'a vu au lemme 5.14, la commutation de *fst* avec β_S est légèrement problématique car on ne peut avoir que la confluence à travers une autre réduction plus large puisqu'on a besoin de *fst* et β_S (pour tous termes t, t', t'' tels que $t \to_{fst} t'$ et $t \to_{\beta_S} t''$, il existe u tel que $t' \to_{(fst;\beta_T^{var})^=;\beta_S} u$ et $t'' \to_{fst} u$).

Cela rend délicate la preuve du fait que $\boldsymbol{\beta}^{var}$*fst* commute avec $\boldsymbol{\beta}$*fst*. On va pour cela passer par une sous-réduction de *fst* qui terminera ; on aura alors trois réductions terminantes (*fst*, β_S et β_T^{var}) ce qui rendra beaucoup plus simples les résultats de commutation.

Pour cela, il faut annoter les $\Lambda\mu$-termes avec un ***potentiel de fst-expansion*** :

Définition 5.17

On considère la variante suivante du $\Lambda\mu$-calcul, où i prend ses valeurs dans les entiers relatifs :

$$t ::= \ x \mid \lambda x.t \mid \mu^i\alpha.t \mid (t)\alpha \mid (t)u$$

Seule la fst réduction est affectée par cette variation de la manière suivante. fst est remplacée par la fst_ι-réduction qui ne peut être appliquée que si l'indice du μ est strictement positif et qui décroît cet indice :

$$\mu^{i+1}\alpha.t \longrightarrow_{fst_\iota} \lambda x.\mu^i\beta.t\,\{(u)x\beta/(u)\alpha\} \qquad \text{avec } i \geq 0.$$

Les autres règles sont inchangées : par exemple, la réduction β_S est maintenant

$$(\mu^i\alpha.t)\beta \longrightarrow_{\beta_S} t\,\{\beta/\alpha\}\,.$$

A priori, les indices, s'ils sont mal placés, peuvent amener à des anomalies lors de la β_S-réduction qui font échouer la confluence de $\boldsymbol{\beta}^{var}fst_\iota$ pour le $\Lambda\mu$-calcul avec indices. En effet, si on considère le terme $t = \mu^1\alpha.(\mu^0\beta.x)\alpha$, alors on a non-confluence car t se réduit d'une part en $\lambda y.\mu^0\alpha.(\mu^0\beta.x)y\alpha$ par une fst_ι-réduction et d'autre part en $\lambda y.\mu^0\alpha.x$ par une β_S-réduction suivie d'une fst_ι-réduction, les deux termes étant des formes normales pour $\boldsymbol{\beta}^{var}fst_\iota$. Le problème vient de ce que l'on peut faire une β_S-réduction en faisant communiquer des variables de stream qui n'ont pas le même potentiel de *fst*-expansion.

On va donc considérer un sous-ensemble du calcul où les indexations sont cohérentes, et qui est obtenu comme l'image de l'ensemble des $\Lambda\mu$-termes de base par une indexation cohérente avec la $\boldsymbol{\beta}^{var}fst$-réduction (mais pas avec la réduction β_T générale qui ne nous intéresse pas ici) :

Définition 5.18 (*Indexation d'un $\Lambda\mu$-terme.*)

Soit t un $\Lambda\mu$-terme quelconque dont les variables de stream libres sont dans V.

On se donne une fonction partielle ρ des variables de stream dans les entiers, définie pour toutes les variables de stream libres de t, ainsi qu'une liste d'entiers l.

On commence par annoter tout sous-terme de t en position d'argument par une liste d'entiers (de la forme : $(u)v^l$), obtenant ainsi t'.

*L'***indexation de** *t est obtenue à partir de t' selon la définition récursive suivante de $[t']^l_\rho$:*

- $[x]^l_\rho = x$;
- $[\lambda x.t]^{n::l}_\rho = \lambda x.[t]^{(n-1)::l}_\rho$;
- $[\lambda x.t]^{\emptyset}_\rho = \lambda x.t$;
- $[(t)u^{l'}]^{n::l}_\rho = ([t]^{(n+1)::l}_\rho)[u]^{l'}_\rho$;
- $[(t)u^{l'}]^{\emptyset}_\rho = (t)u$;
- $[\mu\alpha.t]^{n::l}_\rho = \mu^n\alpha.[t]^l_{\rho\cup\{\alpha\leftarrow n\}}$;
- $[\mu\alpha.t]^{\emptyset}_\rho = \mu\alpha.t$;
- $[(t)\alpha]^l_\rho = ([t]^{\rho(\alpha)::l}_\rho)\alpha$.

Définition 5.19 ($\Lambda\mu$*-terme indexé,* $\Lambda\mu^\iota$*-terme.*)

*Un $\Lambda\mu$-***terme indexé** *t est un terme de la syntaxe avec indice qui provient de la traduction d'un $\Lambda\mu$-terme : $t = [u]^l_\rho$ pour certains u, ρ, l.*

On parlera de **$\Lambda\mu^\iota$-termes** *pour les $\Lambda\mu$-termes indexés.*

Proposition 5.20

(i) À tout $\Lambda\mu$-terme t peut être associée une indexation ι telle que toutes les μ-abstractions reçoivent un indice qui est de plus positif : on peut donc associer à tout $\Lambda\mu$-terme un $\Lambda\mu^\iota$-terme t^ι tel que tous les μ ont des indices positifs.

(ii) Si t est un $\Lambda\mu^\iota$-terme μ-clos et que $t \longrightarrow_{\boldsymbol{\beta}^{var}fst_\iota} u$, alors u est également un $\Lambda\mu^\iota$-terme;

(iii) La réduction $\boldsymbol{\beta}^{var}fst_\iota$ termine pour les $\Lambda\mu$-termes indexés;

(iv) Si $t, t_1, \ldots, t_n$ sont des $\Lambda\mu$-termes tels que $t \longrightarrow^\star_{\boldsymbol{\beta}^{var}fst} t_j$ quel que soit $1 \le j \le n$, il existe une indexation de t telle que les μ-abstractions des t_j reçoivent tous des indices positifs et que les réductions puissent être réalisées dans $\boldsymbol{\beta}^{var}fst_\iota$.

Démonstration :

(i) Il suffit de choisir des listes suffisamment longues d'indices suffisamment grands.

(ii) On vérifie simplement que les trois règles β_T, β_S et fst_ι préservent la propriété d'être résultat d'une indexation d'un $\Lambda\mu$-terme :

- *Cas de β_S :* Soit t^ι un $\Lambda\mu^\iota$-terme obtenu par indexation d'un $\Lambda\mu$-terme t et supposons $t^\iota \longrightarrow_{\beta_S} v$. On a $t^\iota = \mathcal{C}^\iota\left[(\mu^i\alpha.t')\beta\right]$ et $v = \mathcal{C}^\iota[t'\{\beta/\alpha\}]$. Dans t^ι, le lieur de l'occurrence de β intervenant dans la réduction porte le même indice

i que $\mu^i\alpha.t$ par définition de l'indexation et v est donc obtenu par indexation d'un $\Lambda\mu$-terme u ;

– *Cas de* β_T^{var} *:* Soit t^ι un $\Lambda\mu^\iota$-terme obtenu par indexation d'un $\Lambda\mu$-terme t et supposons $t^\iota \longrightarrow_{\beta_T^{var}} v$. On a $t^\iota = \mathcal{C}^\iota[(\lambda x.t')y]$ et $v = \mathcal{C}^\iota[t'\{y/x\}]$. Considérant la définition de l'indexation d'un $\Lambda\mu$-terme, on constate que : $[(\lambda x.t')y^{l'}]_\rho^{n::l} = ([\lambda x.t']_\rho^{(n+1)::l})y = (\lambda x.[t']_\rho^{n::l})y$ et $[t'\{y/x\}]_\rho^{n::l} = [t']_\rho^{n::l}\{y/x\}$.

– *Cas de* fst_ι *:* Soit t^ι un $\Lambda\mu^\iota$-terme obtenu par indexation d'un $\Lambda\mu$-terme t et supposons $t^\iota \longrightarrow_{fst_\iota} v$. En supposant t^ι de la forme $\mathcal{C}^\iota\left[\mu^{i+1}\alpha.t'\right]$, on a $v = \mathcal{C}^\iota\left[\lambda x.\mu^i\alpha.t'\{(u)x\alpha/(u)\alpha\}\right]$. On vérifie aisément que, d'après la définition 5.18, la compatibilité des indices est maintenue grâce à l'ajout de la variable x en argument de tous les sous-termes u auxquels était appliquée une variable de stream α.

(iii) Évident : $\boldsymbol{\beta}^{var}$ termine et chaque occurrence de fst_ι diminue l'indice de l'une des μ-abstractions. Il suffit de prendre pour taille des termes, une paire de la somme des indices des μ-abstractions et du nombre de $\boldsymbol{\beta}^{var}$-rédex contenus dans un terme et de constater que chaque réduction diminue la taille pour l'ordre lexicographique.

(iv) On montre le résultat par récurrence sur le nombre n de $\boldsymbol{\beta}^{var}fst$-réductions partant de t considérées. Le cas de base est immédiat en donnant une indexation à t_1 en un $\Lambda\mu^\iota$-terme dont tous les indices sont positifs et en "remontant" la réduction $t \longrightarrow^\star_{\boldsymbol{\beta}^{var}fst} t_1$ de manière à obtenir une indexation de t compatible avec la réduction considérée. On remarque ensuite que si une $\boldsymbol{\beta}^{var}fst_\iota$-réduction r est permise par une indexation de t, toute indexation qui donne des entiers plus grands ou utilise des listes plus longues fournit une indexation compatible avec la réduction r. Ce qui permet de traiter le cas inductif.

□

Par ailleurs, la contrainte de cohérence des indices imposée par les indexations assure que tous les diagrammes de commutation ou de confluence qu'on pouvait clore dans $\boldsymbol{\beta}^{var}fst$ peuvent être clos dans $\boldsymbol{\beta}^{var}fst_\iota$ aussi, ainsi β_S et $\boldsymbol{\beta}^{var}$ sont fortement confluents et :

Lemme 5.14.ι

Si t, t', t'' sont des $\Lambda\mu$-termes indexés tels que $t \rightarrow^\star_{fst_\iota} t'$ et $t \rightarrow^\star_{\beta_S} t''$, il existe u tel que $t' \rightarrow^\star_{\boldsymbol{\beta}^{var}fst_\iota} u$ et $t'' \rightarrow^\star_{fst_\iota} u$.

et

Lemme 5.15.ι

Si t, t', t'' sont des $\Lambda\mu$-termes indexés tels que $t \rightarrow^\star_{\beta_T^{var}fst_\iota} t'$ et $t \rightarrow^\star_{\beta_S} t''$, il existe u tel que $t' \rightarrow^\star_{\boldsymbol{\beta}^{var}fst_\iota} u$ et $t'' \rightarrow^\star_{\beta_T^{var}fst_\iota} u$.

Démonstration : Ces lemmes sont obtenus en modifiant les preuves des lemmes 5.14 et 5.15 d'à peine plus d'un iota...

□

Dans la suite, on fera référence à la version d'un lemme sur les $\Lambda\mu^\iota$-termes en faisant suivre le numéro du lemme d'un ι.

Dans le lemme suivant, on notera $\longrightarrow_1$ pour $\longrightarrow_{\beta_S}$ et $\longrightarrow_2$ pour $\longrightarrow_{\beta_T^{var}fst_\iota}$, le résultat 5.15.$\iota$ se réécrit : Si t, t', t'' sont tels que $t \rightarrow^\star_2 t'$ et $t \rightarrow^\star_1 t''$, il existe u tel que $t' \rightarrow^\star_{1,2} u$ et $t'' \rightarrow^\star_2 u$.

On est intéressé par le résultat suivant qui exprime la confluence de $\boldsymbol{\beta}^{var}fst_\iota$ sur les $\Lambda\mu^\iota$-termes :

$$\begin{tikzcd}
t \arrow[r, "i"'] \arrow[d, "j"'] \arrow[dr, phantom, "Cas\ i/j."] & t'_1 \arrow[r, "\star", "1,2"'] \arrow[d, dotted, "1,2"', "\star"] \arrow[ddr, phantom, "HR(t'_1)."] & t' \arrow[dd, dotted, "1,2"', "\star"] \\
t''_1 \arrow[r, dotted, "\star", "1,2"'] \arrow[d, "1,2"', "\star"] \arrow[dr, phantom, "HR(t''_1)."] & u_0 \arrow[d, dotted, "1,2"', "\star"] & \\
t'' \arrow[r, dotted, "\star", "1,2"'] & u_1 \arrow[r, dotted, "\star", "1,2"'] & u
\end{tikzcd}$$

FIG. 5.1 – Diagramme de confluence du lemme 5.21.ι.

Lemme 5.21.ι

Si t est un $\Lambda\mu^\iota$-terme et t', t'' sont tels que $t \rightarrow^\star_{1,2} t'$ et $t \rightarrow^\star_{1,2} t''$, alors il existe un $\Lambda\mu^\iota$-terme u tel que $t' \rightarrow^\star_{1,2} u$ et $t'' \rightarrow^\star_{1,2} u$.

Démonstration : On commence par remarquer que t' et t'' sont des $\Lambda\mu^\iota$-termes, tout comme tous les termes apparaissant dans les réductions $t \longrightarrow^\star_{1,2} t'$ et $t \longrightarrow^\star_{1,2} t''$.

On montre le résultat par induction sur la longueur d'une plus longue dérivation de $\boldsymbol{\beta}^{var}\mathit{fst}_\iota$ à partir de t, $lg_\iota(t)$ (ce qui existe grâce à la normalisation forte de $\boldsymbol{\beta}^{var}\mathit{fst}_\iota$) :

1. Si $lg_\iota(t) = 0$, alors le résultat est trivialement vrai puisqu'il n'y a pas de réduction et donc on a le résultat par réflexivité.
2. Supposons le résultat vrai pour tous les termes t tels que $lg_\iota(t) \leq n$ pour un certain entier n. On montre que si t est tel que $lg_\iota(t) = n+1$, alors le résultat est également vrai pour t. En effet, soient t' et t'' tels que $t \longrightarrow^\star_{1,2} t'$ et $t \longrightarrow^\star_{1,2} t''$. Si $t = t'$ ou $t = t''$, le résultat est trivial. Si ce n'est pas le cas, alors on a $t \longrightarrow^\star_{1,2} t'_1 \longrightarrow^\star_{1,2} t'$ et $t \longrightarrow_{1,2} t''_2 \longrightarrow^\star_{1,2} t''$.
   On raisonne par cas sur $t \longrightarrow_{1,2} t'_1$ et $t \longrightarrow_{1,2} t''_1$ et on montre que dans tous les cas, il existe u_0 tel que $t'_1 \longrightarrow^\star_{1,2} u_0$ et $t''_1 \longrightarrow^\star_{1,2} u_0$. Le seul cas qui mérite d'être détaillé est celui de la paire critique entre β_S et fst_ι pour lequel on obtient l'existence de u_0 grâce au lemme 5.14.ι.
   On a $lg_\iota(t'_1), lg_\iota(t''_1) \leq n$ et l'on peut donc appliquer l'hypothèse d'induction à t''_1 d'où l'on conclut à l'existence de u_1 tel que $t'' \longrightarrow^\star_{1,2} u_1$, $u_0 \longrightarrow^\star_{1,2} u_1$. On a donc $t'_1 \longrightarrow^\star_{1,2} u_0 \longrightarrow^\star_{1,2} u_1$ ce qui nous permet d'appliquer l'hypothèse d'induction à t'_1 et d'en déduire l'existence de u tel que $u_1 \longrightarrow^\star_{1,2} u$ et $t' \longrightarrow^\star_{1,2} u$.
   Finalement, on a $t' \longrightarrow^\star_{1,2} u$ et $t'' \longrightarrow^\star_{1,2} u_1 \longrightarrow^\star_{1,2} u$ et la situation est résumée dans la figure 5.1.

On en conclut donc que pour tous $\Lambda\mu^\iota$-termes, t, t', t'' tels que $t \longrightarrow^\star_{1,2} t'$ et $t \longrightarrow^\star_{1,2} t''$, il existe un $\Lambda\mu^\iota$-terme u tel que $t' \longrightarrow^\star_{1,2} u$ et $t'' \longrightarrow^\star_{1,2} u$.

□

On déduit du lemme 5.21.ι la confluence de la $\boldsymbol{\beta}^{var}\mathit{fst}$-réduction :

Lemme 5.21 (*La $\boldsymbol{\beta}^{var}\mathit{fst}$-réduction est confluente.*)

Soient t, t', t'' des $\Lambda\mu$-termes.
Si $t \longrightarrow^\star_{\boldsymbol{\beta}^{var}\mathit{fst}} t', t''$, alors il existe u tel que $t', t'' \longrightarrow^\star_{\boldsymbol{\beta}^{var}\mathit{fst}} u$.

Démonstration : Soit $(-)^{\iota}$ une indexation pour le terme t qui soit compatible avec t' et t'' : $(t)^{\iota} \longrightarrow^{\star}_{\boldsymbol{\beta}^{var}fst_{\iota}} (t')^{\iota}, (t'')^{\iota}$. Par le lemme 5.21.$\iota$, il existe un $\Lambda\mu$-terme indexé v tel que $(t')^{\iota}, (t'')^{\iota} \longrightarrow^{\star}_{\boldsymbol{\beta}^{var}fst_{\iota}} v$.

Soit u le $\Lambda\mu$-terme obtenu en effaçant les indices de v, on a $t', t'' \longrightarrow^{\star}_{\boldsymbol{\beta}^{var}fst} u$.

□

On en déduit la commutation de $\boldsymbol{\beta}^{var}$*fst* et $\boldsymbol{\beta}$*fst* et finalement la confluence de $\boldsymbol{\beta}$*fst* :

Lemme 5.22

$\boldsymbol{\beta}^{var}$fst et $\boldsymbol{\beta}$fst commutent.

Démonstration : On sait que la réduction $\boldsymbol{\beta}^{var}$*fst* commute avec β_T*fst* (lemme 5.16) et avec $\boldsymbol{\beta}^{var}$*fst* (lemme 5.21). Par le lemme de commutation de l'union de deux réductions, on obtient que $\boldsymbol{\beta}^{var}$*fst* et $\boldsymbol{\beta}$*fst* commutent.

□

Proposition 5.23

La $\boldsymbol{\beta}$fst-réduction est confluente.

Démonstration : L'union des réductions $\beta_T^{var}\beta_S$*fst* et β_T*fst* est bien la réduction $\boldsymbol{\beta}$*fst*, d'où la commutation de $\boldsymbol{\beta}$*fst* avec $\boldsymbol{\beta}$*fst*, c'est-à-dire la confluence de $\boldsymbol{\beta}$*fst*.

□

5.3.4 Commutation de $\boldsymbol{\eta}$ et $\boldsymbol{\beta}$*fst*

Proposition 5.24

La $\boldsymbol{\eta}$-réduction est confluente.

Démonstration : La $\boldsymbol{\eta}$-réduction est en fait fortement confluente.

□

Proposition 5.25

La $\boldsymbol{\eta}$-réduction commute avec la $\boldsymbol{\beta}$fst-réduction.

Démonstration : La proposition se démontre en prouvant la commutation de la $\boldsymbol{\eta}$-réduction avec, respectivement, les réductions β_S, β_T et *fst* :

- *β_S et $\boldsymbol{\eta}$ commutent.* β_S et η_T n'ont pas de paire critique ce qui assure la commutation. Concernant β_S et η_S, on a une paire critique qui commute trivialement : $(t)\beta \longleftarrow_{\eta_S} (\mu\alpha.(t)\alpha)\beta \longrightarrow_{\beta_S} (t)\beta$. On a donc la commutation en 0 ou 1 étape selon que les rédex se chevauchent ou non.
- *β_T et $\boldsymbol{\eta}$ commutent.* β_T et η_S n'ont pas de paire critique ; β_T et η_T commutent comme en λ-calcul.
- *fst et $\boldsymbol{\eta}$ commutent.* *fst* et η_T n'ont pas de paire critique et on peut faire commuter la paire critique entre *fst* et η_S :

$$\begin{tikzcd}
\mu\alpha.(t)\alpha \arrow[rr, "\eta_S"'] \arrow[d, "fst"'] & & t \arrow[d, dotted, "="'] \\
\lambda x.\mu\alpha.(t)x\alpha \arrow[r, dotted, "\eta_S"'] & \lambda x.(t)x \arrow[r, dotted, "\eta_T"'] & t
\end{tikzcd}$$

ce qui assure la commutation de *fst* et $\boldsymbol{\eta}$.

Finalement, on a la commutation de $\boldsymbol{\eta}$ avec $\boldsymbol{\beta}$*fst* grâce au lemme 5.9.

□

5.3.5 Conclusion de la preuve de confluence

On obtient finalement une preuve de la confluence de $\Lambda\mu$ sur les termes μ-clos :

Démonstration du théorème 5.12 : On a donc les trois propositions suivantes :

(i) la confluence de βfst (proposition 5.23),

(ii) la confluence de η (proposition 5.24) et

(iii) la commutation de ces deux systèmes de réductions (proposition 5.25).

Ces trois propositions nous permettent donc d'appliquer le lemme de Hindley-Rosen (lemme 5.8, page 106) et d'obtenir ainsi la confluence de la $\beta\eta fst$-réduction.

□

5.4. Conséquences de la confluence de $\Lambda\mu$

5.4.1 Le $\Lambda\mu$-calcul est Church-Rosser

L'équivalence entre la propriété de confluence et la propriété de Church-Rosser (définition 5.3) est d'habitude une évidence. Cependant, le fait que la confluence du $\Lambda\mu$-calcul ne tienne que pour les termes μ-clos complique la situation : en effet, une séquence d'équivalences entre deux $\Lambda\mu$-termes peut faire intervenir des termes ayant des variables de stream libres, empêchant d'appliquer le théorème de confluence pour trouver un réduit commun.

On montre ici que le $\Lambda\mu$-calcul est effectivement Church-Rosser.

Théorème 5.26 (*Le $\Lambda\mu$-calcul est Church-Rosser*)

Si t, u sont deux $\Lambda\mu$-termes μ-clos, et si $t =_{\Lambda\mu} u$ alors il existe $v \in \Sigma_{\Lambda\mu}$ tel que $t, u \longrightarrow^{\star}_{\Lambda\mu} v$.

Démonstration : Soient t et u deux $\Lambda\mu$-termes clos et soient $(t_i)_{0\leq i\leq n}$ et $(u_i)_{0\leq i\leq n}$ tels que $t = t_0$, $u = u_n$ et pour $0 \leq i \leq n$, $t_i \longrightarrow^{\star}_{\Lambda\mu} u_i$ et pour $1 \leq i \leq n$, $t_{i+1} \longrightarrow^{\star}_{\Lambda\mu} u_i$.

On raisonne comme suit : soit $\alpha_1, \dots, \alpha_n$ les variables de stream qui apparaissent libres dans les t_i, et soit $\mathcal{C}[] = \mu\alpha_1 \dots \alpha_n.[]$ On pose alors $t' = \mathcal{C}[t]$ et $u' = \mathcal{C}[u]$, ainsi que $t'_i = \mathcal{C}[t_i]$ et $u'_i = \mathcal{C}[u_i]$ pour $0 \leq i \leq n$.

On a évidemment $t' =_{\Lambda\mu} u'$ et les t'_i et u'_i sont tous des termes μ-clos ce qui nous permet d'appliquer la confluence.

Une induction immédiate nous assure alors que pour tout $0 \leq i \leq n$, il existe v'_i tel que $u'_0 \longrightarrow^{\star}_{\Lambda\mu} v'_i \longleftarrow^{\star}_{\Lambda\mu} u'_i$. On en déduit donc l'existence de v' tel que $t' \longrightarrow^{\star}_{\Lambda\mu} v' \longleftarrow^{\star}_{\Lambda\mu} u'$ (il s'agit en fait de v'_n), c'est-à-dire tel que :

$$\mathcal{C}[t] \longrightarrow^{\star}_{\Lambda\mu} v' \longleftarrow^{\star}_{\Lambda\mu} \mathcal{C}[u]$$

La relation précédente n'entraîne pas directement l'existence de v tel que $t \longrightarrow^{\star}_{\Lambda\mu} v \longleftarrow^{\star}_{\Lambda\mu} u$. Nous avons besoin d'un dernier argument. Soient ρ_1 et ρ_2 les réductions de t' à v' et de u' à v' respectivement. Puisque t et u sont μ-clos, les variables $\alpha_1, \dots, \alpha_n$ qui sont abstraites dans le contexte $\mathcal{C}[]$ n'ont aucune occurrence ni dans t ni dans u.

On en déduit donc que tous termes v_1 et v_2 apparaissant dans ρ_1 et ρ_2 respectivement sont de la forme $\mathcal{C}_1[t_1]$ et $\mathcal{C}_2[u_2]$ avec $t \longrightarrow^{\star}_{\Lambda\mu} t_1$ et $u \longrightarrow^{\star}_{\Lambda\mu} u_2$, et les contextes $\mathcal{C}_1[]$ et $\mathcal{C}_2[]$ sont obtenus en ajoutant des λ-abstractions devant certaines μ-abstractions, résultant des *fst*-réductions appliquées dans ρ_1 et ρ_2.

On a donc $\mathcal{C}[t] \longrightarrow^{\star}_{\Lambda\mu} \mathcal{C}_t[t''] = v' = \mathcal{C}_u[u''] \longleftarrow^{\star}_{\Lambda\mu} \mathcal{C}[u]$, avec $t \longrightarrow^{\star}_{\Lambda\mu} t''$ et $u \longrightarrow^{\star}_{\Lambda\mu} u''$. On conclut en constatant que $\mathcal{C}_t = \mathcal{C}_u$ car on ne change jamais le nombre de μ-abstractions dans ces préfixes or les deux termes $\mathcal{C}_t[t'']$ et $\mathcal{C}_u[u'']$ étant égaux, ils ont les même préfixes : $t'' = v = u''$

On en conclut donc que $t \longrightarrow^\star_{\Lambda\mu} v \longleftarrow^\star_{\Lambda\mu} u$.

□

5.4.2 $\Lambda\mu$-calcul, confluence et formes normales

Du fait de la règle *fst* qui est une règle d'expansion, le $\Lambda\mu$-calcul dispose de très peu de formes normales : seuls les λ-termes peuvent être des formes normales du $\Lambda\mu$-calcul.

L'une des conséquences importantes de la confluence en λ-calcul est qu'elle assure l'unicité de la forme normale. On démontre dans cette section un analogue de cette propriété en $\Lambda\mu$-calcul : les formes normales canoniques d'un terme μ-clos t sont toutes dans la même classe de *fst*-équivalence.

Proposition 5.27

Soient t et u deux $\Lambda\mu$-termes μ-clos en forme normale canonique. Si $t =_{\Lambda\mu} u$, alors $t =_{fst} u$.

Démonstration : Soient t et u deux $\Lambda\mu$-termes μ-clos en forme normale canonique tels que $t =_{\Lambda\mu} u$. Par le théorème 5.26, on sait qu'il existe v tel que $t, u \longrightarrow^\star_{\Lambda\mu} v$.

Les seuls $\Lambda\mu$-rédex de t et u sont des *fst*-rédex et leur réduction ne crée autre chose que d'autres *fst*-rédex. On en déduit donc que $t, u \longrightarrow^\star_{fst} v$ et, finalement, que $t =_{fst} u$.

□

Dans l'interprétation de streams, cela signifie que des formes normales canoniques équivalentes ne diffèrent que par le niveau d'instanciation de leurs streams.

5.4.3 Confluence du $\lambda\mu$-calcul de Parigot avec extensionalité

On commence par remarquer que $\Sigma_{\lambda\mu}$ est stable pour les réductions de $\Lambda\mu$:

Lemme 5.28 (*Stabilité des $\lambda\mu$-termes par $\Lambda\mu$-réduction*)

Soient $t, u \in \Lambda\mu$ tels que $t \longrightarrow^\star_{\Lambda\mu} u$. Si $t \in \Sigma_{\lambda\mu}$, alors $u \in \Sigma_{\lambda\mu}$ et $t \longrightarrow^\star_{\lambda\mu} u$.

Démonstration : Évident puisque l'image d'un $\lambda\mu$-terme par une règle de $\Lambda\mu$ est un $\lambda\mu$-terme et que l'occurrence de la règle en question est une instance d'une règle de $\lambda\mu\eta$ dans ce cas.

□

Théorème 5.29

Le $\lambda\mu\eta$-calcul est confluent sur les termes μ-clos.

Démonstration : Il s'agit d'un corollaire immédiat du théorème 5.12. En effet, si $t, t', t'' \in \Sigma_{\lambda\mu}$ sans μ-variable libre sont tels que $t \longrightarrow^\star_{\lambda\mu\eta} t', t''$, alors on sait que $t \longrightarrow^\star_{\Lambda\mu} t', t''$ (en remplaçant les étapes de μ-réduction par une séquence $fst; \beta_T$) alors par la confluence du $\Lambda\mu$-calcul, on sait qu'il existe $u \in \Sigma_{\Lambda\mu}$ tel que $t', t'' \longrightarrow^\star_{\Lambda\mu} u$.

On en déduit, par le lemme 5.28, que $u \in \Sigma_{\lambda\mu}$, ce qui assure que $t', t'' \longrightarrow^\star_{\lambda\mu\eta} u$ et que le $\lambda\mu\eta$-calcul est confluent sur les termes μ-clos.

□

Comparaison avec la preuve de Py. Walter Py a prouvé dans sa thèse [Py98] la confluence du $\lambda\mu\eta$-calcul. Notre preuve est en partie inspirée de la preuve de Py (chapitre 4) dans le sens où nous suivons le schéma général de la démonstration. Pourtant, notre démonstration diffère en plusieurs points qui font par ailleurs que notre technique est plus simple :

- le $\Lambda\mu$-calcul ne contient pas la μ-réduction, ce qui diminue le nombre de paires critiques à considérer et de cas à traiter ;
- On n'a pas besoin de parler des ρ-réductions bloquées qui nécessitent tout un développement complexe dans la preuve de Py (pour traiter le cas de ρ-redex «bloqués» par une *fst*-réduction) et, en particulier, une preuve auxiliaire par réduction parallèle (sections 4.3.3 et 4.3.4 de la thèse de Py).
- Au contraire, notre approche avec les $\Lambda\mu^{\iota}$-termes utilise pleinement le fait que nous n'utilisons que la β_T^{var}-réduction dans cette partie de la preuve ; l'analyse des liens entre *fst*-réduction et β_S-réduction en est facilitée.

5.5. Résumé des résultats du chapitre

Dans ce chapitre, nous avons montré la confluence du $\Lambda\mu$-calcul, propriété essentielle pour un calcul de ce type et tout particulièrement pour donner un sens à la propriété de séparation ; la propriété de séparation n'a en effet pas beaucoup de sens pour un calcul non confluent. Notre résultat a pour corollaire immédiat la confluence du $\lambda\mu$-calcul de Parigot avec η-réduction. La preuve obtenue est sensiblement plus courte que la preuves qui était connue jusque là pour le $\lambda\mu$-calcul de Parigot avec η-réduction.

Nous avons également prouvé la confluence des sous-systèmes suivants du $\Lambda\mu$-calcul :
- $\boldsymbol{\eta}$ (c'est-à-dire $\eta_T + \eta_S$) ;
- $\boldsymbol{\beta}$ (c'est-à-dire $\beta_T + \beta_S$) ;
- $\boldsymbol{\beta}$***fst*** ;
- $\beta_T\boldsymbol{\eta}$;
- $\beta_S\boldsymbol{\eta}$.

Nous avons également introduit un sous-calcul le $\Lambda\mu^{\iota}$-calcul qui s'est révélé crucial pour compléter la preuve de confluence de $\boldsymbol{\beta}$***fst***.

Le résultat de confluence pour $\Lambda\mu$ n'est valable que pour les termes μ-clos (on a le même phénomène pour $\lambda\mu\eta$) et la confluence n'est en effet pas valable lorsqu'on a des variables de stream libres apparaissant de manière non-contrainte dans le terme. Cette restriction fait que certains résultats qui sont d'habitude conséquences immédiates de la confluence (comme la propriété de Church-Rosser), doivent être redémontrés, bien que ces vérifications ne posent pas de problème majeur.

Lorsque nous développerons les réseaux SANE au chapitre 7, nous verrons que l'on peut se passer de la restriction de μ-clôture dans ce système.

Chapitre 6

$\Lambda\mu$-calcul simplement typé

Résumé:

Nous étudions dans ce chapitre le typage du $\Lambda\mu$-calcul. Nous commençons par proposer un système de types calqué sur le typage en logique classique du $\lambda\mu$-calcul dont on discute certaines propriétés. Considérant la forme des termes typables dans le premier système nous proposons ensuite un système de types étendu, $\Lambda_{\mathcal{S}}$, qui permet de typer plus de $\Lambda\mu$-termes. $\Lambda_{\mathcal{S}}$ permet en particulier de typer des formes de termes qui étaient utilisées dans la preuve de séparation du chapitre 4 et qui n'étaient pas typables dans le système classique. Ce système est original en cela qu'on considère une relation d'équivalence sur la grammaire de nos types, correspondant à une propriété d'associativité des constructeurs de type et qui exprime le fait qu'un terme puisse être à la fois appliqué à un terme et à une variable de stream. Nous étudierons également les liens entre $\Lambda_{\mathcal{S}}$ et une approche développée indépendamment par Herbelin et Ghilezan [HG08] qui n'utilise pas l'associativité. Nous démontrons ensuite la préservation du type et la normalisation forte du calcul typé et nous concluons avec quelques remarques sur un système de typage du second ordre.

Références : Une partie des résultats de ce chapitre ont été publiés dans l'article *On the relations between the syntactic theories of $\lambda\mu$-calculi* [Sau08a] et avaient auparavant été présentés dans *Typing streams in the $\Lambda\mu$-calculus : extended abstract* [Sau07].

Sommaire

6.1. Introduction

Le $\Lambda\mu$-calcul a été introduit comme une extension du $\lambda\mu$-calcul non-typé de Parigot de manière à retrouver la séparation. Nous avons montré dans les précédents chapitres que le $\Lambda\mu$-calcul non-typé satisfaisait de bonnes propriétés (séparation, confluence). Nous allons maintenant étudier les questions de typage pour le $\Lambda\mu$-calcul qui permettent d'obtenir des propriétés plus fines sur le sous-ensemble des termes typables, propriétés qu'il serait inenvisageable d'obtenir du calcul tout entier.

Avoir un système de types pour le $\Lambda\mu$-calcul est d'autant plus important que le $\lambda\mu$-calcul a été initialement introduit par Parigot pour étendre la correspondance de Curry-Howard à la logique classique. Le $\lambda\mu$-calcul a donc été conçu comme un langage typé, même s'il a, dès le début (dans l'article de Parigot [Par92] en particulier), été étudié également dans sa formulation non-typée.

Dans tous les cas, il est certain que l'intérêt porté au $\lambda\mu$-calcul provient à la fois

- de sa simplicité ;
- de ses bonnes propriétés étendant celles du λ-calcul ;
- de sa capacité à coder en style direct des opérateurs de contrôle des langages fonctionnels qu'il est peu commode de capturer en λ-calcul et
- de ses liens avec la logique classique.

Pour ces deux raisons (utilité générale du typage et tradition historique du $\lambda\mu$-calcul), il nous semble essentiel de mener une étude typée du $\Lambda\mu$-calcul.

L'étude des types pour $\Lambda\mu$ est également importante dans la perspective de la propriété de séparation : d'une part, nous avons vu que l'on peut étendre des résultats de séparation au λ-calcul typé qui fournissent dans ce cas un résultat de séparation pour la déduction naturelle intuitionniste ; d'autre part, nous avons vu que ce qui causait l'échec de la séparation en $\lambda\mu$-calcul pouvait être inteprété comme une contrainte de typage forgée dans la syntaxe du calcul non-typé : le fait de forcer une occurrence simultanée de la μ-abstraction et de la construction de nommage $[\alpha]$[1] assure que l'on passe bien par le type $\perp$ en changeant de type, ce qui fait partie de la validité du raisonnement classique, alors même que les constructions syntaxiques et les réductions du $\lambda\mu$-calcul (non-typé) ne nécessitent aucunement que cette contrainte soit imposée dans le cas pur.

Nous nous limiterons principalement dans ce chapitre aux types simples, n'évoquant la question du polymorphisme que très brièvement et à la fin du chapitre.

Nous commencerons par étudier une possibilité naïve pour typer le $\Lambda\mu$-calcul : le typer grâce au système proposé par Parigot ou plus exactement grâce au système proposé par Ph. de Groote [dG98] qui est une version de la déduction naturelle classique avec $\perp$ explicite.

Après avoir expliqué en quoi cette solution n'est pas satisfaisante du point de vue de la séparation, nous élaborerons un système de typage pour le $\Lambda\mu$-calcul à partir de considérations sur la séparation et l'interprétation de streams proposée au chapitre 4.

Ce système, $\Lambda_{\mathcal{S}}$, sera justifié informellement dans un premier temps. $\Lambda_{\mathcal{S}}$ est un système relativement complexe car il est construit autour d'une équivalence sur les types dont l'interprétation sera expliquée en détail en section 6.3.

Nous comparerons ensuite le système $\Lambda_{\mathcal{S}}$ à un système de typage proposé par Herbelin et Ghilezan pour le $\lambda\mu\widehat{\mathsf{tp}}$-calcul en appel par nom, une variante du $\lambda\mu$-calcul avec une variable de continuation liée dynamiquement [HG08].

[1] Ou plutôt, en suivant la syntaxe que nous utilisons dans cette thèse, que le sous-terme t de $\mu\alpha.t$ doit être de la forme $(u)\beta$.

Nous démontrerons ensuite que $\Lambda_{\mathcal{S}}$ satisfait la préservation du typage et la normalisation forte, comparerons la typabilité dans $\Lambda_{\mathcal{S}}$ et dans le système original de Parigot et introduirons une notion de formes normales canoniques *fst*η-longues. Nous terminerons ce chapitre en proposant une extension au second ordre de $\Lambda_{\mathcal{S}}$.

6.2. Typage classique du $\Lambda\mu$-calcul

Pour typer $\Lambda\mu$, on peut naturellement penser à utiliser un système de typage classique comme pour le $\lambda\mu$-calcul tel que présenté en définition 2.41 :

Définition 6.1 (*Jugements de typage pour le $\Lambda\mu$-calcul*)

Les jugements de typage pour le $\Lambda\mu$-calcul sont de la forme :

$$x_1 : A_1, \ldots, x_m : A_m \vdash_{\Lambda\mu} t \ : A | \alpha_1 : B_1, \ldots, \alpha_n : B_n$$

où $A, A_1, \ldots, A_m, B_1, \ldots, B_n$ sont des types construits à partir de variables de types, de la constante $\perp$ et du connecteur $\rightarrow$.

Définition 6.2 (*Système de typage pour le $\Lambda\mu$-calcul*)

On considère les règles de typage suivantes pour les $\Lambda\mu$-termes :

- *typage des variables :*

$$\frac{}{\Gamma, x : A \vdash_{\Lambda\mu} x \ : A|\Delta}\ Var$$

- *typage des λ-abstractions :*

$$\frac{\Gamma, x : A \vdash_{\Lambda\mu} t \ : B|\Delta}{\Gamma \vdash_{\Lambda\mu} \lambda x^A.t \ : A \rightarrow B|\Delta}\ \lambda\text{-}Abs$$

- *typage des λ-applications :*

$$\frac{\Gamma \vdash_{\Lambda\mu} t \ : A \rightarrow B|\Delta \qquad \Gamma \vdash_{\Lambda\mu} u \ : A|\Delta}{\Gamma \vdash_{\Lambda\mu} (t)u \ : B|\Delta}\ \lambda\text{-}App$$

- *typage des μ-abstractions :*

$$\frac{\Gamma \vdash_{\Lambda\mu} t \ : \perp|\Delta, \alpha : A}{\Gamma \vdash_{\Lambda\mu} \mu\alpha^A.t \ : A|\Delta}\ \mu\text{-}Abs$$

- *typage des μ-applications :*

$$\frac{\Gamma \vdash_{\Lambda\mu} t \ : A|\Delta}{\Gamma \vdash_{\Lambda\mu} (t)\alpha \ : \perp|\Delta, \alpha : A}\ \mu\text{-}App$$

C'est par exemple ce que fait de Groote [dG98] et cela correspond au système présenté en figure 3.3. Dans ce cas, on restreint évidemment l'usage de la règle *fst* dans le cas typé aux seuls termes $\mu\alpha^{A\rightarrow B}.t$:

$$\mu\alpha^{A\rightarrow B}.t \longrightarrow_{fst} \lambda x^A.\mu\alpha^B.t\,\{(u)x\alpha/(u)\alpha\}$$

De Groote [dG98] remarque qu'une preuve de $(\mu\alpha^{A\rightarrow B}.t)u^A \ : \ B$ contient un β-redex caché et que le rôle de la μ-réduction est de réduire ce cas [2]. La règle *fst* peut être vue comme

[2] De Groote [dG98] : *«In order to turn hidden redexes into actual ones, some reduction rule other than*

faisant la moitié de ce travail, laissant le reste à la β_T-réduction.

De bonnes propriétés mais de gros inconvénients. Un tel système présente de bonnes propriétés : préservation du type, normalisation forte, etc... Pourtant, nous considérons que cette approche n'est pas satisfaisante. En effet, typer un langage, c'est faire le compromis de rejeter un certain nombre de termes, de programmes (ceux qui ne seront pas typables dans le système) pour pouvoir énoncer des propriétés intéressantes pour les termes qui restent, les termes qui sont validés par le typage. Or, dans la perspective de la séparation, qui est celle qui nous a conduits à concevoir le système $\Lambda\mu$, le typage que nous venons de présenter à la définition 6.2 n'est pas satisfaisant.

Nous considérons en effet que le système précédent rejette certains termes que nous souhaiterions pouvoir typer car ils nous semblent être représentatifs de comportements calculatoires ***sains*** (ils ne causent pas de perte de la terminaison par exemple) et ***utiles*** (il s'agit typiquement des termes construits au cours d'un processus de Böhm out, cf chapitre 4).

Plus généralement, on remarquera que si on se limite au système de typage ci-dessus, on se retrouve avec un langage typé qui n'est pas beaucoup plus riche que le $\lambda\mu$-calcul original (on rajoute essentiellement des termes qui sont des β_T-expansions de $\lambda\mu$-termes typés, ou sous-termes de telles expansions, ainsi que des termes de la forme : $\mu\alpha_1^A, \alpha_2^\perp, \ldots, \alpha_n^\perp.t^\perp$)

6.3. Séparation et typabilité : vers un typage des streams

Restriction de *fst*. Revenons un instant sur la restriction de la règle *fst* proposée au paragraphe précédent. Nous avons introduit cette restriction de manière à avoir la préservation du type qui, sans cette restriction, était évidemment perdue. En fait, ce n'était pas seulement la préservation du type, mais la préservation de la typabilité elle-même qui était en jeu avant la restriction de *fst*. En effet, la règle *fst* générale ne préserve même pas la typabilité comme illustré ci-dessous :

$$\lambda x.\mu\alpha.\mu\beta.(x)\alpha \longrightarrow_{fst} \lambda x.\mu\alpha.\lambda y.\mu\beta.(x)\alpha$$

Le terme de gauche est typable de type $A \to A$ tandis que le terme de droite n'est pas typable (puisque $\lambda y.\mu\beta.(x)\alpha$ devrait recevoir le type $\perp$, ce qui n'est pas possible). Il ne s'agit pas d'un problème propre à la règle *fst* (non plus qu'à la règle ν de $\lambda\mu\eta$) : un tel phénomène se produit déjà avec l'η-expansion à partir de laquelle la règle *fst* est construite. En λ-calcul, η_{exp} préserve la typabilité mais pas les types, mais dès que l'on introduit une constante de type dans notre système de typage, la typabilité n'est plus préservée comme le montre l'exemple :

$$\lambda x.x{+_{\texttt{int}}}x \longrightarrow_{\eta_{exp}} \lambda x.\lambda y.(x{+_{\texttt{int}}}x)y$$

Cette remarque a des conséquences importantes sur la question de la typabilité de la séparation en $\Lambda\mu$-calcul : il est immédiat de noter que les paires paramétrées $\langle u, v\rangle_k$ de la définition 4.15 ne sont pas typables dès que $k > 0$. La preuve que nous avons donnée de la Séparation est donc hautement non-typable, ce qui est habituel déjà en λ-calcul, mais il semble que le problème soit plus fondamental que cela : la Séparation en $\Lambda\mu$-calcul (et donc en $\lambda\mu$-calcul) semble reposer sur des constructions qui n'ont pas d'espoir d'être typables dans le système de types classique considéré par Parigot.

β is needed. This new reduction rule, called μ by Parigot, allows the above proof scheme to be reduced» désignant par le *«above proof scheme»* le μ-redex typé.

Trop peu de termes typés. Plus généralement, aucun terme de la forme $\mu\alpha.\lambda x.t$ ne serait typable alors qu'il s'agit typiquement d'une forme de terme qui peut avoir une forme normale canonique.

La contrainte de typage impose que si deux streams se succèdent, la seconde doit être de type $\perp$, c'est-à-dire intuitivement qu'elle ne peut contribuer en rien de significatif au calcul.

En effet, le système de typage introduit à la définition 6.2 interdit précisément de tels termes : $\lambda x.t$ est un terme λ-abstrait et doit donc être d'un type $\rightarrow$ tandis que le fait qu'on lui applique une μ-abstraction avec la variable de stream α le contraint à être de type $\perp$ (lorsque ce type est explicite ou, dans la présentation de Parigot, à être typé par un séquent sans type distingué). Ces deux contraintes sont incompatibles (voir la règle μAbs de la définition 6.2).

Ce fait n'est pas étonnant : nous avions remarqué à la fin de la section 3.4 du chapitre sur la séparation que la non-séparation de $\lambda\mu\eta$ venait d'une structure de terme trop contrainte (avec la hiérarchisation des termes de $\Sigma_{\lambda\mu}$ en deux niveaux) qui pouvait s'interpréter comme une contrainte de typage. Dans la définition 6.2, nous venons de réintroduire explicitement cette contrainte de typage et nous perdons donc à nouveau ces termes.

Des streams citoyens de première classe dans un contexte typé. Le mécanisme de stream qui était utilisé pour le calcul non-typé de manière à obtenir la séparation se trouve donc désactivé quand le typage de la logique classique est réintroduit ; on perd donc le bénéfice de la syntaxe étendue du $\Lambda\mu$-calcul. Nous allons donc chercher une variante de ce système de types qui reflèterait dans les types la construction de stream : nous aurons besoin de pouvoir considérer des types de streams et les distinguer des types de termes (avec des variables de type spécifiques, éventuellement des constantes de type spécifiques et un constructeur de type de stream, que nous noterons dans un premier temps ::, tel que si $\mathcal{T}$ est un type de terme et $\mathcal{S}$ est un type de stream, $\mathcal{T} :: \mathcal{S}$ sera un type de stream).

En outre, puisque μ est vu comme une abstraction de stream, on pourrait penser à donner un type fonctionnel pour les streams : si le terme t est de type $\mathcal{T}$ quand la stream α est du type de stream $\mathcal{S}$, alors $\mu\alpha.t$ devrait avoir le type d'une stream-fonction de $\mathcal{S}$ dans $\mathcal{T}$ (ce que nous écrirons $\mathcal{S} \Rightarrow \mathcal{T}$). On peut donc penser aux règles de typage suivantes pour les termes μ-astraits du $\Lambda\mu$-calcul :

$$\frac{\Gamma \vdash t\ : \mathcal{T}|\Delta, \alpha : \mathcal{S}}{\Gamma \vdash \mu\alpha^{\mathcal{S}}.t\ : \mathcal{S} \Rightarrow \mathcal{T}|\Delta}\ Abs_{\mathcal{S}} \qquad \frac{\Gamma \vdash t\ : \mathcal{S} \Rightarrow \mathcal{T}|\Delta, \alpha : \mathcal{S}}{\Gamma \vdash (t)\alpha\ : \mathcal{T}|\Delta, \alpha : \mathcal{S}}\ App_{\mathcal{S}}$$

Une incompatibilité de typage. La règle *fst* complique la définition d'un système de types pour $\Lambda\mu$ qui prendrait en compte les streams : alors que $\mu\alpha^{\mathcal{S}}.t$ devrait être d'un type de stream, disons $\mathcal{S} \Rightarrow \mathcal{T}$, le terme obtenu à partir de $\mu\alpha.t$ en appliquant une étape de règle *fst* (c'est-à-dire le terme $\lambda x^A.\mu\beta^{\mathcal{S}'}.t\,\{(u)x\beta/(u)\alpha\}$) devrait avoir un type fonctionnel du λ-calcul : $A \rightarrow B$ (plus précisément $A \rightarrow (\mathcal{S}' \Rightarrow \mathcal{T}')$).

En outre, les streams sont des streams de termes et leurs types devraient être connectés aux types de termes usuels, non seulement par la règle *fst*, mais également en autorisant une stream-fonction à être appliqué à un terme (par exemple sous la forme $(\mu\alpha.t)u$). Réciproquement, on pourrait souhaiter appliquer un terme λ-abstrait à une stream (par exemple $(\lambda x.t)\alpha$). Les streams et les termes ne doivent donc pas vivre dans des mondes imperméables l'un à l'autre.

Les $\Rightarrow$-types et les $\rightarrow$-types devraient donc être reliés d'une manière ou d'une autre : il nous faut maintenant trouver cette relation connectant les types de stream et les types de terme. C'est en fait la règle *fst* qui donne la solution que nous allons développer dans la suite pour établir cette connexion. On analyse donc plus en détail la règle *fst* dans le paragraphe suivant.

Une relation sur les types de stream. La *fst*-réduction a été synthétisée en $\Lambda\mu$-calcul (et auparavant déjà en $\lambda\mu\eta$-calcul par Py [Py98, DP01a][3]) comme le résultat d'une η-expansion suivie d'une μ-réduction. Dans le cas typé, la η-expansion peut avoir lieu uniquement sur des termes de type $\rightarrow$. Cette restriction adaptée au $\Lambda\mu$-calcul donne la condition selon laquelle $\mu\alpha.t$ doit être d'un type de stream de la forme $(\mathcal{T} :: \mathcal{S}) \Rightarrow \mathcal{T}'$. Après une application de *fst*, nous avons maintenant un terme $\lambda x.\mu\beta.t\,\{(u)x\beta/(u)\alpha\}$ qui devrait être de type $\mathcal{T} \rightarrow (\mathcal{S} \Rightarrow \mathcal{T}')$. Dans la suite, le constructeur de types de stream :: sera identifié au constructeur $\rightarrow$ et l'on voit ainsi que cela correspond à une règle d'associativité entre les constructeurs $\rightarrow$ et $\Rightarrow$:

$$\boxed{(\mathcal{T} \rightarrow \mathcal{S}) \Rightarrow \mathcal{T}' =_{\text{assoc}\rightrightarrows} \mathcal{T} \rightarrow (\mathcal{S} \Rightarrow \mathcal{T}')}$$

Même si nous avons motivé cette équivalence d'un point de vue opérationnel (préservation du type), elle peut toujours paraître surprenante d'un point de vue logique : nous avons commencé par prendre du recul par rapport à la logique classique en cherchant un nouveau système de types plus adapté au calcul $\Lambda\mu$ et maintenant nous introduisons en plus une relation d'équivalence sur les types qui, toute justifiée qu'elle soit opérationnellement, n'a pas de sens logique clairement défini. Est-ce bien nécessaire ? Cela a-t-il du sens ? La «logique du $\Lambda\mu$-calcul» n'est pas encore complètement comprise, mais nous motiverons tout de même dans la suite cette règle d'associativité d'un point de vue logique en section 6.4.3.

6.4. Streams simplement typées : $\Lambda_{\mathcal{S}}$

6.4.1 Pré-types et types

Nous définissons maintenant formellement le système de typage $\Lambda_{\mathcal{S}}$ pour le $\Lambda\mu$-calcul.

Définition 6.3 ($\Lambda_{\mathcal{S}}$ *pré-types*)

Les pré-types sont produits par la grammaire suivante :

Pré-types de terme :	$\mathcal{T}, A, B, \ldots$	$::= o_i \mid A \rightarrow B \mid \mathcal{S} \Rightarrow \mathcal{T}$
Pré-types de stream :	$\mathcal{S}, P, Q, \ldots$	$::= \sigma_i \mid \mathcal{T} \rightarrow \mathcal{S} \mid \bot$

o_i et σ_i sont respectivement des variables de type de terme et de stream. Le type $A_1 \rightarrow A_2 \rightarrow \cdots \rightarrow A_n \rightarrow \sigma$ est donc le type d'une stream qui contient au moins n éléments. Nous conservons la constante $\bot$ dans le calcul davantage pour des raisons de tradition que parce qu'elle est véritablement utile : le type $\bot$ peut être vu comme une variable de type de stream distinguée[4]. Elle pourra tout de même être utilisée pour désigner la stream vide. On aurait pu choisir de la retirer du système et conserver les mêmes résultats, comme nous

[3] Cette règle avait en fait été brièvement considérée en 1993 par Parigot [Par93], sans qu'elle soit considérée dans plus de détails. On se rappellera en revanche qu'elle a également été considérée plus sérieusement par Ong [Ong96], voir le chapitre 3, page 85.

[4] On peut également la voir comme une variable de type qui ne peut être substituée par d'autres types.

$$\frac{}{\Gamma, x : \mathcal{T} \vdash x \ : \mathcal{T}|\Delta}\, Var_{\mathcal{T}} \qquad \frac{\Gamma \vdash t \ : \mathcal{T}|\Delta}{\Gamma \vdash t \ : \mathcal{T}'|\Delta} \equiv_{fst} \quad (\text{pourvu que } \mathcal{T} \equiv_{fst} \mathcal{T}')$$

$$\frac{\Gamma, x : \mathcal{T} \vdash t \ : \mathcal{T}'|\Delta}{\Gamma \vdash \lambda x^{\mathcal{T}}.t \ : \mathcal{T} \to \mathcal{T}'|\Delta}\, Abs_{\mathcal{T}} \qquad \frac{\Gamma \vdash t \ : \mathcal{T} \to \mathcal{T}'|\Delta \qquad \Gamma \vdash u \ : \mathcal{T}|\Delta}{\Gamma \vdash (t)u \ : \mathcal{T}'|\Delta}\, App_{\mathcal{T}}$$

$$\frac{\Gamma \vdash t \ : \mathcal{T}|\Delta, \alpha : \mathcal{S}}{\Gamma \vdash \mu\alpha^{\mathcal{S}}.t \ : \mathcal{S} \Rightarrow \mathcal{T}|\Delta}\, Abs_{\mathcal{S}} \qquad \frac{\Gamma \vdash t \ : \mathcal{S} \Rightarrow \mathcal{T}|\Delta, \alpha : \mathcal{S}}{\Gamma \vdash (t)\alpha \ : \mathcal{T}|\Delta, \alpha : \mathcal{S}}\, App_{\mathcal{S}}$$

FIG. 6.1 – $\Lambda_{\mathcal{S}}$: un système de typage pour le $\Lambda\mu$-calcul.

le ferons dans le chapitre 7. On notera tout de même l'utilité de disposer de cette variable lorsque l'on étudiera les relations entre la $\lambda\mu$-typabilité et la $\Lambda_{\mathcal{S}}$-typabilité.

Définition 6.4 ($\equiv_{fst}$)

$\equiv_{fst}$ est une congruence sur les pré-types qui est la clôture symétrique, réflexive et transitive de la relation $\succ_{fst}$ définie par :

$$(\mathcal{T} \to \mathcal{S}) \Rightarrow \mathcal{T}' \succ_{fst} \mathcal{T} \to (\mathcal{S} \Rightarrow \mathcal{T}')$$

Les types de $\Lambda_{\mathcal{S}}$ sont toujours considérés modulo cette relation de congruence :

Définition 6.5 ($\Lambda_{\mathcal{S}}$ *types*)

Un $\Lambda_{\mathcal{S}}$-type est une classe d'équivalence pour $\equiv_{fst}$.

Le $\Lambda\mu$-calcul typé est étudié dans un système à la Church, c'est-à-dire que la syntaxe des $\Lambda\mu$-termes typés est comme suit :

Définition 6.6 ($\Lambda\mu$-*calcul typé*)

Les $\Lambda\mu$-termes typés sont donnés par la syntaxe suivante :

$$t ::= \ x \mid \lambda x^{\mathcal{T}}.t \mid (t)u \mid \mu\alpha^{\mathcal{S}}.t \mid (t)\alpha.$$

On montre en figure 6.1 le système de typage $\Lambda_{\mathcal{S}}$ pour le $\Lambda\mu$-calcul. Dans ce système de typage, on travaillera avec des pré-types et une règle de conversion explicite permettant de passer d'un pré-type à un autre pré-type qui lui est équivalent. Même si la plupart des résultats seront énoncés pour les types vus comme classes d'équivalence, il pourra arriver qu'il soit plus pratique d'énoncer un théorème en utilisant un représentant de la classe d'équivalence. Dans ce cas, on le fera sans le mentionner si cela n'introduit pas d'ambiguïté.

6.4.2 Règles de réduction typées

La règle *fst* est une règle d'expansion et doit donc être traitée avec attention si l'on souhaite avoir la préservation du type et la normalisation forte dans $\Lambda_{\mathcal{S}}$. Dans le cas typé, nous demanderons, pour pouvoir appliquer la règle *fst* à t, que le terme ait un type représenté par un pré-type de la forme $(\mathcal{T}_1 \to \mathcal{S}) \Rightarrow \mathcal{T}_2$.

Cela peut sembler surprenant et très restrictif dans le sens où cela affaiblit beaucoup la règle *fst* et les cas où elle peut être appliquée, mais en fait il n'en est rien : cette contrainte est similaire à la condition sur l'application de l'η-expansion dans le λ-calcul simplement typé et sur la *fst*-réduction dans la section 6.2, et cette restriction est nécessaire pour satisfaire la préservation du type en présence d'une règle d'expansion.

Nous imposons la même contrainte à la règle *fst* dans le cadre de $\Lambda_{\mathcal{S}}$, à l'exception du fait que notre contrainte porte maintenant sur le type de la stream qui est abstraite et non pas sur le type du terme tout entier :

Définition 6.7 (*fst*$^{\rightarrow}$*-réduction*)

La réduction fst$^{\rightarrow}$ est définie comme une restriction de la fst-réduction aux $\Lambda\mu$-termes typés comme suit :

$$\mu\alpha^{A\rightarrow\mathcal{S}}.t \longrightarrow_{fst^{\rightarrow}} \lambda x^A \mu\beta^{\mathcal{S}}.t\,\{(u)x\beta/(u)\alpha\}$$

Définition 6.8 ($\boldsymbol{\Lambda\mu^{\rightarrow}}$)

Le système de réduction $\Lambda\mu^{\rightarrow}$ est le système de réduction sur les $\Lambda\mu$-termes typés dans $\Lambda_{\mathcal{S}}$ constitué de la règle fst$^{\rightarrow}$ et du sous-système de $\longrightarrow_{\Lambda\mu}$, $\boldsymbol{\beta\eta}$.

On peut noter que *fst*$^{\rightarrow}$ est une réduction intermédiaire entre *fst* et *fst*$^{-}$:

Proposition 6.9

Soient t, u deux $\Lambda\mu$-termes typés. Les implications suivantes sont vérifiées :

$$t \longrightarrow_{fst^-} u \quad \Rightarrow \quad t \longrightarrow_{fst^{\rightarrow}} u \quad \Rightarrow \quad t \longrightarrow_{fst} u$$

Les implications réciproques ne sont pas vérifiées.

Démonstration :

- $t \longrightarrow_{fst^{\rightarrow}} u \quad \Rightarrow \quad t \longrightarrow_{fst} u$ est évidente ;
- $t \longrightarrow_{fst^-} u \quad \Rightarrow \quad t \longrightarrow_{fst^{\rightarrow}} u$ se prouve en remarquant que si $t \longrightarrow_{fst^-} u$, alors soit la μ-abstraction qui se voit appliquer la *fst*-réduction était de la forme : $(\mu\alpha^{\mathcal{S}}.t)u$, soit la variable de stream α abstraite par ce μ avait une occurrence dans un sous-terme de la forme $(\lambda x^{\mathcal{T}}.t)\alpha$. Les contraintes de typabilité impliquent que dans le premier cas, le type de $\mu\alpha^{\mathcal{S}}.t$, de la forme $\mathcal{S} \Rightarrow \mathcal{T}$ doit être $\equiv_{fst}$-équivalent à un type $\mathcal{T}' \rightarrow \mathcal{T}''$ ce qui implique que $\mathcal{S}$ est de la forme $\mathcal{T}' \rightarrow \mathcal{S}'$ et donc que *fst*$^{\rightarrow}$ est applicable. De même, dans le second cas, les contraintes de typabilité impliquent que le sous-terme $\lambda x.t$, de type $\mathcal{T} \rightarrow \mathcal{T}'$ soit $\equiv_{fst}$-équivalent à un type $\mathcal{S} \Rightarrow \mathcal{T}''$ ce qui implique que $\mathcal{S}$ soit $\equiv_{fst}$-équivalent à $\mathcal{T} \rightarrow \mathcal{S}'$ avec $\mathcal{S}' \Rightarrow \mathcal{T}'' \equiv_{fst} \mathcal{T}'$ et donc que *fst*$^{\rightarrow}$ est applicable.

□

6.4.3 De $\Rightarrow$ à $\parr$: la logique de $\Lambda_{\mathcal{S}}$

Contrairement à ce que la notation $\Rightarrow$ peut laisser penser, il n'y a pas de dualité à l'œuvre avec ce connecteur.

L'équivalence $\equiv_{fst}$: $(\mathcal{T} \rightarrow \mathcal{S}) \Rightarrow \mathcal{T}' \equiv_{fst} \mathcal{T} \rightarrow (\mathcal{S} \Rightarrow \mathcal{T}')$ devient, si on se rappelle que le connecteur $\rightarrow$ à droite de $\equiv_{fst}$ peut être interprété comme $?(\_)^{\perp} \parr \_$:

$$(\mathcal{T} \rightarrow \mathcal{S}) \Rightarrow \mathcal{T}' \equiv_{fst} ?\,\mathcal{T}^{\perp} \parr (\mathcal{S} \Rightarrow \mathcal{T}')$$

On peut alors penser à interpréter de la même manière le connecteur $\rightarrow$ à gauche de $\equiv_{fst}$ obtenant :

$$(?\,\mathcal{T}^{\perp} \parr \mathcal{S}) \Rightarrow \mathcal{T}' \equiv_{fst} ?\,\mathcal{T}^{\perp} \parr (\mathcal{S} \Rightarrow \mathcal{T}')$$

Ce qui permet de retrouver une règle d'associativité, qui est maintenant une associativité entre $\Rightarrow$ et le connecteur $\parr$. On est donc tenté d'interpréter $\Rightarrow$ comme un autre connecteur $\parr$, l'associativité ne devenant plus qu'une associativité entre deux $\parr$. La règle $Abs_{\mathcal{S}}$ est d'ailleurs bien en accord avec cette interprétation. L'équivalence $\equiv_{fst}$ est donc simplement une propriété d'associativité du $\parr$ qui est parfaitement correcte logiquement :

$$\boxed{(?\,\mathcal{T}^{\perp} \parr \mathcal{S}) \parr \mathcal{T} \equiv_{assoc} ?\,\mathcal{T}^{\perp} \parr (\mathcal{S} \parr \mathcal{T}).}$$

Nous détaillerons cette interprétation des types de $\Lambda_{\mathcal{S}}$ dans le chapitre 7 où nous présenterons une traduction du $\Lambda\mu$-calcul dans (une sorte de) réseaux de démonstration polarisés.

6.5. Relations avec le système de types de $\lambda\mu\widehat{\mathbf{tp}}$

Herbelin et Ghilezan [HG08] ont introduit récemment un calcul $\lambda\mu\widehat{\mathsf{tp}}$ en appel par nom qui étend le $\lambda\mu$-calcul de Parigot avec une variable spéciale, $\widehat{\mathsf{tp}}$, dynamiquement liée, qui permet de modéliser les continuations délimitées (le $\lambda\mu\widehat{\mathsf{tp}}$-calcul en appel par valeur a d'abord été introduit par Ariola et al. [AHS07]). En étudiant le calcul en appel par nom [HG08], ils ont établi que $\lambda\mu\widehat{\mathsf{tp}}$ est un calcul très proche du $\Lambda\mu$-calcul et ont introduit un système de typage pour ce calcul qui est lui-même très lié à $\Lambda_{\mathcal{S}}$ bien qu'introduit indépendamment.

Nous nous intéressons dans cette section aux liens entre $\Lambda\mu$ et son système de type $\Lambda_{\mathcal{S}}$ d'une part et le $\lambda\mu\widehat{\mathsf{tp}}$-calcul de Herbelin et Ghilezan d'autre part. On rappelle les résultats de Herbelin et Ghilezan à propos des liens entre $\Lambda\mu$ et $\lambda\mu\widehat{\mathsf{tp}}$ avant de discuter les différences entre les deux approches.

Définition 6.10 (*$\lambda\mu\widehat{tp}$-termes [HG08]*)

Les termes de $\lambda\mu\widehat{tp}$ ($t, u, \dots \in \Sigma_{\lambda\mu\widehat{tp}}$) sont donnés par la syntaxe suivante :

$$\begin{array}{rcl} t, u & ::= & x \mid \lambda x.\, t \mid (t)u \mid \mu q.c \\ c & ::= & [q]t \\ q & ::= & \alpha \mid \widehat{tp} \end{array}$$

Le calcul $\lambda\mu\widehat{\mathsf{tp}}$ est présenté par un ensemble de règles de réduction et une théorie équationnelle.

Définition 6.11 (*Réduction de $\lambda\mu\widehat{tp}$ [HG08]*)

Le système de réduction de $\lambda\mu\widehat{tp}$ est donné par les règles de réduction suivantes :

$$\begin{array}{lll} (\lambda x.\, t)\; u & \longrightarrow_{\beta} & t\,\{u/x\} \\ (\mu\alpha.\, c)\; t & \longrightarrow_{\mu} & \mu\alpha.c\,\{[\alpha](u)t/[\alpha]u\} \\ [\beta]\mu\alpha.c & \longrightarrow_{\mu^n_{var}} & c\,\{\beta/\alpha\} \\ \mu\widehat{tp}.[\widehat{tp}]t & \longrightarrow_{\eta^n_{\widehat{tp}}} & t \qquad \textit{même si } \widehat{tp} \textit{ apparaît libre dans } t \end{array}$$

$\Pi(\_) : \Lambda\mu \longmapsto \lambda\mu\widehat{\mathsf{tp}}$	$\Sigma(\_) : \lambda\mu\widehat{\mathsf{tp}} \mapsto \Lambda\mu$
$\begin{array}{lcl} \Pi(x) & \triangleq & x \\ \Pi(\lambda x.t) & \triangleq & \lambda x.\Pi(t) \\ \Pi((t)u) & \triangleq & (\Pi(t))\Pi(u) \\ \Pi(\mu\alpha.t) & \triangleq & \mu\alpha.[\widehat{\mathsf{tp}}]\Pi(t) \\ \Pi((t)\alpha) & \triangleq & \mu\widehat{\mathsf{tp}}.[\alpha]\Pi(t) \end{array}$	$\begin{array}{lcl} \Sigma(x) & \triangleq & x \\ \Sigma(\lambda x.t) & \triangleq & \lambda x.\Sigma(t) \\ \Sigma((t)u) & \triangleq & (\Sigma(t))\Sigma(u) \\ \Sigma(\mu\alpha.[\beta]t) & \triangleq & \mu\alpha.(\Sigma(t))\beta \\ \Sigma(\mu\alpha.[\widehat{\mathsf{tp}}]t) & \triangleq & \mu\alpha.(\Sigma(t)) \\ \Sigma(\mu\widehat{\mathsf{tp}}.[\alpha]t) & \triangleq & (\Sigma(t))\alpha \\ \Sigma(\mu\widehat{\mathsf{tp}}.[\widehat{\mathsf{tp}}]t) & \triangleq & \Sigma(t) \end{array}$

FIG. 6.2 – Traductions entre $\Lambda\mu$ et $\lambda\mu\widehat{\mathsf{tp}}$ [HG08].

Définition 6.12 (*Théorie équationnelle de $\lambda\mu\widehat{tp}$ [HG08]*)

La théorie équationnelle est donnée par les équations suivantes :

$$\begin{array}{llll}
(\lambda x.\, t)\; u & =_\beta & t\,\{u/x\} & \\
[\beta](\mu\alpha.\, c)t_1 \ldots t_k & =_{\mu_n} & c\,\{[\beta](u)t_1 \ldots t_k/[\alpha]u\} & \\
\mu\alpha.[\alpha]t & =_{\eta_\mu} & t & \textit{si } \alpha \textit{ n'est pas libre dans } t \\
\lambda x.(t)x & =_\eta & t & \textit{si } x \textit{ n'est pas libre dans } t \\
\big[\widehat{tp}\big]\mu\widehat{tp}.c & =_{\mu_{\widehat{tp}}} & c & \\
\mu\widehat{tp}.\big[\widehat{tp}\big]t & =_{\eta^n_{\widehat{tp}}} & t & \textit{même si } \widehat{tp} \textit{ apparaît libre dans } t
\end{array}$$

Les théories équationnelles de $\Lambda\mu$ et de $\lambda\mu\widehat{\mathsf{tp}}$ se correspondent via les traductions définies en figure 6.2 :

Proposition 6.13 (*Correspondance équationnelle entre $\Lambda\mu$ et $\lambda\mu\widehat{tp}$ [HG08]*)

Soient $t, u \in \Sigma_{\Lambda\mu}$, $t', u' \in \Sigma_{\lambda\mu\widehat{tp}}$. On a :
- *Si $t =_{\Lambda\mu} u$ alors $\Pi(t) =_{\lambda\mu\widehat{tp}} \Pi(u)$;*
- *Si $t =_{\lambda\mu\widehat{tp}} u$ alors $\Sigma(t) =_{\Lambda\mu} \Sigma(u)$;*
- *$\Sigma(\Pi(t))$ est syntaxiquement égal à t ;*
- *$\Pi(\Sigma(t')) =_{\mu_{\widehat{tp}}} t'$.*

On peut maintenant étudier le système de types simples de $\lambda\mu\widehat{\mathsf{tp}}$. Les jugements de typage de $\lambda\mu\widehat{\mathsf{tp}}$ sont définis ci-dessous tandis que le système de types lui-même est présenté en figure 6.3.

Définition 6.14 (*Jugements de typage de $\lambda\mu\widehat{tp}$ [HG08]*)

Les jugements de typage de $\lambda\mu\widehat{tp}$ sont donnés ci-dessous :

$$\begin{array}{rcl} A, B & ::= & o_i \mid A_\Sigma \rightarrow B \\ \Sigma & ::= & \perp \mid A \cdot \Sigma \end{array}$$

$$x_1 : A^1_{\Sigma_1}, \ldots, x_k : A^k_{\Sigma_k} \vdash_\Sigma t\; : A | \alpha_1 : B_1, \ldots, \alpha_l : B_l$$

Les jugements de typage de $\lambda\mu\widehat{\mathsf{tp}}$ ont donc une forme différente des jugements de typage de $\Lambda_{\mathcal{S}}$: il y a une annotation des $\vdash$ par des listes de type, de même que les composantes de gauche des $\rightarrow$ sont annotées.

$$\overline{\Gamma, x : A_\Sigma \vdash_\Sigma x \ : A|\Delta}$$

$$\frac{\Gamma, x : A_\Sigma \vdash_\Xi t \ : B|\Delta}{\Gamma \vdash_\Xi \lambda x.t \ : (A_\Sigma \to B)|\Delta} \qquad \frac{\Gamma \vdash_\Xi t \ : (A_\Sigma \to B)|\Delta \qquad \Gamma \vdash_\Sigma u \ : A|\Delta}{\Gamma \vdash_\Xi (t)u \ : B|\Delta}$$

$$\frac{\Gamma \vdash_\Sigma c \ : \bot|\Delta, \alpha : A}{\Gamma \vdash_\Sigma \mu\alpha.c \ : A|\Delta} \qquad \frac{\Gamma \vdash_{A\cdot\Sigma} c \ : \bot|\Delta}{\Gamma \vdash_\Sigma \mu\widehat{\mathsf{tp}}.c \ : A|\Delta}$$

$$\frac{\Gamma \vdash_\Sigma t \ : A|\Delta, \alpha : A}{\Gamma \vdash_\Sigma [\alpha]t \ : \bot|\Delta, \alpha : A} \qquad \frac{\Gamma \vdash_\Sigma t \ : A|\Delta}{\Gamma \vdash_{A\cdot\Sigma} [\widehat{\mathsf{tp}}]t \ : \bot|\Delta}$$

FIG. 6.3 – Système de types simples pour $\lambda\mu\widehat{\mathsf{tp}}$.

On peut faire la connection suivante entre les types de $\Lambda_{\mathcal{S}}$ et de $\lambda\mu\widehat{\mathsf{tp}}$: dans un jugement de la forme $x_1 : A^1_{\Sigma_1}, \ldots, x_k : A^k_{\Sigma_k} \vdash_\Sigma t \ : A|\alpha_1 : B_1, \ldots, \alpha_l : B_l$, les variables de termes x_i de type $A^i_{\Sigma_i}$ se verront attribuer un type $\tau(A^i \cdot \Sigma_i)$, le terme t se verra attribuer un type $\tau(A \cdot \Sigma)$ tandis que les variables de stream α_i recevront un type $\sigma(B_i)$ où $\tau(\Sigma)$ et $\sigma(B)$ sont définis comme suit :

Définition 6.15 ($\tau(\Sigma)$, $\sigma(B)$)

- $\tau(\bot) = o$;
- $\tau(A \cdot \Sigma) = \sigma(A) \Rightarrow \tau(\Sigma)$;
- $\sigma(o_i) = o_i \to \bot$;
- $\sigma(A_\Sigma \to B) = \tau(A \cdot \Sigma) \to \sigma(B)$.

On vérifie simplement que les images de τ sont des (pré-)types de terme tandis que les images de σ sont des (pré-)types de stream et que les règles de typage de $\lambda\mu\widehat{\mathsf{tp}}$ correspondent bien aux règles de $\Lambda_{\mathcal{S}}$, le rôle de l'associativité étant joué lors de la règle pour typer la λ-abstraction et la λ-application puisque la pile de type annotant le $\vdash$ n'est pas prise en compte dans la construction de type, c'est-à-dire, en termes de $\Lambda_{\mathcal{S}}$, qu'on construit la flèche $\to$ «sous» le constructeur $\Rightarrow$.

C'est ainsi le fait que dans $\lambda\mu\widehat{\mathsf{tp}}$, on ait un type principal et une annotation du $\vdash$ qui permet à Herbelin et Ghilezan d'éviter d'avoir une règle de conversion ou une relation d'équivalence sur les types.

On voit ainsi que le constructeur de piles de type de $\lambda\mu\widehat{\mathsf{tp}}$ n'est rien d'autre que le constructeur de types $\Rightarrow$ de $\Lambda_{\mathcal{S}}$ de même que l'opération de mise en indice $A \cdot \Sigma$ et que les types de terme sont simplement des types de stream (on rajoute un $\bot$ final, pour que cela se conforme à la syntaxe de stream de $\Lambda_{\mathcal{S}}$ mais ce n'est pas essentiel).

Cette section montre en particulier que le $\Lambda\mu$-calcul peut capturer les continuations délimitées, en appel par nom [HG08], qui forment un ensemble d'opérateurs de contrôle plus fins et plus expressifs que ceux capturés par le $\lambda\mu$-calcul de Parigot. On reviendra sur ce point en présentant une machine abstraite pour $\Lambda\mu$ en section 8.3.2.

6.6. Propriétés de $\Lambda_{\mathcal{S}}$

6.6.1 $\Lambda_{\mathcal{S}}$ type davantage de termes que le système $\lambda\mu$ de Parigot

Dans cette section, on montre que $\Lambda_{\mathcal{S}}$ peut typer tous les $\lambda\mu$-termes typables et on explicite la relation entre les types de $\lambda\mu$ et ceux de $\Lambda\mu$:

Théorème 6.16

Soit $t \in \Sigma_{\lambda\mu}$. S'il existe Γ, Δ et A tels que $\Gamma \vdash_{\lambda\mu} t\ : A|\Delta$, alors il existe $\Gamma_t^{\mathcal{S}}, \Delta_t^{\mathcal{S}}$ et $A_t^{\mathcal{S}}$ tels que $\Gamma_t^{\mathcal{S}} \vdash_{\Lambda_{\mathcal{S}}} t\ : A_t^{\mathcal{S}}|\Delta_t^{\mathcal{S}}$.

Définition 6.17

On considère $o_\perp$ une variable de type de terme spéciale et on définit les deux transformations suivantes sur des types du $\lambda\mu$-calcul de Parigot dans les pré-types de $\Lambda_{\mathcal{S}}$.

(i) Pré-types de terme :
- $(o)^{\mathcal{T}} = (o \to \perp) \Rightarrow o_\perp$;
- $(A \to B)^{\mathcal{T}} = A^{\mathcal{T}} \to B^{\mathcal{T}}$.

(ii) Pré-types de stream :
- $(o)^{\mathcal{S}} = o \to \perp$;
- $(A \to B)^{\mathcal{S}} = A^{\mathcal{T}} \to B^{\mathcal{S}}$.

Proposition 6.18

Étant donné un type simple A, alors $A^{\mathcal{T}} \equiv_{fst} A^{\mathcal{S}} \Rightarrow o_\perp$.

Démonstration : On a les égalités suivantes :

$$
\begin{array}{lrl}
(i) & o^{\mathcal{S}} \Rightarrow o_\perp & = (o \to \perp) \Rightarrow o_\perp \\
 & & = o^{\mathcal{T}} \\
(ii) & (A \to B)^{\mathcal{S}} \Rightarrow o_\perp & = (A^{\mathcal{T}} \to B^{\mathcal{S}}) \Rightarrow o_\perp \\
 & & \equiv_{fst} A^{\mathcal{T}} \to (B^{\mathcal{S}} \Rightarrow o_\perp) \\
 & & = A^{\mathcal{T}} \to B^{\mathcal{T}} \\
 & & = (A \to B)^{\mathcal{T}}
\end{array}
$$

□

On prouve maintenant le théorème :

Démonstration du théorème 6.16 : Le résultat est démontré par induction sur une $\lambda\mu$-dérivation de typage du terme t.

Un jugement de typage du $\lambda\mu$-calcul $(x_i : A_i)_{i\in I} \vdash t\ : C|(\alpha_j : B_j)_{j\in J}$ se traduit en un jugement de la forme : $(x_i : {A_i}^{\mathcal{T}})_{i\in I} \vdash t\ : C^{\mathcal{T}}|(\alpha_j : {B_j}^{\mathcal{S}})_{j\in J}$. On a donc :

- $\dfrac{}{\Gamma, x : A \vdash_{\lambda\mu} x\ : A|\Delta}\ Var_{\mathcal{T}}$ est transformé en

$$\dfrac{}{\Gamma^{\mathcal{T}}, x : A^{\mathcal{T}} \vdash_{\Lambda_{\mathcal{S}}} x\ : A^{\mathcal{T}}|\Delta^{\mathcal{S}}}\ Var_{\mathcal{T}}$$

- $\dfrac{\Gamma, x : A \vdash_{\lambda\mu} t\ : B|\Delta}{\Gamma \vdash_{\lambda\mu} \lambda x^A.t\ : A \to B|\Delta}\ \lambda\text{-Abs}$ est transformé en

$$\dfrac{\Gamma^{\mathcal{T}}, x : A^{\mathcal{T}} \vdash_{\Lambda_{\mathcal{S}}} t\ : B^{\mathcal{T}}|\Delta^{\mathcal{S}}}{\Gamma^{\mathcal{T}} \vdash_{\Lambda_{\mathcal{S}}} \lambda x^{A^{\mathcal{T}}}.t\ : (A \to B)^{\mathcal{T}}|\Delta^{\mathcal{S}}}\ \lambda\text{-Abs}$$

- $\dfrac{\Gamma \vdash_{\lambda\mu} t\ : A \to B|\Delta \qquad \Gamma \vdash_{\lambda\mu} u\ : A|\Delta}{\Gamma \vdash_{\lambda\mu} (t)u\ : B|\Delta}\ \lambda\text{-App}$ est transformé en

$$\dfrac{\Gamma^{\mathcal{T}} \vdash_{\Lambda_{\mathcal{S}}} t\ : (A \to B)^{\mathcal{T}}|\Delta^{\mathcal{S}} \qquad \Gamma^{\mathcal{T}} \vdash_{\Lambda_{\mathcal{S}}} u\ : A^{\mathcal{T}}|\Delta^{\mathcal{S}}}{\Gamma^{\mathcal{T}} \vdash_{\Lambda_{\mathcal{S}}} (t)u\ : B^{\mathcal{T}}|\Delta^{\mathcal{S}}}\ \lambda\text{-App}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{x : \mathcal{T}_x \vdash x \ : \mathcal{T}_x|}\,Var_{\mathcal{T}} \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{y : A^{\mathcal{S}} \Rightarrow o_\perp \vdash y \ : A^{\mathcal{S}} \Rightarrow o_\perp|}\,Var_{\mathcal{T}}}{y : A^{\mathcal{S}} \Rightarrow o_\perp \vdash (y)\alpha \ : o_\perp|\alpha : A^{\mathcal{S}}}\,App_{\mathcal{S}}}{y : A^{\mathcal{S}} \Rightarrow o_\perp \vdash \mu\beta.(y)\alpha \ : B^{\mathcal{S}} \Rightarrow o_\perp|\alpha : A^{\mathcal{S}}}\,Abs_{\mathcal{S}}}{\vdash \lambda y.\mu\beta.(y)\alpha \ : (A^{\mathcal{S}} \Rightarrow o_\perp) \to (B^{\mathcal{S}} \Rightarrow o_\perp)|\alpha : A^{\mathcal{S}}}\,Abs_{\mathcal{T}}}{x : ((A^{\mathcal{S}} \Rightarrow o_\perp) \to (B^{\mathcal{S}} \Rightarrow o_\perp)) \to (A^{\mathcal{S}} \Rightarrow o_\perp) \vdash (x)\lambda y.\mu\beta.(y)\alpha \ : A^{\mathcal{S}} \Rightarrow o_\perp|\alpha : A^{\mathcal{S}}}\,App_{\mathcal{T}}}{x : ((A^{\mathcal{S}} \Rightarrow o_\perp) \to (B^{\mathcal{S}} \Rightarrow o_\perp)) \to (A^{\mathcal{S}} \Rightarrow o_\perp) \vdash ((x)\lambda y.\mu\beta.(y)\alpha)\alpha \ : o_\perp|\alpha : A^{\mathcal{S}}}\,App_{\mathcal{S}}}{x : ((A^{\mathcal{S}} \Rightarrow o_\perp) \to (B^{\mathcal{S}} \Rightarrow o_\perp)) \to (A^{\mathcal{S}} \Rightarrow o_\perp) \vdash \mu\alpha.((x)\lambda y.\mu\beta.(y)\alpha)\alpha \ : A^{\mathcal{S}} \Rightarrow o_\perp|}\,Abs_{\mathcal{S}}}{\vdash \texttt{call/cc} \ : (((A^{\mathcal{S}} \Rightarrow o_\perp) \to (B^{\mathcal{S}} \Rightarrow o_\perp)) \to (A^{\mathcal{S}} \Rightarrow o_\perp)) \to (A^{\mathcal{S}} \Rightarrow o_\perp)|}\,Abs_{\mathcal{T}}$$

avec $\mathcal{T}_x = ((A^{\mathcal{S}} \Rightarrow o_\perp) \to (B^{\mathcal{S}} \Rightarrow o_\perp)) \to (A^{\mathcal{S}} \Rightarrow o_\perp)$.

FIG. 6.4 – Dérivation de typage pour `call/cc` dans $\Lambda_{\mathcal{S}}$.

– $\dfrac{\Gamma \vdash_{\lambda\mu} t \ : B|\Delta, \alpha : A}{\Gamma \vdash_{\lambda\mu} \mu\alpha^A(t)\beta \ : A|(\Delta, \beta : B) \setminus \alpha : A}\,\mu$ est transformé en

$$\dfrac{\dfrac{\Gamma^{\mathcal{T}} \vdash_{\Lambda_{\mathcal{S}}} t \ : B^{\mathcal{T}}|\Delta^{\mathcal{S}}, \alpha : A^{\mathcal{S}}, \beta : B^{\mathcal{S}}}{\Gamma^{\mathcal{T}} \vdash_{\Lambda_{\mathcal{S}}} (t)\beta \ : o_\perp|\Delta^{\mathcal{S}}, \alpha : A^{\mathcal{S}}, \beta : B^{\mathcal{S}}}\,App_{\mathcal{S}}}{\Gamma^{\mathcal{T}} \vdash_{\Lambda_{\mathcal{S}}} \mu\alpha^{A^{\mathcal{S}}}.(t)\beta \ : A^{\mathcal{S}} \Rightarrow o_\perp|(\Delta^{\mathcal{S}}, \beta : B^{\mathcal{S}}) \setminus A^{\mathcal{S}}}\,Abs_{\mathcal{S}}$$

La proposition précédente est suffisante pour assurer que la dérivation obtenue en appliquant ces règles est en effet une dérivation de typage de $\Lambda_{\mathcal{S}}$.

□

Remarque 6.19

Le théorème précédent permet de comprendre plus précisément quelles sont les limitations du $\lambda\mu$-calcul de Parigot s'agissant de la flexibilité du $\Lambda\mu$-calcul : les images des $\lambda\mu$-termes n'ont jamais besoin d'assigner un type de la forme $\mathcal{S}_1 \Rightarrow (\mathcal{S}_2 \Rightarrow A)$.

Remarque 6.20

La propriété précédente ne serait pas vraie si on l'énonçait à propos du système de typage classique présenté au début du chapitre, c'est-à-dire si on fait la comparaison avec l'ensemble des termes de $\Sigma_{\Lambda\mu}$. Par exemple, le terme $t = \lambda x, y, z.((z)(y)x)(y)(x)\alpha$ peut, dans le système de la définition 6.2, être typé avec le type $\perp \to (\perp \to A) \to (A \to A \to B) \to B$ avec α recevant le type $\perp$. Pourtant ce terme ne peut pas être typé en $\Lambda_{\mathcal{S}}$: dans $\Lambda_{\mathcal{S}}$, x et $(x)\alpha$ ne pourraient pas avoir le même type alors qu'ils sont tous les deux arguments du même terme y dans t.

En fait, cette restriction n'est pas surprenante puisque le système de typage de la définition 6.2 suppose que l'on donne le même type $\perp$ à tous les termes de la forme $(t)\alpha$.

Typer `call/cc` dans $\Lambda_{\mathcal{S}}$. L'encodage du `call/cc` en $\Lambda\mu$-calcul est le terme :

$$\lambda x.\mu\alpha.((x)\lambda y.\mu\beta.(y)\alpha)\alpha.$$

Dans le système de types classique de la définition 6.2, ce terme est typé par la Loi de Peirce : $((A \to B) \to A) \to A$. Dans $\Lambda_{\mathcal{S}}$, le `call/cc` peut recevoir le type $(((A^{\mathcal{S}} \Rightarrow o_\perp) \to (B^{\mathcal{S}} \Rightarrow o_\perp)) \to (A^{\mathcal{S}} \Rightarrow o_\perp)) \to (A^{\mathcal{S}} \Rightarrow o_\perp)$ comme montré par la dérivation de typage de la figure 6.4. On peut noter que la structure de la Loi de Peirce se retrouve au niveau des types de stream (remarquer l'alternance des types $A^{\mathcal{S}}$ et $B^{\mathcal{S}}$).

6.6.2 Préservation du type

Dans cette section, on prouve que le $\Lambda\mu$-calcul typé satisfait la préservation du type. La propriété de préservation du type dépend de trois lemmes montrant que le type est préservé par substitution si la substitution respecte les contraintes de typage. On a un lemme pour la substitution de β_T, un lemme pour celle de β_S et enfin un lemme indiquant que la substitution de la *fst*-réduction préserve également le type :

Lemme 6.21 (*Préservation du type par substitution $t\{u/x\}$*)

Si $\Gamma, x : A \vdash t \ : B|\Delta$ et $\Gamma \vdash u \ : C|\Delta$ avec $A \equiv_{fst} C$, alors $\Gamma \vdash t\{u/x\} \ : D|\Delta$ avec $B \equiv_{fst} D$.

Lemme 6.22 (*Préservation du type par substitution $t\{\beta/\alpha\}$*)

Si $\Gamma \vdash t \ : A|\Delta, \alpha : \mathcal{S}, \beta : \mathcal{S}'$ avec $\mathcal{S} \equiv_{fst} \mathcal{S}'$ alors $\Gamma \vdash t\{\beta/\alpha\} \ : C|\Delta, \beta : \mathcal{S}'$ avec $A \equiv_{fst} C$.

Lemme 6.23 (*Préservation du type par substitution $t\{(u)x\beta/(u)\alpha\}$*)

Si $\Gamma \vdash t \ : A|\Delta, \alpha : B \to \mathcal{S}$ alors $\Gamma, x : C \vdash t\{(u)x\beta/(u)\alpha\} \ : D|\Delta, \beta : \mathcal{S}'$ avec $B \equiv_{fst} C$, $\mathcal{S} \equiv_{fst} \mathcal{S}'$ et $A \equiv_{fst} D$

Démonstration : Les trois lemmes se démontrent par une induction simple sur la structure du terme t auquel s'applique la substitution.

□

Théorème 6.24 (*Préservation du type*)

La réduction des $\Lambda\mu$-termes typés préserve le type : soient $t, u \in \Lambda\mu$, si on peut dériver $\Gamma \vdash t \ : A|\Delta$ et si $t \longrightarrow^{\star}_{\Lambda\mu^{\to}} u$ alors on peut dériver $\Gamma \vdash u \ : A|\Delta$.

Démonstration : On raisonne par induction sur la longueur d'une dérivation de t à u et par cas sur la première règle de réduction appliquée.

□

6.6.3 Normalisation forte

Finalement, on démontre la propriété essentielle qu'est la normalisation forte : il n'y a pas de réduction infinie dans $\Lambda\mu^{\to}$:

Théorème 6.25 (*Le $\Lambda\mu$-calcul est fortement normalisant*)

Soit t un terme bien typé dans $\Lambda_{\mathcal{S}}$. Il n'y a aucune réduction infinie à partir de t.

Le théorème se démontre grâce à une méthode inspirée de l'une des preuves données par Parigot dans [Par97] pour démontrer la normalisation forte du $\lambda\mu$-calcul simplement typé. La méthode consiste à exhiber une traduction des $\Lambda\mu$-termes typés dans le λ-calcul simplement typé et à déduire la normalisation forte du $\Lambda\mu$-calcul typé de celle du λ-calcul typé.

On commence par donner une traduction des types de $\Lambda_{\mathcal{S}}$ dans les types simples :

Définition 6.26

On enrichit d'abord l'ensemble de variables de type du λ-calcul simplement typé : à chaque variable de type de stream σ, on associe une nouvelle variable de type simple également appelée σ. En outre, on ajoute une nouvelle variable $\sigma_\perp$. La traduction associe à chaque pré-type de $\Lambda_\mathcal{S}$ un type simple du λ-calcul de la manière suivante :

- $|o_i| = o_i$ *si* o_i *est une variable de type de terme.*
- $|\mathcal{T}_1 \to \mathcal{T}_2| = |\mathcal{T}_1| \to |\mathcal{T}_2|$
- $|(\mathcal{T}_1 \to \mathcal{S}) \Rightarrow \mathcal{T}_2| = |\mathcal{T}_1| \to |\mathcal{S} \Rightarrow \mathcal{T}_2|$
- $|\sigma_i \Rightarrow \mathcal{T}| = \sigma_i \to |\mathcal{T}|$
- $|\perp \Rightarrow \mathcal{T}| = \sigma_\perp \to |\mathcal{T}|$

Cette traduction définit en fait une traduction des types de $\Lambda_\mathcal{S}$ dans les types simples puisque tous les pré-types d'une même classe seront envoyés sur le même type simple :

Proposition 6.27

Si $\mathcal{T}_1 \equiv_{fst} \mathcal{T}_2$*, alors* $|\mathcal{T}_1| = |\mathcal{T}_2|$.

On traduit maintenant les $\Lambda\mu$-termes typés en λ-termes :

Définition 6.28

Pour toute variable de stream α, on considère de nouvelles variables du λ-calcul : $\alpha_1, \dots, \alpha_n, \dots$ La traduction est alors définie comme suit :

$$\begin{array}{rcl}
\lceil x \rceil^{\Lambda_\mathcal{S}} & = & x \\
\lceil \lambda x^A.t \rceil^{\Lambda_\mathcal{S}} & = & \lambda x^{|A|}.\lceil t \rceil^{\Lambda_\mathcal{S}} \\
\lceil (t)u \rceil^{\Lambda_\mathcal{S}} & = & (\lceil t \rceil^{\Lambda_\mathcal{S}})\lceil u \rceil^{\Lambda_\mathcal{S}} \\
\lceil \mu\alpha^{A_1 \to \dots A_n \to \sigma}.t \rceil^{\Lambda_\mathcal{S}} & = & \lambda\alpha_1{}^{|A_1|}.\dots.\lambda\alpha_n{}^{|A_n|}.\lambda\alpha_{n+1}{}^{|\sigma|}.\lceil t \rceil^{\Lambda_\mathcal{S}} \\
\lceil (t)\alpha \rceil^{\Lambda_\mathcal{S}} & = & (\lceil t \rceil^{\Lambda_\mathcal{S}})\alpha_1 \dots \alpha_{n+1} \text{ si } \alpha \text{ est de type } A_1 \to \dots A_n \to \sigma
\end{array}$$

où σ est soit une variable de type de stream soit $\perp$.

Il est à nouveau évident que la traduction ne dépend pas des représentants des $\Lambda_\mathcal{S}$-types considérés, de telle sorte que l'on a :

Proposition 6.29

Si t est un $\Lambda\mu$-terme bien typé, alors $\lceil t \rceil^{\Lambda_\mathcal{S}}$ est un λ-terme simplement typé.

__**Démonstration :**__ Il s'agit d'une induction facile sur la structure d'une dérivation de typage de t :

- les étapes de $\equiv_{fst}$ disparaissent,
- $Abs_\mathcal{T}$ (resp. $App_\mathcal{T}$) est renommé en $\to$-intro (resp. $\to$-elim) et
- $Abs_\mathcal{S}$ (resp. $App_\mathcal{S}$) est remplacé par $n+1$ étapes de $\to$-intro (resp. $\to$-elim) si n est l'arité du type de stream.

□

Proposition 6.30 (*Simulation*)

Étant donnés deux $\Lambda\mu$-termes t et u, on a les faits suivants :
- *Si $t \longrightarrow_{fst^{\rightarrow}} u$ alors $\lceil t \rceil^{\Lambda s} = \lceil u \rceil^{\Lambda s}$*
- *Si $t \longrightarrow_{\beta_T} u$ alors $\lceil t \rceil^{\Lambda s} \longrightarrow_{\beta} \lceil u \rceil^{\Lambda s}$*
- *Si $t \longrightarrow_{\eta_T} u$ alors $\lceil t \rceil^{\Lambda s} \longrightarrow_{\eta} \lceil u \rceil^{\Lambda s}$*
- *Si $t \longrightarrow_{\beta_S} u$ alors $\lceil t \rceil^{\Lambda s} \longrightarrow_{\beta}^{+} \lceil u \rceil^{\Lambda s}$*
- *Si $t \longrightarrow_{\eta_S} u$ alors $\lceil t \rceil^{\Lambda s} \longrightarrow_{\eta}^{+} \lceil u \rceil^{\Lambda s}$*

Démonstration : La preuve ne présente pas de difficulté.

□

Remarque 6.31

On peut en fait être plus précis et relier le nombre n de β ou η réductions nécessaires pour simuler une étape de β_S ou de η_S à l'arité du type de la variable de stream qui est en jeu dans la réduction : il s'agit de $n = \text{arité}(\mathcal{S}) + 1$.

Remarque 6.32

On notera que les traductions que l'on a choisies pour les constructions $\mu\alpha.t$ et $(t)\alpha$ étaient cruciales pour avoir la simulation de β_S et η_S par la clôture transitive et non-réflexive $\beta\eta$, ce qui simplifiera la preuve de normalisation forte dans la suite.

Proposition 6.33 (*$fst^{\rightarrow}$ termine*)

Soit t un $\Lambda\mu$-terme typé. Il n'y a pas de $fst^{\rightarrow}$-dérivation infinie à partir de t.

Démonstration : On prouve cela facilement parce qu'à chaque étape de réduction de $fst^{\rightarrow}$, l'arité du type d'une certaine variable de stream (celle à laquelle on applique la règle) décroît. Cela assure la terminaison de $fst^{\rightarrow}$.

□

Proposition 6.34 (*$\lceil \_ \rceil^{\Lambda s}$ augmente les longueurs de réduction*)

Si $t \longrightarrow_{\Lambda\mu^{\rightarrow}}^{\star} u$ avec m étapes de $\boldsymbol{\beta\eta}$-réduction, alors il existe une $\beta\eta$-réduction de $\lceil t \rceil^{\Lambda s}$ à $\lceil u \rceil^{\Lambda s}$ en au moins m étapes de réduction.

Remarque 6.35

Dans la proposition précédente, le nombre de réductions $fst^{\rightarrow}$ n'est pas pris en compte.

On peut finalement prouver la normalisation forte du $\Lambda\mu$-calcul typé :

Démonstration du théorème 6.25 : Supposons qu'il existe une suite de réductions typées infiniment longue à partir d'un $\Lambda\mu$-terme typé t : $\delta = (t_i)_{i \geq 0}$ avec $t = t_0$ et $t_i \longrightarrow_{\Lambda\mu^{\rightarrow}} t_{i+1}$.

Cette suite de réductions contient seulement un nombre fini d'étapes de $\boldsymbol{\beta\eta}$-réductions d'après la proposition 6.34 : autrement, nous obtiendrions une suite infinie de $\beta\eta$-réductions à partir de $\lceil t \rceil^{\Lambda s}$ dans le λ-calcul simplement typé, ce qui est impossible.

Il existe donc un entier n_0 tel que toutes les réductions de δ à partir de $t_{n_0} \longrightarrow_{fst^{\rightarrow}} t_{n_0+1}$ sont des réductions $\longrightarrow_{fst^{\rightarrow}}$. Par conséquent, si δ était une séquence infinie, nous aurions une réduction $fst^{\rightarrow}$ infinie à partir de t_{n_0}, ce qui contredit la terminaison de $fst^{\rightarrow}$ que nous assure la proposition 6.33.

Finalement, on peut conclure que le $\Lambda\mu$-calcul typé est fortement normalisant.

□

$$\frac{\Gamma \vdash_{\Lambda\mu^2} t \ : A|\Delta \qquad \text{avec } X_{\mathcal{T}} \notin FV(\Gamma, \Delta)}{\Gamma \vdash_{\Lambda\mu^2} t \ : \forall_t X_{\mathcal{T}}.A|\Delta} \forall_t - i \qquad \frac{\Gamma \vdash_{\Lambda\mu^2} t \ : \forall_t X_{\mathcal{T}}.A|\Delta}{\Gamma \vdash_{\Lambda\mu^2} t \ : A\{B/X_{\mathcal{T}}\}|\Delta} \forall_t - e$$

$$\frac{\Gamma \vdash_{\Lambda\mu^2} t \ : A|\Delta \qquad \text{avec } Y_{\mathcal{S}} \notin FV(\Gamma, \Delta)}{\Gamma \vdash_{\Lambda\mu^2} t \ : \forall_s Y_{\mathcal{S}}.A|\Delta} \forall_s - i \qquad \frac{\Gamma \vdash_{\Lambda\mu^2} t \ : \forall_s Y_{\mathcal{S}}.A|\Delta}{\Gamma \vdash_{\Lambda\mu^2} t \ : A\{P/Y_{\mathcal{S}}\}|\Delta} \forall_s - e$$

FIG. 6.5 – Règles de typage de la quantification pour le $\Lambda\mu$-calcul du second ordre.

La preuve de normalisation forte pour $\Lambda_{\mathcal{S}}$ montre à la fois que le système $\Lambda_{\mathcal{S}}$ est satisfaisant tout en mettant à jour certaines de ses limites : comme le $\lambda\mu$-calcul simplement typé, le $\Lambda\mu$-calcul simplement typé peut se plonger dans le λ-calcul simplement typé.

On propose dans la suite un système du second ordre pour le $\Lambda\mu$-calcul.

6.6.4 Vers le second ordre

Nous terminons ce chapitre en présentant un système F [Gir72, Gir86] pour $\Lambda_{\mathcal{S}}$ dans une présentation à la Curry [Kri90]. Ce système incorpore une quantification sur les types de terme, $\forall_t$, et une quantification sur les types de stream, $\forall_s$.

Définition 6.36 ($\Lambda\mu$-*calcul du second-ordre*)

Le $\Lambda\mu$-**calcul du second ordre** *est défini comme suit :*

- **Pré-Types** *(de terme et de stream) :*
  $\mathcal{T}, A, B, \ldots \quad ::= o_i \mid X_{\mathcal{T}} \mid A \to B \mid \mathcal{S} \Rightarrow \mathcal{T} \mid \forall_t X_{\mathcal{T}}.A \mid \forall_s Y_{\mathcal{S}}.A$
  $\mathcal{S}, P, Q, \ldots \quad ::= \sigma_i \mid Y_{\mathcal{S}} \mid \mathcal{T} \to \mathcal{S} \mid \bot$
- **Types** *: il s'agit des classes d'équivalence de pré-types modulo* $\equiv_{fst}$
- **Termes** *:*

$$t ::= \ x \mid \lambda x.t \mid (t)u \mid \mu\alpha.t \mid (t)\alpha.$$

Les nouvelles règles de typage pour le $\Lambda\mu$-calcul du second ordre sont données en figure 6.5.

Exemple de polymorphisme. On peut ainsi typer un terme comme :

$$t = \mu\alpha.\lambda x.(x)\alpha$$

avec le type $\forall_s Y_{\mathcal{S}}.Y_{\mathcal{S}} \Rightarrow (Y_{\mathcal{S}} \Rightarrow X_{\mathcal{T}}) \to X_{\mathcal{T}}$

On peut placer t dans le contexte $\mathcal{C}_1 = ([])y\alpha\lambda x.x$ en instanciant $Y_{\mathcal{S}}$ par $X_{\mathcal{T}} \to Y_{\mathcal{S}}$ ou dans le contexte $\mathcal{C}_2^u = ([])yz\alpha u$ où u peut prendre les valeurs $\lambda y, z.\mu\alpha.y$ ou $\lambda y, z.\mu\alpha.z$ en instanciant $Y_{\mathcal{S}}$ par $X_{\mathcal{T}} \to X_{\mathcal{T}} \to Y_{\mathcal{S}}$.

6.6.5 Formes $fst\boldsymbol{\eta}$-longues

On définit dans cette section une notion de formes $fst\boldsymbol{\eta}$-longues qui sont, pour le $\Lambda\mu$-calcul simplement typé, l'équivalent des formes normales η-longues du λ-calcul simplement typé pour le $\Lambda\mu$-calcul.

Définition 6.37 (***Forme fstη-longue***)

Soit t un $\Lambda\mu$-terme typé ne contenant pas de pré-redex (donc en forme normale pour βfst^-) de type $\mathcal{S}_1 \Rightarrow \mathcal{S}_2 \Rightarrow \dots \mathcal{S}_m \Rightarrow A_1 \to \dots A_n \to o$.

On définit la traduction $[t]^{fst\eta\ell}$ comme suit :

$$
\begin{array}{lcll}
\left[\lambda x^A.u\right]^{fst\eta\ell}_{abs} & = & \lambda x^A.[u]^{fst\eta\ell}_{abs} & \\
\left[\mu\alpha^{\mathcal{S}_1}.u\right]^{fst\eta\ell}_{abs} & = & \lambda x_1^{\alpha B_1}\dots\lambda x_n^{\alpha B_n}.\mu\alpha^\sigma.[u]^{fst\eta\ell}_{abs} & \text{si } \mathcal{S}_1 = B_1 \to \dots B_n \to \sigma \\
\left[t\right]^{fst\eta\ell}_{abs} & = & [\mu\alpha_1^{\mathcal{S}_1}\dots\mu\alpha_m^{\mathcal{S}_m}\lambda x_1^{A_1}\dots\lambda x_n^{A_n}. & \text{si } t = x, (u)v \text{ ou } (u)\beta \text{ et} \\
 & & (t)\alpha_1\dots\alpha_m x_1\dots x_n]^{fst\eta\ell}_{abs} & \text{si } (m,n) \neq (0,0) \\
 & & & \\
\left[t\right]^{fst\eta\ell}_{abs} & = & [t]^{fst\eta\ell}_{app} & \text{si } m = n = 0 \text{ (ie. } t \text{ de type } o) \\
\left[t\right]^{fst\eta\ell}_{app} & = & [t]^{fst\eta\ell}_{abs} & \text{si } t = \lambda x.u \text{ ou } t = \mu\alpha.u \\
 & & & \\
\left[(u)v\right]^{fst\eta\ell}_{app} & = & ([u]^{fst\eta\ell}_{app})[v]^{fst\eta\ell}_{abs} & \\
\left[(u)\alpha\right]^{fst\eta\ell}_{app} & = & ([u]^{fst\eta\ell}_{app})[x_1^\alpha]^{fst\eta\ell}_{abs}\dots[x_n^\alpha]^{fst\eta\ell}_{abs}\alpha & \text{si } u : (B_1 \to \dots B_n \to \sigma) \Rightarrow \mathcal{T} \\
\left[x\right]^{fst\eta\ell}_{app} & = & x &
\end{array}
$$

6.7. Conclusion

Nous avons introduit un système de types pour le $\Lambda\mu$-calcul qui, s'il s'éloigne de la correspondance avec la logique classique, permet de typer des termes de $\Lambda\mu$ qui ne seraient pas typables en logique classique et qui sont pourtant essentiels au bon déroulement de la preuve de séparation du chapitre 4, c'est-à-dire qu'ils manifestent des comportements calculatoires propres au $\Lambda\mu$-calcul et que l'on ne trouvait pas en $\lambda\mu$-calcul.

Ce système de types simples, qui renforce l'interprétation de streams déjà évoquée lorsque nous avons introduit le $\Lambda\mu$-calcul, a de bonnes propriétés (préservation du type et normalisation forte) et s'étend naturellement au second ordre. On notera également le lien très étroit entre le $\Lambda\mu$-calcul et le $\lambda\mu\widehat{\mathsf{tp}}$-calcul de Herbelin et Ghilezan [HG08] qui permet une analyse du contrôle délimité en appel par nom.

Le système $\Lambda_{\mathcal{S}}$ repose sur une règle d'associativité entre constructeurs de type de terme et de type de stream. Nous avons fait dans ce chapitre un parallèle informel entre l'associativité de ces deux constructeurs et l'associativité entre un $⅋$ et le connecteur $?(\_)^\perp ⅋ \_$. Nous développerons cette analogie au cours du chapitre suivant en définissant un système de réseaux qui étend les réseaux MELL dont nous étudierons la dynamique propre et dont nous montrerons qu'ils permettent d'encoder et de simuler $\Lambda\mu$.

Chapitre 7

Une analyse du $\Lambda\mu$-calcul via des réseaux

Résumé:

Nous étudions dans ce chapitre le $\Lambda\mu$-calcul via un encodage dans une classe particulière de réseaux, les Streams Associative Nets (SANE). On commence par définir les SANE, qui sont des réseaux intermédiaires entre les réseaux usuels de $MELL$ [Gir87a] et les réseaux polarisés de Laurent [Lau02]. La connexion entre les réseaux $MELL$ et les réseaux polarisés provient de l'associativité déjà mentionnée qui sera ici une véritable associativité entre deux sortes de connecteurs $\parr$. Nous étudions les propriétés de la réduction des réseaux SANE qui est notamment confluente et nous encodons le $\Lambda\mu$-calcul pur avec un procédé similaire à celui qui permet d'encoder le λ-calcul pur dans les réseaux $MELL$; cet encodage nous permet d'obtenir un résultat de simulation du $\Lambda\mu$-calcul par les SANE. Nous prouvons enfin un résultat de séparation pour les réseaux SANE.

Références. Ce chapitre développe les résultats du rapport de recherche *Stream associative nets and $\Lambda\mu$-calculus* [PS08] rédigé en collaboration avec Michele Pagani.

Sommaire

Remarque liminaire : Le contenu de ce chapitre est le fruit d'une collaboration avec Michele Pagani. Certains résultats ont donc été établis en commun, d'autres ont été établis par Pagani et d'autres, enfin, par l'auteur de cette thèse. Lorsqu'un résultat du rapport de recherche [PS08] est important pour le développement de ce chapitre mais que notre contribution à sa démonstration est mineure, nous le mentionnons en renvoyant le lecteur au rapport de recherche pour la preuve sous la forme d'une remarque suivant l'énoncé du résultat, de manière à ce qu'il n'y ait pas d'ambiguïté. Il nous est arrivé à plusieurs point de ce chapitre de développer des preuves qui n'étaient qu'ébauchées dans le rapport de recherche ; nous présentons les preuves, que la version du rapport de recherche soit d'un auteur ou de l'autre.

7.1. Des réseaux pour le $\Lambda\mu$-calcul

Dans la présente section, on définit des réseaux de démonstration qui nous permettront de coder le $\Lambda\mu$-calcul et de montrer un résultat de simulation du $\Lambda\mu$-calcul par les réseaux.

On procède comme pour le codage du λ-calcul dans les réseaux : on commence par isoler un fragment de la logique linéaire dont l'élimination des coupures nous permettra d'encoder la dynamique du $\Lambda\mu$-calcul et nous prenons un point fixe sur ces types pour coder le calcul pur comme avec l'équation $o = ?o^\perp \parr o$ pour le codage du λ-calcul pur dans les réseaux usuels.

On a déjà remarqué au chapitre 6 que l'on pouvait typer les $\Lambda\mu$-termes à l'aide d'un système de type faisant intervenir deux constructeurs $\rightarrow$ et $\Rightarrow$ en munissant ces types d'une équivalence correspondant à une associativité entre ces deux constructeurs. Par ailleurs, nous avons noté que le type $\mathcal{S} \Rightarrow \mathcal{T}$ ne faisait pas intervenir de dualisation de $\mathcal{S}$, ce qui est imposé par la règle d'associativité.

7.1.1 Formules pour les SANE

De manière à encoder le $\Lambda\mu$-calcul dans les réseaux de preuve, nous définissons le fragment suivant de la logique linéaire, conçu à partir du système de typage $\Lambda_{\mathcal{S}}$, comme discuté en section 6.4.3 :

Définition 7.1 (*Fragment ASLL (associative stream linear logic)*)

On considère le fragment suivant des formules de la logique linéaire :

$$\begin{array}{llcllll}
\mathcal{T}\text{-formules} & T,\ldots & := & o & \mid & ?A \parr T & \mid \; S \parr T \\
& A,\ldots & := & o^\perp & \mid & !T \otimes A & \mid \; P \otimes A \\
\\
\mathcal{S}\text{-formules} & S,\ldots & := & s & \mid & ?A \parr S & \\
& P,\ldots & := & s^\perp & \mid & !T \otimes S &
\end{array}$$

Puisque nous voulons considérer le $\Lambda\mu$-calcul pur, nous introduisons les équations récursives suivantes sur les formules (dans le même esprit que pour les réseaux purs [Dan90], [Reg92]) :

Définition 7.2 (*Équations sur les formules*)

On ajoute les équations suivantes sur les formules de la définition 7.1 :

$$\begin{array}{rclcrcl} o &=& ?o^{\perp} \parr o & \qquad & o^{\perp} &=& !o \otimes o^{\perp} \\ o &=& s \parr o & & o^{\perp} &=& s^{\perp} \otimes o^{\perp} \\ s &=& ?o^{\perp} \parr s & & s^{\perp} &=& !o \otimes s^{\perp} \end{array}$$

Ce qui nous donne exactement trois paires de formules duales dans le fragment considéré :

Définition 7.3 (*Formules de SANE*)

Formules négatives :	o	$?o^{\perp}$	s
Formules positives :	$o^{\perp}$	$!o$	$s^{\perp}$

Nous allons voir que les formules $o, o^{\perp}, !o$ et $?o^{\perp}$ sont traitées comme dans les réseaux purs de Danos et Regnier (c'est-à-dire que o et $o^{\perp}$ sont linéaires tandis que $!o$ et $?o^{\perp}$ permettent la duplication et l'effacement), mais que les formules s et $s^{\perp}$ sont de nature différente et seront traitées comme en logique linéaire polarisée (c'est-à-dire que les règles structurelles sont librement autorisées sur les formules négatives). Les formules s et $s^{\perp}$ seront utilisées pour typer les streams.

Remarque 7.4

On pourrait être tenté d'ajouter l'équation $?o^{\perp} = s$, nous ramenant ainsi à seulement deux paires de formules : $o, o^{\perp}$ et $s, s^{\perp}$, mais cela introduirait alors des incompatibilités lors de l'élimination des coupures (c'est-à-dire des coupures mal typées) comme par exemple une coupure entre un stream $\parr$ et une promotion.

Notation 7.5

Dans la suite, nous utiliserons la notation n pour faire référence, sans distinction, aux occurrences de s ou de $?o^{\perp}$.

7.1.2 Définition des réseaux SANE

Les ***réseaux de stream associatifs*** (Stream associative nets ou SANE), ou simplement ***réseaux*** dans la suite de ce chapitre, sont construits avec des ***cellules*** et des ***fils*** : un réseau est une combinaison de cellules, connectées les unes aux autres par des ***fils***, à la manière des réseaux d'interaction de Lafont [Laf95].

Plus précisément, tout réseau a un ensemble fini de ***ports libres***, également appelés ***conclusions du réseau*** et a donc un ensemble de ports constitué de ses ports libres et des ports de ses cellules (on suppose ces deux ensembles disjoints).

Définition 7.6 (***Cellules***)

Une **cellule** *est la donnée d'un* **type**, *qui est un symbole dans l'ensemble* $\{\otimes, \parr, c, !, ?d\}$, *et d'un nombre de* **ports** *au moins égal à* 1, *dont un exactement est le* **port principal** *ou* **port conclusion**, *tandis que les autres ports sont les* **ports auxiliaires** *ou* **ports prémisses**.

Nous représenterons les cellules comme des triangles et dessinerons les ports aux bords de ces triangles : le port principal d'une cellule est distingué en le plaçant à l'un des sommets du triangle. Les cellules de SANE sont présentées en figure 7.1.

Les ports des cellules de SANE sont annotés par des formules comme indiqué en figure 7.1. Pour distinguer les trois cellules de type $\parr$ *et les trois cellules de type* $\otimes$, *on les notera (en suivant l'ordre de la figure 7.1) :* $\parr^?, \parr^s, \parr^{?s}, \otimes^!, \otimes^{s^\perp}, \otimes^{!s^\perp}$.

De même qu'on a des formules positives et des formules négatives, on divise les cellules en deux classes : les **cellules positives** *(*$\otimes$ *et* $!$*) et les* **cellules négatives** *(*$\parr$, $?d$, *et* c*). Enfin, on appellera* **cellule de stream** *les cellules dont le port principal est étiqueté par* s *ou* $s^\perp$.

Les cellules sont données en figure 7.1, avec leurs règles de typage. La contraction généralisée a une arité variable $n \geq 0$, le cas d'égalité à zéro correspondant à la règle habituelle d'affaiblissement ; dans le cas $n = 1$, on adoptera la convention selon laquelle il s'agit d'un simple fil et si $n \geq 2$ cela correspond à un arbre de règles de contractions habituelles modulo associativité.

La cellule promotion est un cas particulier de cellule paramétrée par un réseau : si π est un réseau avec $n+1$ ports libres, alors $!\pi$ est une cellule avec un port principal et n ports auxiliaires. Le réseau π peut lui-même contenir des cellules promotion. On représentera parfois le réseau associé à une cellule promotion à l'intérieur de la cellule promotion elle-même comme avec la notation habituelle des boîtes.

Réseaux et cellules doivent donc être définis par récurrence mutuelle sur la profondeur du réseau :

Définition 7.7 (***Profondeur d'un réseau***)

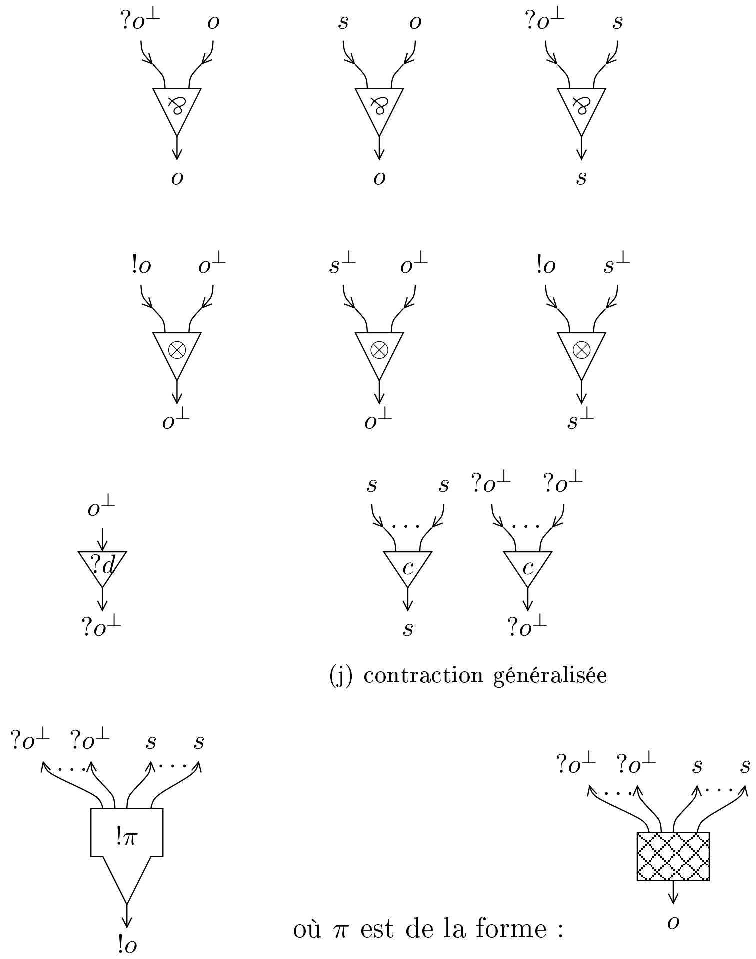
La **profondeur (exponentielle)** *d'un réseau* π *est le nombre maximal de cellules promotion emboîtées dans le réseau* π. *On définit de la même manière la* **profondeur exponentielle d'une cellule et d'un port** *dans un réseau donné* π.

Définition 7.8 (***Fil orienté***)

Un **fil orienté** *est un fil équipé d'une orientation, c'est-à-dire une paire ordonnée de ses deux ports.*

Un typage associe une formule A *à un fil orienté* w *de telle sorte que le fil orienté obtenu en inversant l'orientation de* w *est associé à la formule* $A^\perp$.

Un réseau π est donc un ensemble de fils et de cellules tel qu'on peut typer les fils orientés de π par des formules $o, o^\perp, s, s^\perp, ?o^\perp$ et $!o$, de telle sorte que les contraintes de la figure 7.1 soient satisfaites. Chaque port libre de π est équipé d'un type : il s'agit du type associé au fil connecté à ce port, orienté du port en direction du port libre.



(k) cellule promotion

FIG. 7.1 – Les cellules des réseaux SANE.

Définition 7.9 (*Axiomes et coupures d'un réseau*)

Une **coupure** *est un fil entre deux ports qui sont principaux ou auxiliaires dans le cas d'une cellule promotion. Un* **axiome** *est un fil entre deux ports libres ou deux ports auxiliaires (pour une cellule qui n'est pas une cellule promotion).*

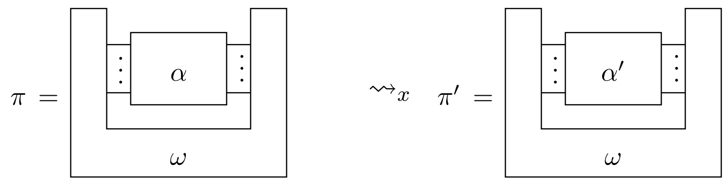
Une coupure dont une extrémité est un port auxiliaire d'une cellule promotion est appelé **coupure-commutative exponentielle**. *Nous avons trois types de coupures : les coupures* **multiplicatives** *(de labels $o, o^\perp$), les coupures* **exponentielles** *(de labels $!o, ?o^\perp$) et les coupures* **stream** *(de labels $s, s^\perp$).*

7.1.3 Règles de réécriture

On définit dans cette section les règles de réécriture pour les réseaux SANE qui nous permettront d'étudier ensuite leur dynamique.

Définition 7.10 (*Règle de réécriture sur les réseaux*)

Une règle de réécriture $\rightsquigarrow_x$ sur les réseaux est une transformation de graphe $\pi \rightsquigarrow_x \pi'$ qui consiste à prendre un sous-réseau α de π, le **redex**, *et à lui substituer un réseau α', le* **réduit**, *qui a les mêmes (nombres et types de) ports libres que α :*



Dans la suite, on définira les règles de réduction en représentant seulement les redex et les réduits (c'est-à-dire sans mentionner le contexte ω dans lequel s'inscrit cette réduction). Dès que $\rightsquigarrow_x$ est confluent et faiblement normalisant, on est assuré de l'existence et de l'unicité de la $\rightsquigarrow_x$-forme normale de π que l'on dénotera alors par $\mathsf{NF}^x(\pi)$.

La règle de réécriture que nous étudierons dans ce chapitre est notée $\rightsquigarrow_{SANE}$ et est l'union de quatre règles plus spécifiques :
- la ***réduction de coupure*** $\rightsquigarrow_{cut}$,
- la ***réduction de Retoré*** $\rightsquigarrow_r$,
- l'***expansion de fil*** $\rightsquigarrow_w$ et
- la ***réduction d'associativité*** $\rightsquigarrow_a$.

Ces deux dernières règles sont les deux éléments essentiels de nos réseaux SANE, qui permettent de faire interagir les liens streams et les liens exponentiels.

On présente maintenant successivement les quatre réductions qui constituent $\rightsquigarrow_{SANE}$.

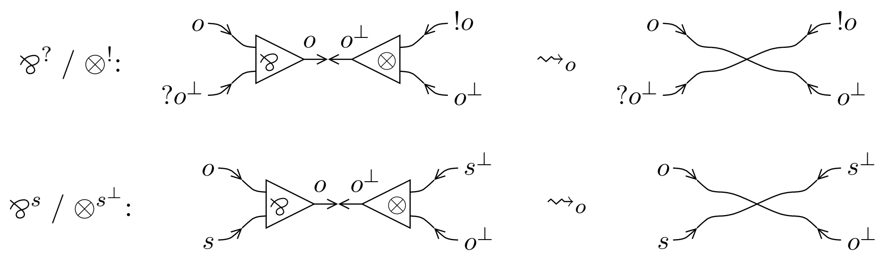
Réduction de coupure $\rightsquigarrow_{cut}$. On commence par rappeler la réduction $\rightsquigarrow_{cut}$, la réduction habituelle pour la logique linéaire polarisée (voir les travaux de Laurent [Lau03b]).

Définition 7.11 (*Réduction* $\rightsquigarrow_{cut}$)

On définit $\rightsquigarrow_{cut}$ comme l'union de deux réductions $\rightsquigarrow_o$ et $\rightsquigarrow_s$, définies ci-dessous en définitions 7.12 et 7.14.

Définition 7.12 (*Réduction* $\rightsquigarrow_o$)

La relation $\rightsquigarrow_o$ réduit les coupures labellisées par une formule o et est définie comme suit :



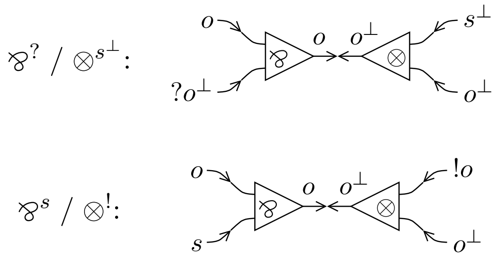


FIG. 7.2 – Coupures irréductibles.

Remarque 7.13

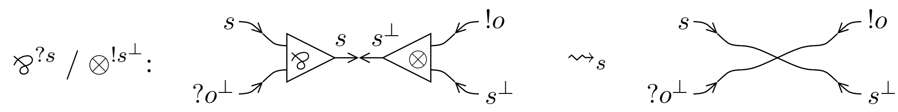
On remarquera que parmi les coupures marquées o, deux types de coupures ne sont pas réduites par $\rightsquigarrow_o$, elles sont montrées en figure 7.2 : ces coupures jouent un rôle crucial dans SANE puisqu'elles permettent une communication entre les ports labellisés par des formules de stream exponentielles. Cette communication n'est pour le moment que potentielle : jusqu'à présent, ces coupures sont irréductibles.

Définition 7.14 (*Réduction* $\rightsquigarrow_s$)

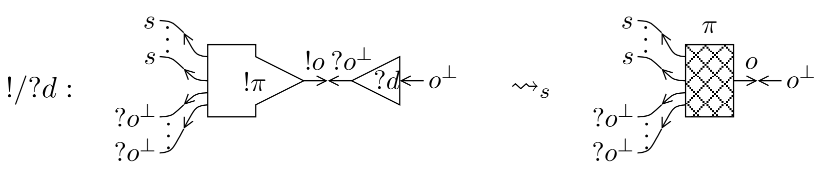
La réduction $\rightsquigarrow_s$ réduit les coupures portant une formule s ou $?o^\perp$.

Elle est définie par quatre types de réductions : $⅋^{?s}$ / $\otimes^{!s^\perp}$, $!/?d$, $\mathcal{S}/!$ et $\mathcal{S}/c$, détaillées aux définitions 7.15, 7.16, 7.18 et 7.19.

Définition 7.15 (*Réduction* $⅋^{?s}$ / $\otimes^{!s^\perp}$)



Définition 7.16 (*Réduction* $!/?d$)



On notera que la règle $!/?d$ échange les polarités de la coupure réduite, c'est-à-dire qu'elle crée un fil où les extrémités positives et négatives sont échangées.

Pour définir les deux autres règles constituant $\rightsquigarrow_s$, $\mathcal{S}/!$ et $\mathcal{S}/c$, on introduit la notion de $\otimes^{!s^\perp}$-arbre (qui est une adaptation immédiate à notre cadre des $\otimes$-arbres définis dans [Lau03b]).

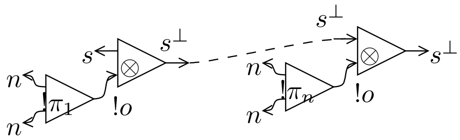


FIG. 7.3 – Un $\otimes^{!s^\perp}$-arbre générique.

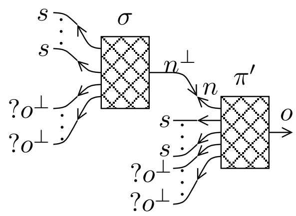


FIG. 7.4 – Réseau π'' de la définition 7.18.

Définition 7.17 ($\otimes^{!s^\perp}$*-arbre*)

Un $\otimes^{!s^\perp}$**-arbre** *est un réseau connexe et acyclique qui ne contient, en profondeur 0, que des cellules de type* $\otimes^{!s^\perp}$ *ou des cellules promotion et qui n'a pas de coupure.*

Étant donné un port p de type $n^\perp$ d'un réseau π, on appelle le $\otimes^{!s^\perp}$**-arbre de** *p le* $\otimes^{!s^\perp}$*-arbre maximal qui est un sous-réseau de π et qui a p comme port libre.*

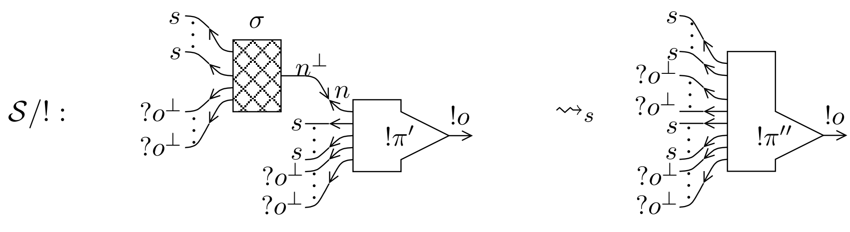
Les fils d'un $\otimes^{!s^\perp}$-arbre sont typables en utilisant seulement n (ou, de manière équivalente, $n^\perp$) et jamais $o, o^\perp$.

On peut prouver par induction sur la taille d'un $\otimes^{!s^\perp}$-arbre σ que les conclusions de σ sont exactement composées d'une conclusion de type $n^\perp$, appelée la ***racine*** de σ, et de $m \geq 0$ conclusions de types négatifs $?o$ ou s, appelées les ***feuilles*** de σ. La forme générale d'un $\otimes^{!s^\perp}$-arbre est représentée en figure 7.3.

Soit $\langle p, q\rangle$ une coupure labellisée par une formule n, soit p son extrémité de type $n^\perp$ et soit σ le $\otimes^{!s^\perp}$-arbre de p. La coupure $\langle p, q\rangle$ peut être de deux types en fonction de q, $\mathcal{S}/!$ ou $\mathcal{S}/c$:

Définition 7.18 (*Réduction* $\mathcal{S}/!$)

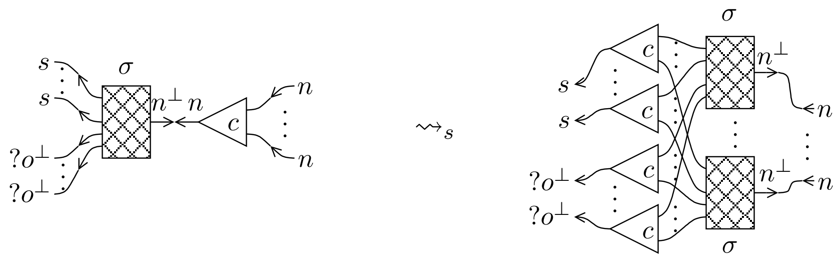
Si q est un port auxiliaire d'une promotion, alors on dit que $\langle p, q\rangle$ a le type $\mathcal{S}/!$, et on la réduit de la manière suivante (où π'' est présenté en figure 7.4) :



Définition 7.19 (*Réduction $\mathcal{S}/c$*)

Si q est port principal d'une contraction, on dit que $\langle p, q\rangle$ a le type $\mathcal{S}/c$, et on la réduit de la manière suivante :

$\mathcal{S}/c$:

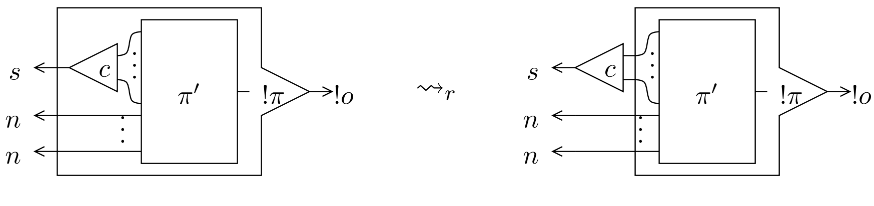


Réduction de Retoré, $\rightsquigarrow_r$. La règle $\rightsquigarrow_r$ consiste essentiellement à considérer les liens contraction comme des opérateurs associatifs qui peuvent flotter librement hors d'une cellule promotion. Diverses solutions ont été adoptées pour les réseaux de démonstration de la logique linéaire [CK97, CG99].

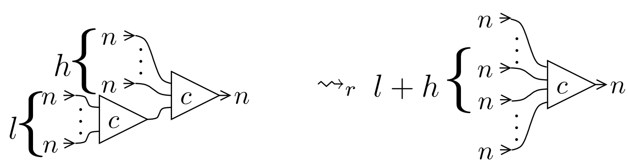
Définition 7.20 (*Réduction $\rightsquigarrow_r$*)

On définit $\rightsquigarrow_r$ comme l'union des réductions pull et fusion :

pull :



fusion :

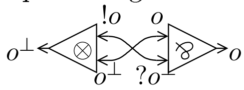


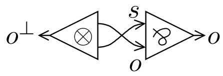
où, si $l = 0, h = 1$, le membre de droite consiste en une cellule contraction d'arité 1, c'est-à-dire qu'il s'agit d'un fil.

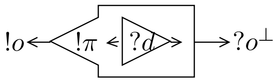
Expansions de fils. L'expansion de fils $\rightsquigarrow_w$ correspond au choix d'une orientation de l'équivalence extensionnelle, dans la direction de l'expansion.

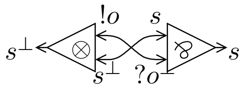
Définition 7.21 (*Réduction* $\rightsquigarrow_w$)

On définit $\rightsquigarrow_w$ comme l'union des quatre règles suivantes wo?, wos, w? et ws :

- *wo? :* $o^\perp \longleftrightarrow o \quad \rightsquigarrow_{wo?}$ 

- *wos :* $o^\perp \longleftrightarrow o \quad \rightsquigarrow_{wos}$ 

- *w? :* $!o \longleftrightarrow ?o^\perp \quad \rightsquigarrow_{w?}$ 

- *ws :* $s^\perp \longleftrightarrow s \quad \rightsquigarrow_{ws}$ 

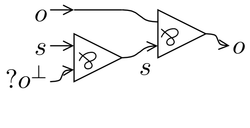
Remarque 7.22

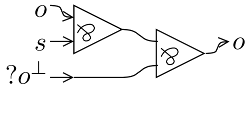
On souligne le fait que $\rightsquigarrow_w$ peut être appliquée à tout fil de type o ou s et non seulement aux axiomes comme c'est le cas quand on a des liens axiomes explicites [Dan90]. On note également que ws est la seule étape de $\rightsquigarrow_{SANE}$ qui crée des cellules de type $⅋^{?s}$ et $\otimes^{!s^\perp}$.

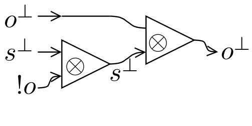
Réduction d'associativité. Jusqu'à présent, les diverses réductions présentées étaient relativement usuelles, dans la tradition des réseaux polarisés [Lau03b]. On introduit maintenant la véritable nouveauté des SANE, qui est la règle de réécriture $\rightsquigarrow_a$, construite à partir de l'associativité entre les différents types de ⅋ (et dualement de $\otimes$). Cette règle correspond à la relation $\succ_{fst}$ du système de typage $\Lambda_{\mathcal{S}}$ défini au chapitre 6 :

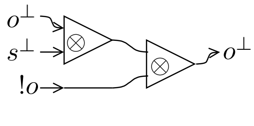
Définition 7.23 (*Réduction* $\rightsquigarrow_a$)

La réduction $\rightsquigarrow_a$ est définie par les deux réductions suivantes :



$\rightsquigarrow_a$




$\rightsquigarrow_a$


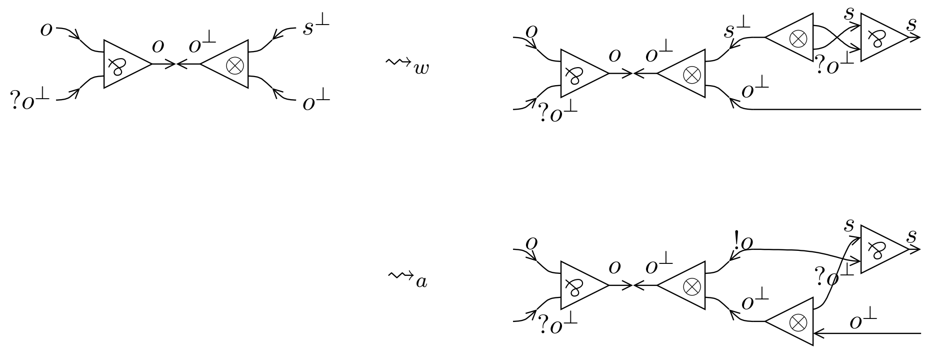
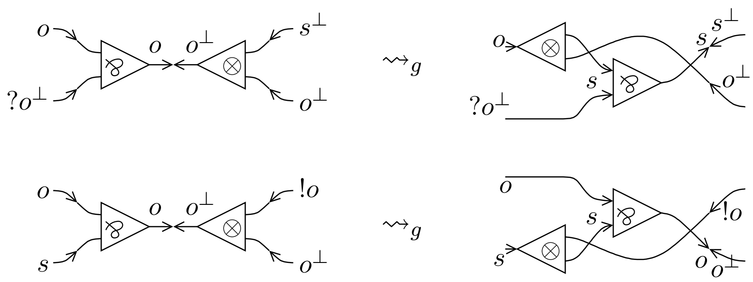


Fig. 7.5 – Des coupures irréductibles aux coupures réductibles.

Remarque 7.24

Le point clé à propos de la réduction d'associativité est que grâce à $\rightsquigarrow_a$ *et* $\rightsquigarrow_{ws}$ *on peut transformer les coupures irréductibles de la figure 7.2 en coupures réductibles, comme le montre par exemple la figure 7.5.*

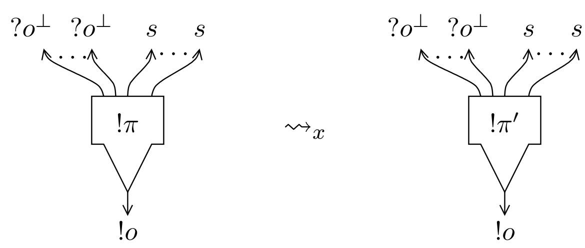
Dans la suite, il sera utile de considérer la règle dérivée $\rightsquigarrow_g$ *:*



qui est dérivable à partir de $\rightsquigarrow_{SANE}$ *: plus précisément,* $\rightsquigarrow_g = \rightsquigarrow_w \rightsquigarrow_a \rightsquigarrow_o$*. De plus* $\rightsquigarrow_{SANE} + \rightsquigarrow_g$ *a la même clôture transitive que* $\rightsquigarrow_{SANE}$*.*

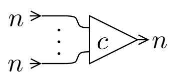
Pour achever la présentation de la réduction de nos réseaux, il nous faut encore considérer deux éléments : la clôture par promotion et l'équivalence par commutativité des ports auxiliaires des cellules contraction.

Clôture par promotion et équivalence par commutativité. La présence des cellules promotion nécessite de clore les règles de réduction définies jusque là par les cellules promotion. Pour chaque règle $\rightsquigarrow_x$, on ajoute le cas suivant :

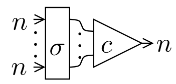


où $\pi \rightsquigarrow_x \pi'$.

En outre, on ajoute une équivalence $\sim_{comm}$ sur les réseaux qui est générée par l'équation de base suivante, pour chaque permutation σ :



$\sim_{comm}$



On remarque que cette équation n'interagit pas avec les réductions précédentes :

Proposition 7.25

Soient π_1, π'_1 et π_2 des réseaux tels que $\pi_1 \sim_{comm} \pi_2$ et $\pi_1 \rightsquigarrow_{SANE} \pi'_1$, alors il y a un réseau π'_2 tel que $\pi'_1 \sim_{comm} \pi'_2$ et $\pi_2 \rightsquigarrow_{SANE} \pi'_2$.

Cela signifie que $\rightsquigarrow_{SANE}$ est compatible avec l'équivalence $\sim_{comm}$: à partir de maintenant, on considérera toujours les réseaux à $\sim_{comm}$-équivalence près.

Définition 7.26 (*Réduction $\rightsquigarrow_{SANE}$*)

La réduction $\rightsquigarrow_{SANE}$ est définie comme l'union des réductions :
- $\rightsquigarrow_{cut}$;
- $\rightsquigarrow_{r}$;
- $\rightsquigarrow_{w}$;
- $\rightsquigarrow_{a}$;

sur les réseaux considérés modulo $\sim_{comm}$-équivalence.

7.1.4 Critère de correction

On définit dans cette sous-section les réseaux corrects et on énonce certaines de leurs propriétés :

Définition 7.27 (*Chemin dans un réseau*)

Un **chemin** *$(\phi, \psi \dots)$ dans un réseau est une suite de ports $\phi = \langle p_1, \dots, p_n \rangle$ telle que :*

(i) tous les ports sont à profondeur exponentielle 0 ;

(ii) un chemin ne passe jamais deux fois par le même port $(i < j \leq n \Rightarrow p_i \neq p_j)$;

(iii) le chemin est connexe : deux ports consécutifs p_i, p_{i+1} dans ϕ sont soit des ports de la même cellule, soit les deux extrémités du même fil ;

(iv) le chemin n'a pas de point de rebroussement au cours de la traversée d'une cellule : si $i \leq n-2$, p_i, p_{i+1}, p_{i+2} ne sont pas des ports de la même cellule.

Définition 7.28 (*Chemin négatif*)

Un chemin ϕ **croise** *un fil orienté $\langle p, q \rangle$ si $\langle p, q \rangle$ est une sous-suite de ϕ. Un chemin ϕ est* **négatif** *quand chaque fil orienté croisé par ϕ a un type négatif (o, $?o^{\perp}$ ou s).*

Définition 7.29 (*Cycle négatif*)

Un **cycle négatif** *est un chemin $\langle p_1, \dots, p_n \rangle$ tel que $\langle p_1, p_2, \dots, p_n, p_1 \rangle$ est un chemin négatif.*

Définition 7.30 (*Correction*)

Un réseau est **correct** *si :*
- *il ne contient pas de cycle négatif ;*
- *la somme du nombre de ses conclusions positives et de ses ?d-cellules à profondeur 0 vaut un ;*
- *et si, récursivement, les réseaux associés aux cellules promotions sont corrects.*

Théorème 7.31 (***Stabilité de la correction***)

La correction est préservée par $\rightsquigarrow_{SANE}$: pour tout réseau correct π, si $\pi \rightsquigarrow_{SANE} \pi'$ alors π' est correct.

Démonstration : La preuve est standard, voir [Lau03b]. On traite le cas d'une réduction $\pi_1 \rightsquigarrow_s \pi_2$ entre une promotion et une déréliction en profondeur 0.
- Le réseau π_1 contient une déréliction en profondeur 0, donc aucune conclusion positive puisqu'il est correct. Le réseau réduit π_2 a les même conclusions, et n'a donc aucune conclusion positive, et contient une déréliction en profondeur. En effet, le réseau π de la cellule promotion est correct et ses conclusions sont les conclusions auxiliaires de la cellules promotion (toutes négatives) et une conclusion o, elle aussi négative. Par correction de π on sait qu'il y a une déréliction en profondeur 0 dans π et donc dans π_2.
- Les réseaux associés aux cellules promotions n'ont pas changé, ils sont donc corrects.
- Supposons qu'il existe un cycle négatif dans le réseau π_2. Ce cycle ne pouvait déjà exister dans π_1 par l'hypothèse de correction de π_1. Le cycle doit être créé par la réduction, c'est-à-dire qu'il doit passer à travers le réseau π, mais toutes les conclusions de π sont négatives, il est donc impossible de rentrer dans π dans un chemin négatif.

□

Le théorème 7.35 affirme que $\rightsquigarrow_{s,r,a}$ est fortement normalisante pour les réseaux corrects :

Théorème 7.32 (***$\rightsquigarrow_{s,r,a}$ est fortement normalisante.***)

Soit π un réseau correct. Toute suite de $\rightsquigarrow_{s,r,a}$-réductions à partir de π est finie.

Remarque 7.33

On trouvera la preuve dans [PS08].

Théorème 7.34 (***Confluence de $\rightsquigarrow_{SANE}$***)

La réduction $\rightsquigarrow_{SANE}$ est confluente sur les réseaux corrects.

Remarque 7.35

On trouvera la preuve de ce théorème dans [PS08].

La preuve est obtenue en considérant la réduction $\rightsquigarrow_{SANE,g}$ plutôt que $\rightsquigarrow_{SANE}$ (les deux ont la même clôture transistive comme déjà noté) et en décomposant la réduction en plusieurs sous-systèmes dont on prouve successivement la confluence et la commutation (on passe en particulier par la confluence de la réduction $\rightsquigarrow_{s,r,a}$). Le théorème est utilisé pour montrer la confluence de $\rightsquigarrow_a$, $\rightsquigarrow_r$ et $\rightsquigarrow_s$ et de leurs unions. On montre en figures 7.6 et 7.7 quelques paires critiques.

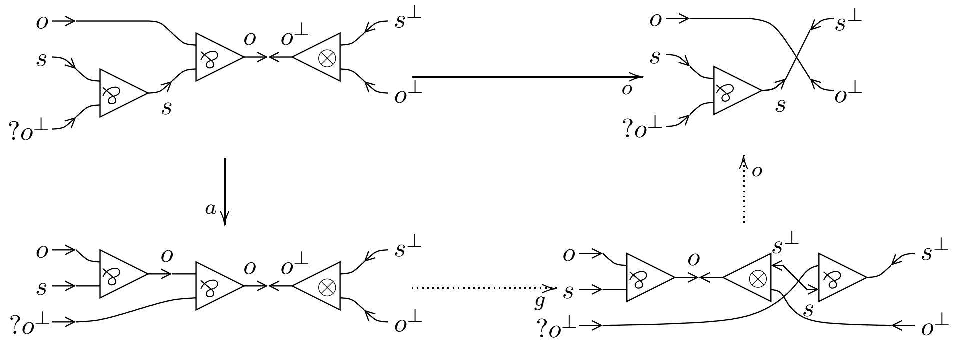


FIG. 7.6 – Paire critique $\rightsquigarrow_a$ / $\rightsquigarrow_o$.

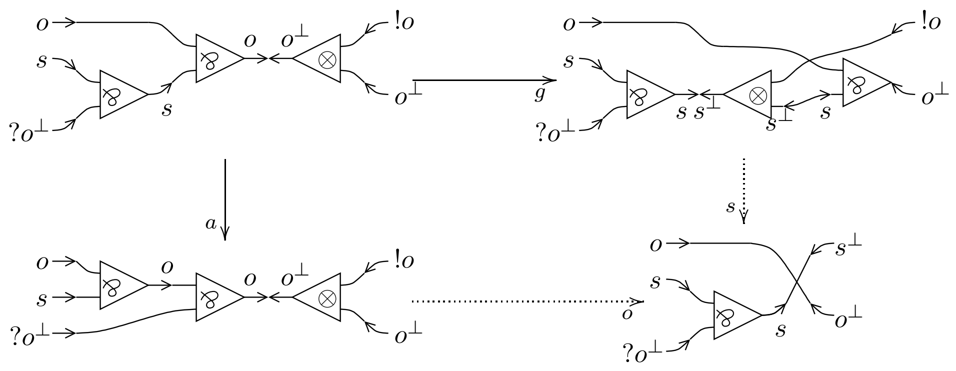


FIG. 7.7 – Paire critique $\rightsquigarrow_a$ / $\rightsquigarrow_g$.

7.1.5 Simulation du $\Lambda\mu$-calcul par les SANE

On propose dans cette section une traduction des $\Lambda\mu$-termes dans les réseaux de stream corrects, que l'on notera $(\,)^\circ$.

Définition 7.36 (*Traduction des $\Lambda\mu$-termes dans SANE*)

Soient t un $\Lambda\mu$-terme tel que $FV_T(t) \subseteq L \subset \mathcal{V}_T$ et $FV_S(t) \subseteq M \subset \mathcal{V}_S$ (L, M finis).

La traduction $(t, L, M)^\circ$ est donnée en figure 7.8. On fait correspondre à chaque conclusion du réseau défini un élément et un seul de $L \cup M \cup \{t\}$, de telle sorte que :

(i) t est associé avec l'unique conclusion de type o,

(ii) toute variable de L est associée à une conclusion de type $?o^\perp$ et

(iii) toute variable de M est associée à une conclusion de type s.

Remarque 7.37

Pour simplifier les notations de la traduction présentée en figure 7.8, on désignera les ports libres de $(t, L, M)^\circ$ de type $?o^\perp$ par la variable de L correspondante et les ports libres de $(t, L, M)^\circ$ de type s par la variable de M correspondante, en supposant $L = \{x_1, \ldots, x_l\}$ et $M = \{\alpha_1, \ldots, \alpha_m\}$.

La définition suivante caractérise les réseaux qui sont des traductions de $\Lambda\mu$-termes (théorème 7.39) :

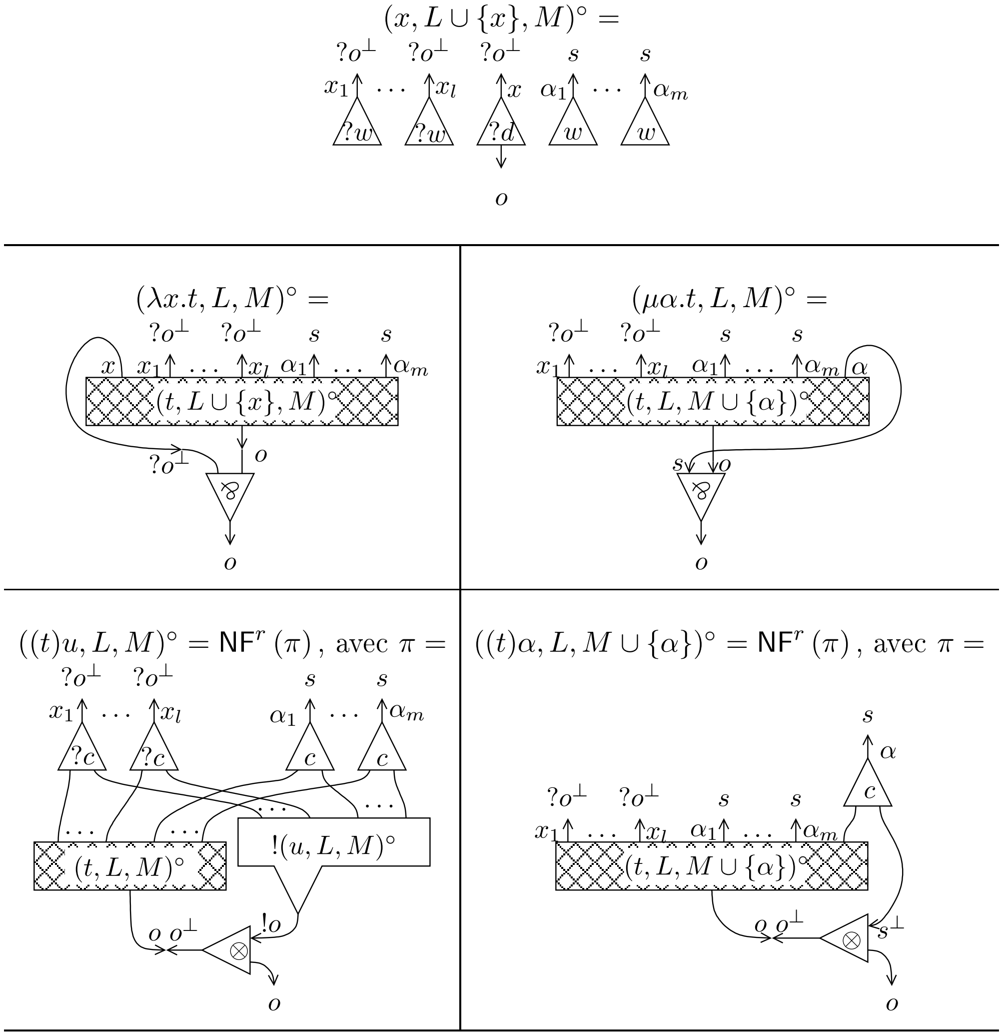


FIG. 7.8 – Traduction des $\Lambda\mu$-termes dans *SANE*.

Définition 7.38 ($\Lambda\mu$-*réseau*)

*Un $\Lambda\mu$-***réseau** *est un réseau correct π en forme normale pour $\rightsquigarrow_{s,r,a}$ tel que :*

1. *π ne contient pas de cellule de type $\parr^{?,s}$ ou $\otimes^{!,s^\perp}$;*
2. *tout port libre de π est négatif ;*
3. *et récursivement, les réseaux associés aux cellules promotion sont des $\Lambda\mu$-réseaux.*

Théorème 7.39 (*Séquentialisation*)

Un réseau π est un $\Lambda\mu$-réseau si, et seulement si, il y a un $\Lambda\mu$-terme t, des ensembles finis $L \subset \mathcal{V}_T$ et $M \subset \mathcal{V}_S$ tels que $\pi = (t, L, M)^\circ$.

En outre, il y a une bijection naturelle entre les coupures de $(t, L, M)^\circ$ et les $\Lambda\mu$-coupures (ou pré-redex, voir la définition 3.17) de t telles qu'une coupure de type $\parr^? / \otimes^!$ (resp. $\parr^? / \otimes^{s^\perp}$, $\parr^s / \otimes^!$, $\parr^s / \otimes^{s^\perp}$) correspond à une $\Lambda\mu$-coupure de type $(\mathcal{T})\mathcal{T}$ (resp. $(\mathcal{T})\mathcal{S}$, $(\mathcal{S})\mathcal{T}$, $(\mathcal{S})\mathcal{S}$). En particulier, $(t, L, M)^\circ$ est sans coupure (et donc est une forme normale pour $\rightsquigarrow_{cut,g,a,r}$) si, et seulement si t est canonique.

Remarque 7.40

On trouvera la preuve du théorème dans [PS08].

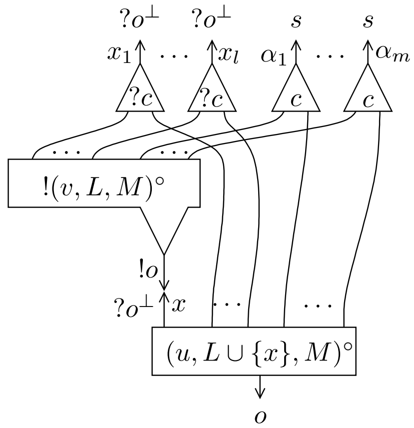
On s'intéresse maintenant plus en détail à la dynamique de $\rightsquigarrow_{SANE}$ pour la comparer à celle de $\Lambda\mu$. Dans ce qui suit on considérera la variante de $\longrightarrow_{\Lambda\mu}$ où les règles d'extensionalité sont orientées dans le sens de l'expansion, puisque nous avons considéré des expansions de fils pour les réduction $\rightsquigarrow_w$ et on prouvera que $\rightsquigarrow_{SANE}$ simule $\longrightarrow_{\Lambda\mu}$: si $t \longrightarrow_{\Lambda\mu} u$, $FV_T(t) \subseteq L$, $FV_S(t) \subseteq M$, alors $(t, L, M)^\circ \rightsquigarrow_{SANE} (u, L, M)^\circ$ (théorème 7.45).

Les deux lemmes suivants sont des variantes simples de lemmes correspondants pour les réseaux polarisés [Lau03b] ; ils sont démontrés dans [PS08].

Lemme 7.41 (*λ-substitution*)

Soit π le réseau suivant :

$\pi =$



alors $\mathsf{NF}^{s,r}(\pi) = (u\{v/x\}, L, M)^\circ$.

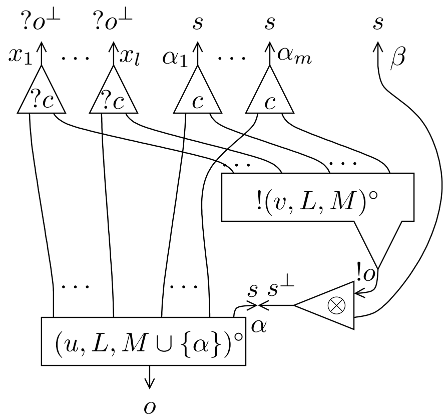
Remarque 7.42

On trouvera la preuve du lemme dans [PS08].

Lemme 7.43 (*μ-substitution*)

Soit π le réseau suivant :

$\pi =$



alors $\mathsf{NF}^{s,r,a}(\pi) = (u\{(t)v\beta/(t)\alpha\}, L, M \cup \{\beta\})^\circ$.

Remarque 7.44

On trouvera la preuve du lemme dans [PS08].

Dans l'énoncé suivant, on omet de préciser les ensembles L et M pour ne pas surcharger les notations.

Théorème 7.45 (*Simulation*)

Soient t, u deux $\Lambda\mu$-termes, alors :

a) $t \longrightarrow_{\beta_T} u$ implique que $t^\circ \rightsquigarrow_o \cdot \rightsquigarrow^*_{s,r} u^\circ$;

b) $t \longrightarrow_{\beta_S} u$ implique que $t^\circ \rightsquigarrow_o u^\circ$;

c) $t \longrightarrow_{fst} u$ implique que $t^\circ \rightsquigarrow_{ws} \cdot \rightsquigarrow_{w?} \cdot \rightsquigarrow^*_{s,r,a} u^\circ$;

d) $t \longrightarrow_{\eta_T^{exp}} u$ implique que $t^\circ \rightsquigarrow_{wo?} \cdot \rightsquigarrow_{w?} u^\circ$;

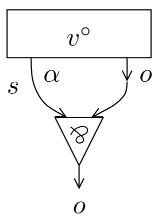
e) $t \longrightarrow_{\eta_S^{exp}} u$ implique que $t^\circ \rightsquigarrow_{wos} \cdot \rightsquigarrow_{w?} u^\circ$.

Démonstration : On peut se restreindre au cas où le redex qui est réduit dans $t \longrightarrow_{\Lambda\mu} u$ est le redex de tête de t, le cas plus général s'en déduit par une induction immédiate sur la complexité exponentielle de t en utilisant la clôture des réductions de $\longrightarrow_{SANE}$ par promotion.

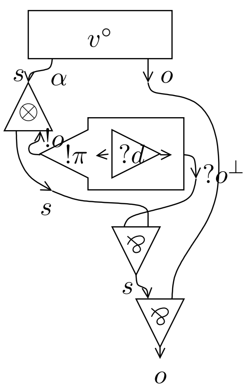
a) Supposons $t \longrightarrow_{\beta_T} u$. Le rédex de t réduit à cette étape correspond, dans $(t, L, M)^\circ$, à une coupure de type $⅋^?$ / $\otimes^!$ d'après le théorème 7.39. En réduisant cette coupure par une étape de $\rightsquigarrow_o$, on obtient un réseau π qui est dans la configuration de celui représenté dans le lemme 7.41. Le lemme nous permet alors de conclure que $\mathsf{NF}^{s,r}(\pi) = (u, L, M)^\circ$;

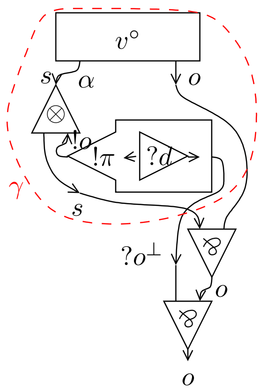
b) Supposons $t \longrightarrow_{\beta_S} u$. Le rédex de t réduit à cette étape correspond, dans $(t, L, M)^\circ$, à une coupure de type $⅋^s$ / $\otimes^{s^\perp}$ d'après le théorème 7.39. La réduction de cette coupure de $(t, L, M)^\circ$ nous fait obtenir directement le réseau $(u, L, M)^\circ$;

c) Supposons $t \longrightarrow_{fst} u$ et $t = \mu\alpha.v$. En omettant d'écrire les ports libres/variables de L et M, on a :

$t^\circ =$ 

et donc

$t^\circ \rightsquigarrow_{ws} \cdot \rightsquigarrow_{w?}$ 

$\rightsquigarrow_a$ 

Le lemme 7.43 nous assure que $\mathsf{NF}^{s,r,a}(\gamma) = v\{(w)x\beta/(w)\alpha\}^\circ$, d'où l'on conclut puisque $u = \lambda x.\mu\beta.v\{(w)x\beta/(w)\alpha\}$.

d) et e) Supposons. $t \longrightarrow_{\boldsymbol{\eta}^{exp}} u$ Les réductions de $\rightsquigarrow_w$ simulent directement cette expansion puisque $t^\circ \rightsquigarrow_{wo?} \cdot \rightsquigarrow_{w?} \pi$ avec π la traduction de $\lambda x.(t)x$ et $t^\circ \rightsquigarrow_{wos} \cdot \rightsquigarrow_{w?} \pi$ avec π la traduction de $\mu\alpha.(t)\alpha$

$\square$

Remarque 7.46

On notera qu'on n'a besoin, dans la preuve du théorème 7.45, que d'une version très restreinte du lemme 7.43 puisqu'on ne l'utilise que dans le cas où v est une variable. On en déduit en revanche simplement qu'on peut simuler la μ-réduction $((\mu\alpha.t)u \longrightarrow_\mu \mu\alpha.t\,\{(v)u\alpha/(v)\alpha\})$. Ceci n'est pas surprenant puisque, comme nous l'avons déjà noté, la μ-réduction est dérivable en $\Lambda\mu$-calcul. Pour autant, il est intéressant de noter que la simulation de la μ-réduction peut se faire via $\rightsquigarrow_g \cdot \rightsquigarrow^*_{s,r,a}$ où $\rightsquigarrow_g$ est la règle qui a été introduite en remarque 7.24 comme étant $\rightsquigarrow_w\rightsquigarrow_a\rightsquigarrow_o$ et qui a déjà été utilisée à plusieurs reprises auparavant.

On notera également qu'on voit se dégager du théorème de simulation une notion de substitutions explicites qui correspondrait aux réductions $\rightsquigarrow_{s,r}$ dans SANE ($\rightsquigarrow_{s,r,a}$ dans le cas de la fst-réduction). Plusieurs systèmes de substitutions explicites sont déjà connus pour $\Lambda\mu$, mais on peut noter que cette substitution explicite vient avec de bonnes propriétés puisqu'elle est fortement normalisante (théorème 7.35) et confluente (voir la remarque concernant le théorème de confluence 7.34).

7.2. Un théorème de Séparation pour les SANE

On montre dans cette partie un théorème de séparation pour les réseaux SANE.

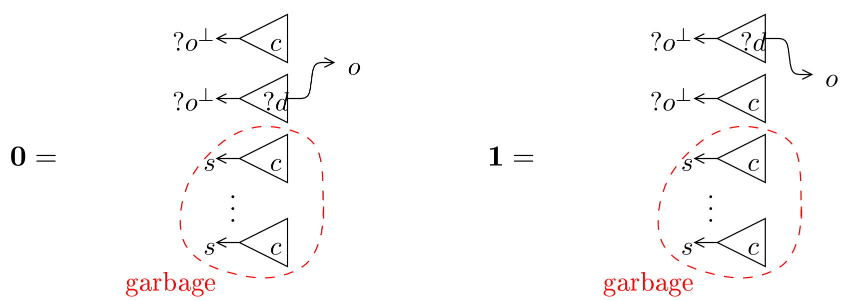
Définition 7.47 (*Valeur SANE*)

Un réseau correct est une **valeur** *quand il ne contient ni coupure ni redex pour les règles $\rightsquigarrow_{cut}$, $\rightsquigarrow_r$, $\rightsquigarrow_a$ (c'est-à-dire qu'il ne contient que des redex pour l'expansion des axiomes $\rightsquigarrow_w$).*

Les valeurs SANE sont le pendant dans les réseaux des formes normales canoniques définies dans le chapitre 3.

Définition 7.48 (*Réseaux 0 et 1*)

On définit deux réseaux particuliers, **1** *et* **0** *que nous utiliserons pour séparer les réseaux. Les valeurs seront définies modulo un certain nombre d'affaiblissements (c'est-à-dire de cellules contraction 0-aire) jouant le rôle de garbage collector :*



Dans la suite, les fils contractés du garbage collector de **1** et **0** ne perturberont jamais la preuve du théorème de séparation, on omettra donc le plus souvent de les mentionner pour ne pas surcharger les notations.

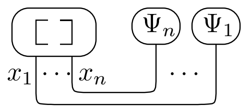
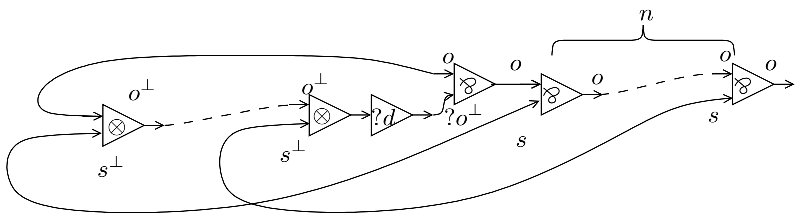


FIG. 7.9 – Contexte de tête.

Le réseau suivant nous servira à de multiples reprises au cours de la séparation.

Définition 7.49 (ϕ^n)

Soit n un entier, on définit le réseau ϕ^n comme suit :

$\phi^n =$ 

Définition 7.50 (*Contextes*)

Soit I un ensemble d'occurrences de formules de SANE. Les **contextes de type** *I sont définis comme des réseaux construits comme précédemment mais avec une cellule additionnelle, notée [], qui a les ports I. Les réseaux corrects sont simplement les réseaux corrects avec cette cellule additionnelle.*
Soit π un réseau de conclusions I, alors $\mathcal{C}(\pi)$ est défini comme le réseau obtenu à partir de $\mathcal{C}$ en remplaçant toutes les occurrences de [] dans $\mathcal{C}$ par π.

La correction de π et de $\mathcal{C}$ implique la correction de $\mathcal{C}(\pi)$ (la réciproque n'est pas vraie).

Définition 7.51 (*Contextes de tête*)

Une classe de contextes corrects particulièrement importante est celle des **contextes de tête** *: $\mathcal{H} =< []|x_1 : \Psi_1, \ldots, x_n : \Psi_n >$ où $x_1, \ldots x_n$ sont les conclusions du trou [] et les Ψ_i sont des réseaux (quand cela ne sera pas ambiguë, on écrira simplement $\mathcal{H} =< []|\Psi_1, \ldots, \Psi_n >$, voire $\mathcal{H} = \vec{\Psi}$). Dans le cas d'un contexte de tête $\mathcal{H} = \vec{\Psi}$, on écrit aussi $\pi^{\vec{\Psi}}$ pour $\mathcal{H}(\pi)$.*

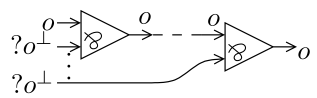
Remarque 7.52

Les contextes de tête sont donc simplement des contextes tels que chaque conclusion du trou [] est connectée à un réseau disjoint des autres, de la forme illustrée en figure 7.9.

Il est évident que la composition de deux contextes de tête est encore un contexte de tête.

Définition 7.53 ($\mathcal{A}(-,-)$)

Soit n un entier, on définit le réseau $\mathcal{A}\left(\parr^?, n\right)$ comme l'arbre composé de n cellules de type $\parr^?$:

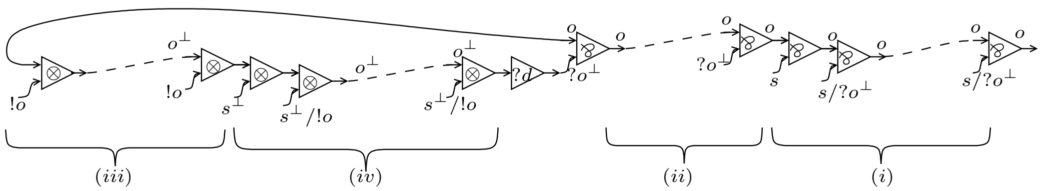


De manière similaire, on a $\mathcal{A}(x,n)$, pour $x \in \{\parr^s, \parr^{?s}, \otimes^!, \otimes^{s^\perp}, \otimes^{!s^\perp}\}$.

On introduit une classe particulière de réseaux qui nous servira au cours de la preuve de séparation : le lemme qui suit la définition indique certaines configurations dans lesquelles on peut séparer simplement certains réseaux.

Définition 7.54 (*Réseaux klmn*)

Les **réseaux** *$klmn$ sont les réseaux corrects tels que, partant de leur conclusion de type o, on trouve un sous-réseau de la forme suivante :*



Définition 7.55 (*Mesure d'un réseau klmn*)

On associe à un réseau $klmn$ une **mesure** *(k,l,m,n) comme suit (en reprenant les notations de la figure de la définition 7.54) :*

En entrant par la conclusion o, il y a un $o^\perp$-chemin maximal qui croise des cellules $\parr$ puis des cellules $\otimes$ et finalement atteint la déréliction de profondeur 0. La mesure est alors définie comme suit :

(i) k est le nombre de cellules $\parr^s$ que l'on croise sur ce $o^\perp$-chemin ;

(ii) l est le nombre de cellules $\parr^?$ que l'on croise après la dernière cellule $\parr^s$ et avant d'atteindre le $\parr^?$ lié à la déréliction (ce dernier $\parr^?$ ne compte pas) ;

(iii) n est le nombre de $\otimes^!$ qui sont croisés avant que le premier $\otimes^{s^\perp}$ ne soit atteint ;

(iv) m est le nombre de $\otimes^{s^\perp}$ qui sont croisés avant d'atteindre la déréliction.

Lemme 7.56

Étant donnés deux réseaux $klmn$ π et π' de mesures (k,l,m,n) et (k',l',m',n').

Si $(k,l,m,n) \neq (k',l',m',n')$ alors il y a un contexte de tête $\mathcal{C}$ tel que :

- *$\mathcal{C}(\pi) \rightsquigarrow^\star_{SANE} \mathbf{0}$ et*
- *$\mathcal{C}(\pi') \rightsquigarrow^\star_{SANE} \mathbf{1}$.*

Démonstration : On prouve le résultat en étudiant les différents cas possibles pour $(k,l,m,n) \neq (k',l',m',n')$:

(i) **si** $k < k'$. Soit $\mathcal{C}$ le contexte de la figure 7.10. On peut vérifier que si on coupe $\mathcal{C}$ avec π aux conclusions de types respectivement $o^\perp$ et o, on obtient $\mathcal{C}(\pi) \rightsquigarrow^\star_{SANE} \mathbf{0}$ et $\mathcal{C}(\pi') \rightsquigarrow^\star_{SANE} \mathbf{1}$. En effet :
- les k cellules $\parr^s$ de π (ainsi que les $\parr^?$ qui sont intercalés entre eux) et les k premiers $\parr^s$ de π' sont consommés par la réduction avec le sous-réseau

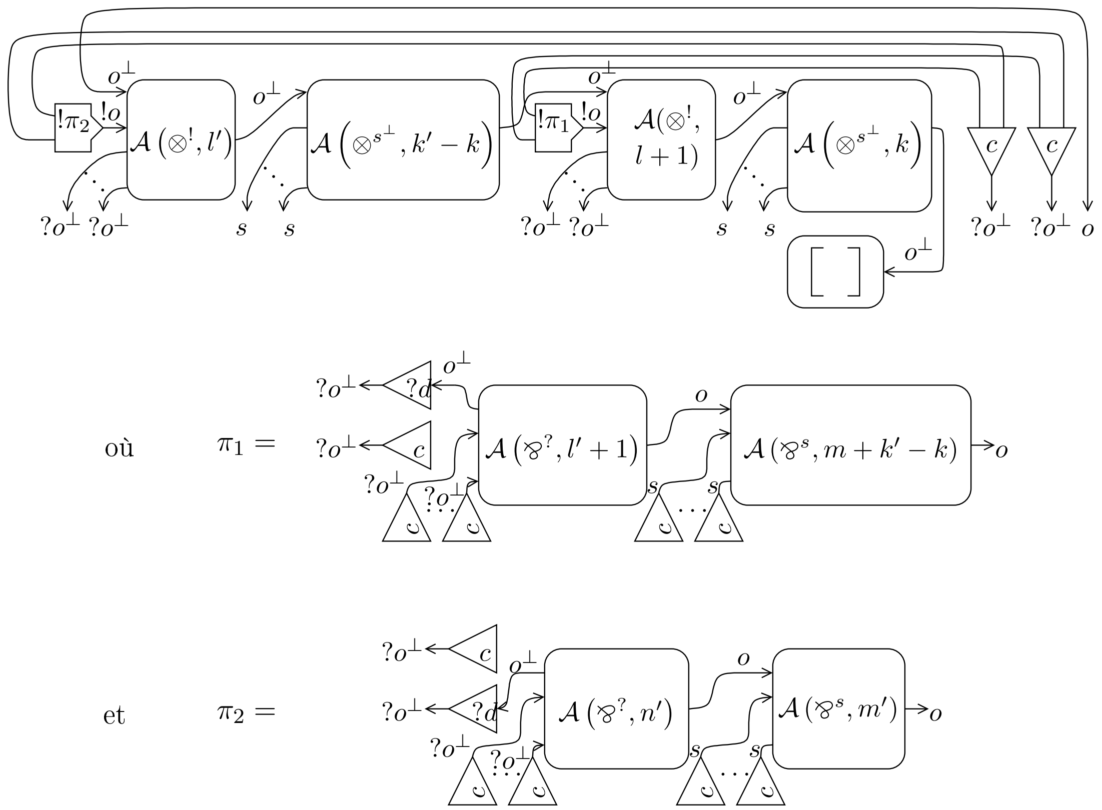


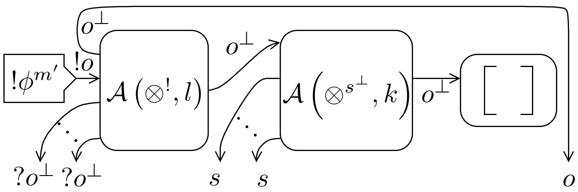
FIG. 7.10 – Contexte pour le lemme 7.56, dans le cas $k < k'$.

$\mathcal{A}\left(\otimes^{s^\perp}, k\right)$;

- les l cellules $⅋^?$ qui suivent et le $⅋^?$ qui a pour prémisse la cellule déréliction dans π sont consommés par le sous-réseau $\mathcal{A}\left(\otimes^!, l+1\right)$ du contexte tandis que, interagissante avec π', cette portion du contexte rencontre un certain nombre de $⅋^?$ puis un $⅋^s$;
- dans le cas de $\mathcal{C}(\pi)$, on a donc $!\pi_1$ connecté par une coupure à la déréliction tandis que la conclusion $o^\perp$ de la structure $\mathcal{A}\left(\otimes^{s^\perp}, k'-k\right)$ est connectée au port auxiliaire du dernier $\otimes^!$ ou $\otimes^{s^\perp}$ du réseau *klmn* π. On a donc une cellule déréliction qui conclut un réseau contenant une structure de tenseurs avec $k'-k+m$ cellules $\otimes^{s^\perp}$ et l' cellules $⅋^?$ après le dernier $\otimes^{s^\perp}$. Ce réseau est donc coupé avec π_1 et leur réduction conduit au réseau 0, les affaiblissements étant obtenus via les affaiblissements de π_1.
- dans le cas de $\mathcal{C}(\pi')$, les $k'-k$ cellules $⅋^s$ restantes se réduisent avec la sous-structure $\mathcal{A}\left(\otimes^{s^\perp}, k'-k\right)$ et les l' cellules $⅋^?$ se réduisent face à $\mathcal{A}\left(\otimes^!, l'\right)$. Finalement, on a $!\pi_2$ coupé avec le sous-arbre de π' enraciné à la déréliction de profondeur et la réduction conduit au réseau 1.

(ii) **Si** $k = k'$ **et** $l < l'$**.** Ce cas est résolu de manière similaire en utilisant de légères variantes de $\mathcal{C}, \pi_1$ et π_2, notées $\mathcal{C}', \pi_1'$ et π_2', essentiellement dues au fait que $k' = k$, on utilise ici le fait que $l < l'$ pour faire que la déreliction en profondeur 0 de π soit coupée avec $!\pi_1'$ et que la déreliction en profondeur 0 de π' soit coupée avec π_2'.

(iii) **Si** $k = k'$**,** $l = l'$ **et** $m < m'$**.** Soit $\mathcal{C}$ le contexte suivant ($\phi^{m'}$ est le réseau de la définition 7.49) :



alors $\mathcal{C}(\pi)$ et $\mathcal{C}(\pi')$ se réduisent vers les réseaux π_0 et π'_0 qui sont des réseaux $klmn$ avec $k_{\pi_0} \neq k_{\pi'_0}$, on peut donc conclure grâce à (i).

(iv) **Si $k = k'$, $l = l'$, $m = m'$ et $n < n'$.** Ce cas se traite de manière similaire au cas (iii) avec un contexte qui diffère légèrement du contexte $\mathcal{C}$ ci-dessus par une variante de ϕ^n qui permet de ramener le problème au cas (ii).

□

Le lemme suivant indique comment traiter les réseaux contenant des cellules ⅋ en profondeur 0. Ce lemme permet de préparer les réseaux à séparer de manière à les mettre dans une forme adaptée à la séparation.

Lemme 7.57

Si π et π' sont des valeurs non-équivalentes avec les mêmes conclusions et telles qu'il y a des cellules ⅋ en profondeur 0, alors il y a un contexte de tête correct $\mathcal{C}$ tel que $\mathcal{C}(\pi) \rightsquigarrow^\star_{SANE} \pi_1$ et $\mathcal{C}(\pi') \rightsquigarrow^\star_{SANE} \pi'_1$ où π_1 et π'_1 sont des valeurs non-équivalentes avec la même profondeur que π et π' et tels que π_1 et π'_1 n'ont pas de cellule ⅋ en profondeur 0.

Démonstration : La preuve consiste en une simple induction sur le nombre de cellules ⅋ en profondeur 0 dans π et π'.

Si aucun des deux réseaux n'a de cellule de type ⅋ en profondeur 0, il n'y a bien évidemment rien à faire. Supposons maintenant qu'on ait le résultat pour toute paire de valeurs non-équivalentes contenant au plus n cellules de type ⅋ et supposons que les valeurs π et π' totalisent $n+1$ cellules de type ⅋.

On va s'intéresser à des cellules ⅋ que nous appellerons ***cellules-conclusions*** et telles qu'il existe un chemin négatif du port principal de cette cellule à une conclusion du réseau qui ne traverse (possiblement) que des cellules contractions.

On commence par montrer que tout réseau qui est une valeur contenant une cellule ⅋ en profondeur 0 contient une cellule-conclusion. En effet, soit π une valeur contenant une cellule ⅋ en profondeur 0 et soit C l'une quelconque de ces cellules. Considérons un chemin négatif maximal à partir du port principal de C. Les types rencontrés au cours d'un tel chemin peuvent être soit o, soit s, mais en aucun cas il ne peut s'agir de $?\,o^\perp$: un tel chemin négatif peut pénétrer dans une cellule dans l'un des cas suivants uniquement :

- la cellule est de type ⅋ et le chemin entre par un port auxiliaire de type s ou o et en ressort par le port principal, de type s ou o ;
- la cellule est de type c, le chemin pénètre dans la cellule par le port auxiliaire et ressort par le port principal, de type s ;
- aucun autre cas n'est possible puisque (i) l'entrée dans une cellule de type $\otimes$ impliquerait l'existence d'une coupure si l'on considère la première cellule de ce type rencontrée sur le chemin, (ii) l'entrée par une promotion est également impossible puisqu'elle supposerait que l'on parcourt un fil orienté de type $?\,o^\perp$, ce qui n'est pas, (iii) un chemin négatif ne peut en aucun cas atteindre une cellule déréliction, il ne peut qu'en partir.

Par ailleurs, puisqu'il n'existe pas de cycle négatif dans une valeur, un chemin maximal doit forcément s'arrêter et cela ne peut être que sur un port libre du réseau, sans cela on contredirait soit le fait que π est une valeur, soit la maximalité du chemin.

Il existe, dans le chemin maximal dont on vient de montrer l'existence, une dernière cellule de type $\parr$ atteinte avant de rencontrer une conclusion du réseau ; il s'agit d'une cellule-conclusion.

On montre maintenant que l'on peut diminuer le nombre de cellules $\parr$ en éliminant une cellule-conclusion. On étudie donc les conclusions négatives de π et π' qui ont des cellules-conclusions dans π ou dans π' ; elles sont de type o ou s. On distingue les cas suivants :

i) il y a une conclusion x de type o au-dessus de laquelle il y a une cellule-conclusion $\parr^?$ dans π. On va alors couper cette conclusion avec un $\otimes^!$ en plaçant π et π' dans un contexte de tête $\mathcal{H} =< []|x : \Psi >$ où Ψ est un réseau réduit à une cellule $\otimes^!$. On raisonne par cas sur la conclusion correspondante de π' :
   - Soit la conclusion x de π' a également au-dessus d'elle une cellule $\parr^?$. Dans ce cas, en réduisant les deux coupures obtenues dans $\mathcal{H}(\pi)$ et $\mathcal{H}(\pi')$, on obtient deux nouvelles valeurs dont le nombre de cellules $\parr^?$ (et donc de cellules $\parr$) a diminué de deux.
   - Soit la conclusion x de π' a au-dessus d'elle une cellule $\parr^s$. Dans ce cas, on peut réduire la coupure dans $\mathcal{H}(\pi)$ grâce à une réduction $\rightsquigarrow_{cut}$ et on peut appliquer, dans $\mathcal{H}(\pi')$, une réduction $\rightsquigarrow_{ws}$ ce qui crée un rédex pour $\rightsquigarrow_a$. En effectuant cette réduction, on crée une coupure de type $\parr^?$ / $\otimes^!$ que l'on réduit, obtenant le réseau π'_1. On a ainsi diminué de 1 le nombre de cellules $\parr$ de π (il y a une cellule $\parr^?$ en moins), celui de π' n'a pas évolué. Le réseau π'_1 est une valeur car l'introduction du $\otimes^!$ par la réduction $\rightsquigarrow_w$ ne peut pas avoir créé de redex : le cas contraire signifierait qu'il existait déjà un redex dans pour $\rightsquigarrow_a$ dans π, ce qui est impossible puisque π est une valeur.
   - Enfin, la conclusion x de π' peut être connectée au port auxiliaire d'un $\otimes$. Dans ce cas, $\mathcal{H}(\pi)$ ne contient ni coupure ni redex, il s'agit d'une valeur et on obtient deux nouvelles valeurs dont le nombre de $\parr$ a diminué de 1.

ii) il y a une conclusion x de type o au-dessus de laquelle il y a une cellule-conclusion $\parr^s$ dans π et il n'y a pas de cellule-conclusion de type $\parr^?$ au-dessus de x dans π'. Dans ce cas, on place les réseaux π et π' dans un contexte de tête $\mathcal{H} =< []|x : \Psi >$ où Ψ est un réseau réduit à une cellule $\otimes^{s^\perp}$. Si π' a effectivement une cellule-conclusion au-dessus de x, alors $\mathcal{H}(\pi)$ et $\mathcal{H}(\pi')$ se réduisent vers des valeurs ayant chacune une cellule $\parr$ de moins en profondeur 0. Si π' n'a pas de cellule-conclusion au-dessus de x, alors $\mathcal{H}(\pi)$ se réduit vers une valeur ayant une cellule $\parr$ de moins et $\mathcal{H}(\pi')$ est une valeur ayant le même nombre de $\parr$ en profondeur 0 que π'.

iii) s'il y a une conclusion x de type s au-dessus de laquelle il y a une cellule-conclusion dans π, on procède comme suit. Comme x est de type s, elle peut être contractée et on peut donc avoir plusieurs cellules-conclusions au-dessus de x, ces cellules étant toutes de type $\parr^{?s}$; dans l'autre cas, le fil de type s est port auxiliaire soit d'une promotion, soit d'un $\otimes^s$ ou $\otimes^{!s^\perp}$. On va donc placer π et π' dans un contexte de tête de la forme $\mathcal{H} =< []|x : \Psi >$ où Ψ est un réseau réduit à une cellule $\otimes^{!s^\perp}$. En réduisant $\mathcal{H}(\pi)$ et $\mathcal{H}(\pi')$, on diminue le nombre de $\parr$ du nombre de cellules-conclusions au-dessus de x dans π et π', et donc d'au moins 1. Dans les autres cas, on a éventuellement créé un rédex d'associativité (la conclusion était port auxiliaire d'un $\otimes^{s^\perp}$) ou une coupure commutative exponentielle (la conclusion était port auxiliaire d'une promotion), mais dans tous les cas on se ramène à une valeur après avoir réduit ce redex et on a diminué le nombre de cellules $\parr$.

□

Définition 7.58 (*Sous-réseaux positifs maximaux*)

Soit π une valeur. On définit π_+, le **sous-réseau positif maximal** *de π et on distingue une conclusion particulière du réseau, x_π, comme suit :*

- *Si π a une conclusion positive x, alors $x_\pi = x$ et π_+ est l'arbre tenseur maximal qui est sous-réseau de π avec x pour conclusion.*
- *Si π a une déréliction en profondeur 0, alors x_π est l'unique conclusion qui est sous la déréliction et π_+ est le sous-réseau contenant les cellules menant de x_π à la déréliction ainsi que l'arbre tenseur maximal au-dessus de la déréliction.*

Remarque 7.59

Le réseau π_+ et la conclusion x_π sont bien définis.

En effet, si π a une conclusion positive, la définition ne pose pas de problème. Sinon, il y a une déréliction en profondeur 0. En suivant un chemin négatif au-dessous de la déréliction, on ne peut entrer dans des cellules que par des ports auxiliaires de cellules $\parr^?, \parr^s, \parr^{?s}$ ou de contraction (autrement le réseau contiendrait une coupure or il s'agit d'une valeur). L'absence de cycle négatif (le réseau est également correct) nous assure qu'on finit par atteindre une conclusion du réseau, qui est l'unique conclusion du réseau sous la déréliction.

Théorème 7.60 (*Séparation pour les SANE*)

Soient π et π' deux réseaux corrects avec les mêmes conclusions, qui sont des valeurs non-équivalentes : $\pi \not\simeq_{SANE} \pi'$.

Il existe un contexte (de tête) correct $\mathcal{C}$ tel que $\mathcal{C}(\pi) \leadsto^{\star}_{SANE} \mathbf{0}$ et $\mathcal{C}(\pi') \leadsto^{\star}_{SANE} \mathbf{1}$.

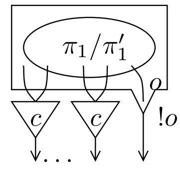
Démonstration : Pour obtenir le résultat par induction, on prouve en fait un résultat plus fort :

(*) soient π et π' deux réseaux corrects ayant les mêmes conclusions (que l'on note $I = \{x_1^{t_1}, \ldots, x_k^{t_k}\}$) qui sont des valeurs non-équivalentes : $\pi \not\simeq_{SANE} \pi'$. Pour tout $J = \{x_{i_1}^{?\,o^\perp}, \ldots, x_{i_l}^{?\,o^\perp}\} \subseteq I$, si $\mathcal{N} = \{n_1, \ldots, n_l\}$ est une famille d'entiers distincts qui sont suffisamment grands et si $\vec{\phi}$ est le contexte de tête $< []|x_{i_1} :!\phi^{n_1}, \ldots, x_{i_l} :!\phi^{n_l} >$, alors il existe un contexte de tête correct $\mathcal{C}$ tel que $\mathcal{C}(\pi^{\vec{\phi}}) \leadsto^{\star}_{SANE} \mathbf{0}$ et $\mathcal{C}({\pi'}^{\vec{\phi}}) \leadsto^{\star}_{SANE} \mathbf{1}$.

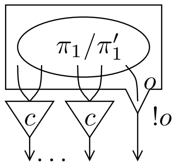
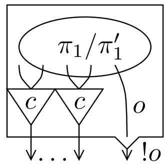
Nous ne donnons pas une preuve complètement détaillée, ce qui serait trop long et superflu, mais nous tentons de traiter les cas principaux et les cas les plus complexes. La preuve procède par induction sur la somme des profondeurs de π et π'. Soit J un sous-ensemble des conclusions de type $?o^\perp$. Grâce au lemme 7.57, on peut supposer que ni π ni π' n'ont de cellule $\parr$ en profondeur 0 : il y a un contexte dans lequel les réseaux se réduisent à des valeurs non-équivalentes de profondeurs identiques à celles des réseaux initiaux et n'ayant pas de cellule $\parr$ en profondeur 0. Soient $x_i = x_\pi$ et $x_j = x_{\pi'}$. On raisonnera sur la structure des sous-réseaux positifs maximaux π_+ et π'_+ :

1. **S'il y a une conclusion positive** x (et alors $x = x_i = x_j$), on raisonne sur le type de x.

   (a) *Si la conclusion positive est de type* $!\,o$, on a alors des réseaux π et π' de la forme :

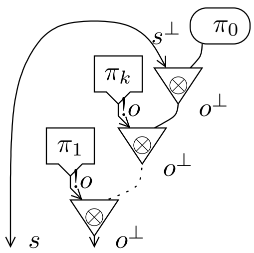
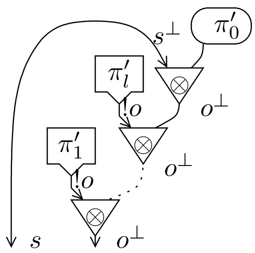
$\pi/\pi' =$ 

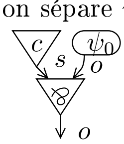
puisqu'il n'y a pas de $\parr$ en profondeur 0. Par l'équivalence associée à la réduction $\rightsquigarrow_r$, on a :

$\pi/\pi' =$  $\sim_r$  $= \tilde{\pi}/\tilde{\pi}'$

On peut donc appliquer l'hypothèse d'induction aux réseaux π_1/π'_1 avec les contractions appropriées de telle sorte que les réseaux ont la même conclusions excepté pour le $!o$ qui devient maintenant o. Par l'hypothèse d'induction, si ϕ satisfait les conditions de l'hypothèse d'induction, on trouve ainsi un contexte séparant pour ${\pi_1}^{\vec{\phi}}/{\pi'_1}^{\vec{\phi}}$: $\mathcal{C} =< []|\psi_1^{n^\perp}, \ldots, \psi_{l'}^{n^\perp}, \psi_{l'+1}^{o^\perp} >$ et en ajoutant une déréliction à la conclusion de type $o^\perp$ de $\psi_{l'+1}^{o^\perp}$, pour former $(?\,\psi_{l'+1})^{?\,o^\perp}$, on en fait un contexte séparant, $\mathcal{C}$, pour $\tilde{\pi}^{\vec{\phi}}/\tilde{\pi'}^{\vec{\phi}}$ et donc pour $\pi^{\vec{\phi}}/\pi'^{\vec{\phi}}$. Donc pour tout contexte de tête $\vec{\phi}$ associé à des entiers distincts suffisamment grands $n_1, \ldots, n_k$, on peut trouver un contexte de tête séparant $\pi^{\vec{\phi}}$ et $\pi'^{\vec{\phi}}$.

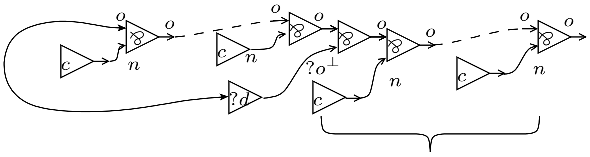
(b) *Si la conclusion positive est de type* $o^\perp$, on étudie la forme des arbres tenseur enracinés en x et on compare les arbres, en prouvant la séparation par induction sur le nombre de $\otimes^{s^\perp}$ dans l'arbre. Le premier cas est celui où les deux arbres tenseurs contiennent $\otimes^{s^\perp}$:

$\pi =$  $\pi' =$ 

Si $\pi_0 \not\sim \pi'_0$, alors on sépare ${\pi_0}^{\vec{\phi}}$ et ${\pi'_0}^{\vec{\phi}}$ avec $\mathcal{C} =< []|\psi_0^o, \psi_1^{n^\perp}, \psi_{l'}^{n^\perp} >$ et en échangeant ψ_0 en , dans $\mathcal{C}$, on obtient le contexte de tête $\mathcal{C}'$ qui permet de séparer $\pi^{\vec{\phi}}$ et $\pi'^{\vec{\phi}}$.

Si $\pi_0 \sim \pi'_0$, alors puisque $\pi \not\sim \pi'$, de deux choses l'une : soit (α) il y a $i \leq k = l$ tel que $\pi_i \not\sim \pi'_i$, soit (β) $k \neq l$, soit (γ) les fils typés par s atteignent deux conclusions différentes de I. Dans tous les cas, on peut séparer :

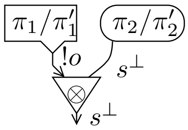
– dans le cas (α), on utilise l'induction sur les sous-réseaux π_i/π'_i, obtenant un contexte séparant $\mathcal{C} =< []|\psi_0^{o^\perp}, \psi_1^{n^\perp}, \psi_{l'}^{n^\perp} >$ on coupe $\psi_0^{o^\perp}$ avec la conclusion de type o de π/π' (qui est celle de π_0/π'_0) et on coupe la conclusion de π/π' de type $o^\perp$ avec un réseau qui est un arbre $\parr$ dual de l'arbre de tenseur de π/π' (qui sont identiques) tel que tous les ports auxiliaires de $\parr^?$ et $\parr^s$ sont des affaiblissements, sauf celui qui rencontrera π_i/π'_i qui est une déréliction dont le port auxiliaire est connecté au port auxiliaire de type o du plus haut $\parr$, comme ci-dessous :



– dans les cas (β) et (γ), on fait une expansion au niveau des conclusions de type s, puis une réduction associative et on peut ensuite appliquer la méthode du premier cas.

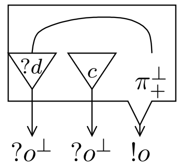
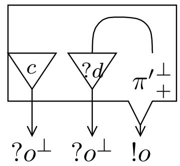
Les cas où l'un des deux arbres au moins contient des cellules $\otimes^!$ ou bien où il y a un lien axiome mais aucune cellule $\otimes$ sont traités de manière très similaire.

(c) *Si la conclusion positive est de type* $s^\perp$, on procède essentiellement comme dans le cas précédent par induction sur le nombre de $\otimes^{!s^\perp}$ des arbres tenseur, et par cas sur les sous-réseaux impliqués dans le fait que π et π' ne sont pas équivalents :



2. **S'il y a une déréliction en profondeur** 0 **et que** $x_i, x_j \notin J$. On sait que x_i et x_j sont de type $?\,o^\perp$ (en effet, on a supposé qu'il n'y a pas de cellule $\parr$ en profondeur 0).

   (a) *Si* $i \neq j$, on définit deux contextes ψ_i et ψ_j comme suit :

$\psi_i =$  $\psi_j =$ 

   où $\pi_+^\perp$ (resp. ${\pi'}_+^\perp$) est un arbre $\parr$ dual de l'arbre tenseur composant le sous-réseau positif maximal de π (resp. de π') où tous les ports auxiliaires restant sont affaiblis. Ce sous-réseau $\pi_+^\perp$ joue le rôle d'un effaceur de réseaux et est similaire au sous-réseau construit au cas $1.b.\alpha$.

   Alors on a $< \pi^{\vec{\phi}} | \psi_i, \psi_j > \rightsquigarrow^\star_{SANE} \mathbf{0}$ et $< {\pi'}^{\vec{\phi}} | \psi_i, \psi_j > \rightsquigarrow^\star_{SANE} \mathbf{1}$.

   (b) *Si* $i = j$, alors nous sommes essentiellement dans le même cas qu'avec une conclusion positive, bien qu'il y ait une légère subtilité parce que la conclusion x_i est maintenant de type $?\,o^\perp$ et qu'elle peut donc être contractée. Cette contraction pourrait interférer avec une étape d'induction et pour résoudre ce problème, on doit utiliser les réseaux ϕ^n de la définition 7.49.

   Soient p, q les nombres de $\otimes^{s^\perp}$ dans les arbres tenseurs de π et π' respectivement. On coupe la conclusion x_i de π/π' avec ϕ^n pour $n > p, q$ (et n étant distinct de tous les indices dans $\mathcal{N}$). On a alors la réduction représentée en figure 7.11.

   Le réseau représenté comme équivalent à $< \pi^{\vec{\phi}} | \phi^n >$ est en effet équivalent puisqu'ils se réduisent tous les deux vers le même réseau quand on réduit la coupure $\mathcal{S}/c$. On obtient $\tilde{\pi}^{\vec{\phi}, \phi^n}$ en réduisant la coupure sur la conclusion dérélictée x_i'' dans la figure ci-dessus. $\tilde{\pi}^{\vec{\phi}, \phi^n}$ et $\tilde{\pi'}^{\vec{\phi}, \phi^n}$ sont des réseaux $klmn$.

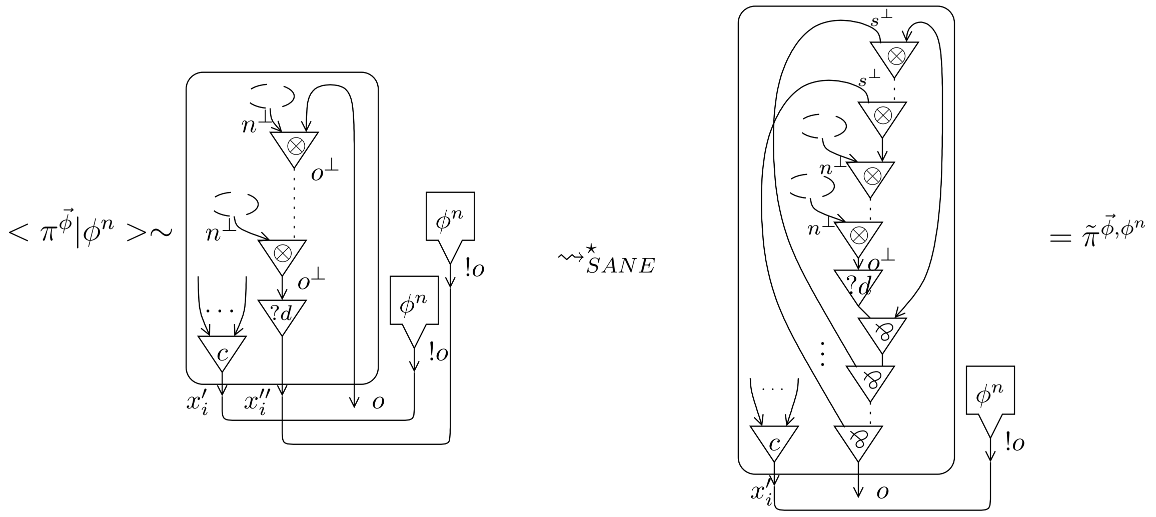
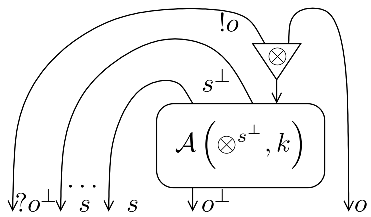


FIG. 7.11 – Réduction de $< \pi^{\vec{\phi}} | \phi^n >$.

Si $p \neq q$, on peut appliquer le lemme 7.56 puisque $k = n - p$ et $k' = n - q$, de manière à avoir la séparation. Sinon, on coupe la conclusion de type o avec un réseau de la forme suivante :



Après réduction de la coupure, on obtient un réseau avec une dérélic-tion comme cellule conclusion (il n'y a plus de contraction). Si on considère les réseaux sans les dérélictions, alors on a des réseaux qui ont une conclusion positive, qui ont la même profondeur que les réseaux originaux et J est maintenant devenu $J \cup x'_i$ (le contexte de tête $\vec{\phi}$ est remplacé par $\vec{\phi}, \phi^n$). Ces réseaux peuvent être séparés par un contexte de tête $\mathcal{C} =< [] | \psi_0, \psi_1, \ldots, \psi_{l'} >$.
Soit ψ_0^o l'élément du contexte séparant qui est coupé avec la conclusion de type $o^\perp$. ψ_0^o a une conclusion de type o, ses autres conclusions étant de type n ; en conséquence ψ_0^o peut être placé dans une boîte promotion pour obtenir un réseau $\psi_0'^{!o}$ qui fournit un contexte séparant pour $\pi^{\vec{\phi}} / \pi'^{\vec{\phi}}$.

3. **Finalement, s'il y a une déréliction en profondeur** 0 **et si** $\{x_i, x_j\} \cap J \neq \emptyset$. Soient k, k' les nombres de $\otimes^{s^\perp}$ dans les arbres tenseur maximaux de π / π'.
   (a) *si* $i \neq j$, il y a alors deux cas : (α) soit $x_i \in J$ et $x_j \notin J$, (β) soit $\{x_i, x_j\} \subseteq J$. On considère alors seulement le cas où ils sont tous les deux dans J puisque l'autre cas peut être traité de manière similaire comme on le verra par la suite.
   Par hypothèse, on sait qu'on a $n_i \neq n_j$ et que l'on peut choisir n_i et n_j aussi grands que l'on veut. Considérons donc $n_i > k$ et $n_j > k'$. $\pi^{\vec{\phi}}$ et $\pi'^{\vec{\phi}}$ ont les structures des réseaux présentés en figure 7.12.

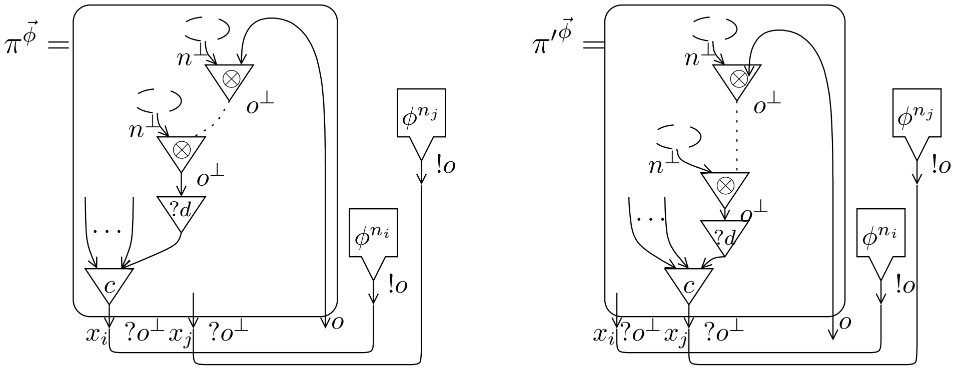
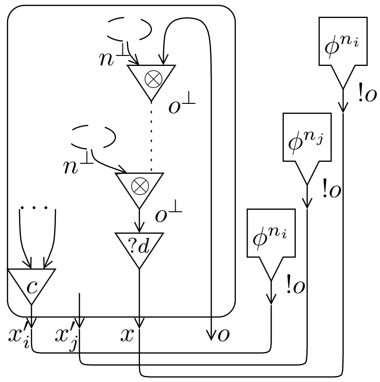


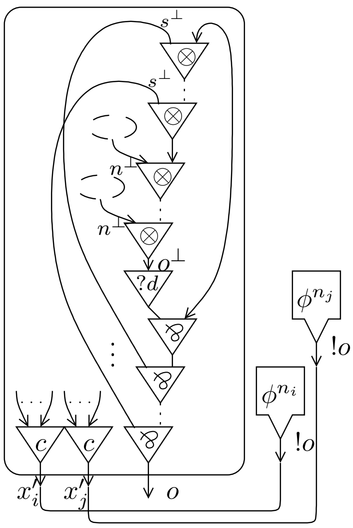
FIG. 7.12 – Réseaux $\pi^{\vec{\phi}}$ et $\pi'^{\vec{\phi}}$ quand $i \neq j$ et $\{x_i, x_j\} \subseteq J$.

Et on a donc, en renommant x_j en x'_j et x_i en x'_i ou en x dans $\tilde{\pi}^{\vec{\phi}}$ (on fait la même chose avec $\pi'^{\vec{\phi}}$: x_i devient x'_i et x_j devient x'_j ou x dans $\tilde{\pi}'^{\vec{\phi}}$) :

$\pi^{\vec{\phi}} \sim$



$\rightsquigarrow^{\star}_{SANE}$



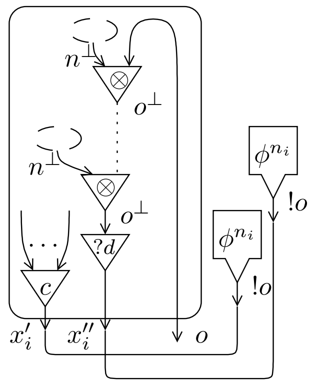
$= \tilde{\pi}^{\vec{\phi}}$

Si $k = k'$ alors $n_i - k \neq n_j - k'$ de sorte que le lemme 7.56 peut être appliqué et si $k \neq k'$ alors le lemme peut aussi être appliqué puisque $m = n_i$ et $m' = n_j$ de telle sorte que l'on peut trouver un contexte séparant.
Le cas où $x_j \in J$ et $x_i \notin J$ est traité de manière similaire : dans ce cas, on peut choisir ce avec quoi on veut couper x_i et en particulier on peut choisir

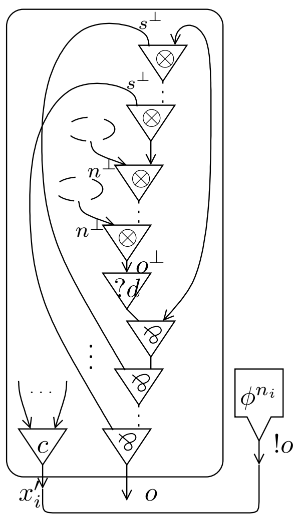
un ϕ^{n_i}.

(b) *si* $\{x_i, x_j\} \subseteq J$ *et* $i = j$. Dans ce cas, considérons $n_i > k, k'$. On a alors :

$\pi^{\vec{\phi}} \sim$



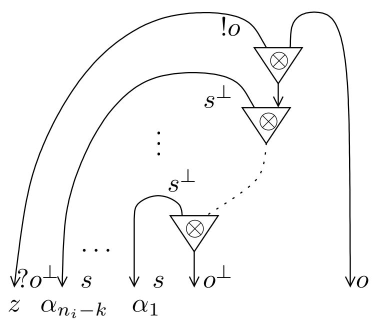
$\rightsquigarrow^{\star}_{SANE}$



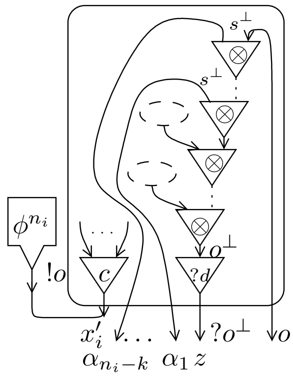
$= \tilde{\pi}^{\vec{\phi}}$

Si $k \neq k'$ on peut encore conclure grâce au lemme 7.56.

Si $k = k'$, alors il est possible de couper la conclusion de type o de $\tilde{\pi}^{\vec{\phi}}$ avec :



qui se réduit vers :



Ces réseaux $\tilde{\pi}_0/\tilde{\pi}'_0$ ont les mêmes profondeurs que π/π' et ont maintenant des conclusions dans $J \cup \{z^{?o^\perp}, \alpha_1^s, \dots, \alpha_{n_i-k}^s\}$ et une déréliction en profondeur 0 avec $x_{\tilde{\pi}_0} = x_{\tilde{\pi}'_0} \notin J$, ce qui peut être traité par le second cas de la preuve, assurant ainsi qu'il existe un contexte séparant pour $\pi^{\vec{\phi}}/\pi'^{\vec{\phi}}$ et nous permettant de conclure la preuve.

□

7.3. Remarques conclusives

Les réseaux SANE ont été conçus pour permettre d'encoder le $\Lambda\mu$-calcul dans un formalisme de réseaux permettant une analyse fine et très locale des réductions. L'un des objectifs de ce chapitre était également d'étudier la propriété de séparation dans un tel système.

Les réseaux que nous avons présentés sont originaux d'une part parce qu'ils empruntent des éléments aux réseaux MELL usuels et d'autres aux réseaux polarisés de Laurent, mais également, et surtout, du fait de la règle d'associativité. Nous avons ainsi :

i) les principales propriétés d'un système de réseaux (notion de correction, stabilité de la correction par réduction, confluence de la réduction, normalisation forte de certains sous-systèmes de réduction),
ii) un théorème de séquentialisation des $\Lambda\mu$-réseaux et de simulation de $\Lambda\mu$ (avec règles d'expansion pour η_T et η_S) dans les SANE et
iii) un théorème de séparation des réseaux corrects (et non pas seulement des $\Lambda\mu$-réseaux).

Certains éléments tendent à montrer que si les SANE peuvent simuler $\Lambda\mu$, il s'agit par ailleurs d'un système plus riche que $\Lambda\mu$:

– le résultat de confluence de la réduction de SANE vaut sans avoir besoin d'imposer un équivalent de l'hypothèse de μ-clôture nécessaire en $\Lambda\mu$-calcul (c'est-à-dire, en bref, que nos réseaux sont confluents même s'ils ont des conclusions de type $s/s^\perp$). Ceci est dû au fait que les réductions de SANE sont localisées et qu'une *fst*-réduction peut être initiée au niveau de l'équivalent d'un pré-redex de type $(\mathcal{T})\mathcal{S}$, de la forme $(\lambda x.t)\alpha$, et n'a pas besoin d'être initié au niveau de son μ-lieur ;
– l'encodage de $\Lambda\mu$ dans SANE ne fait jamais intervenir de cellule de type $⅋^{?s}$ ou $\otimes^{!s^\perp}$ (et d'ailleurs la définition des $\Lambda\mu$-réseaux 7.38 les exclut expressément). Ces cellules n'interviennent qu'au cours de la réduction d'un $\Lambda\mu$-réseau, au fur et à mesure des utilisations de règles d'associativité $\rightsquigarrow_a$: ces structures $⅋^{?s}$ et $\otimes^{!s^\perp}$ n'interviennent que dans la dynamique de la simulation de $\Lambda\mu$et non pas dans l'encodage lui-même.

Un autre calcul sous-jacent ? On a pourtant tout une classe de réseaux contenant des cellules $\parr^{?s}$ et $\otimes^{!s^\perp}$ ce qui suggère qu'un second calcul serait sous-jacent aux réseaux SANE, un calcul où les $\parr^{?}$ seraient abandonnés au profit de cellules $\parr^{?s}$. Ce calcul serait un calcul ayant pour construction principale la construction de stream comme nous l'avons déjà évoqué à la fin du chapitre sur la séparation. les constructions $\parr^{?s}$ et $\otimes^{!s^\perp}$ seraient des constructions d'assemblage de streams : $[x \cdot \alpha]$ (prête à être abstraite) et $[t \cdot \mathcal{S}]$ (prête à être appliquée), l'abstraction se réalisant avec les cellules $\parr^{s}$ (de la forme $\Lambda[x \cdot \alpha]$) et l'application par les cellules $\otimes^{s^\perp}$ (de la forme $(t)[u \cdot \mathcal{S}]$). Nous développerons cette approche dans le prochain chapitre.

Chapitre 8

$\lambda\mu$-calculs, streams et contrôle

Résumé:

Dans ce dernier chapitre concernant le $\Lambda\mu$-calcul, nous avons regroupé divers résultats concernant le $\Lambda\mu$-calcul et plus largement les $\lambda\mu$-calculs, résultats qui utilisent ou développent le contenu des chapitres précédents ou apportent des réponses à des questions soulevées précédemment.

Nous commencerons par comparer les diverses versions de $\lambda\mu$-calculs en appel par nom que nous avons rencontrées dans cette thèse ; $\lambda\mu$, $\lambda\mu\eta$, $\lambda\mu\epsilon$ et $\Lambda\mu$. Nous développerons également des approfondissements de l'interprétation de streams du $\Lambda\mu$-calcul et du parallèle avec le contrôle délimité via $\lambda\mu\widehat{\mathtt{tp}}$, en particulier via la définition d'une machine abstraite pour $\Lambda\mu$, la $\Lambda\mu$-KAM, et d'une variante du λ-calcul qui possède une construction de streams, le $\Lambda\mathcal{S}$-calcul.

Références. Les résultats comparant les divers $\lambda\mu$-calculs ont été publiés dans l'article *On the relations between the syntactic theories of $\lambda\mu$-calculi* [Sau08a].

Sommaire

8.1. Introduction

Dans ce dernier chapitre concernant le $\Lambda\mu$-calcul, nous avons regroupé divers résultats concernant le $\Lambda\mu$-calcul et plus largement les $\lambda\mu$-calculs. Ces résultats utilisent ou développent le contenu des chapitres précédents ou apportent des réponses à des questions soulevées précédemment. Il s'agit principalement de conclure cette partie en resituant le calcul que nous avons développé dans les chapitres précédents parmi les autres $\lambda\mu$-calculs et de faire le bilan de l'interprétation de streams et des liens entretenus avec les opérateurs de contrôle en présentant quelques résultats supplémentaires reliant $\Lambda\mu$ à $\lambda\mu\widehat{\mathtt{tp}}$. En outre, nous sommes restés jusqu'ici au niveau de l'analogie ou de la notation concernant les streams ; nous présentons ici des éléments plus formels à l'appui de cette interprétation : une machine abstraite, la $\Lambda\mu$-KAM, une sémantique dénotationnelle proposée par Thomas Streicher et un calcul de streams issu de l'analyse menée via les réseaux SANE.

Divers $\lambda\mu$-calculs. Nous commencerons par comparer les diverses versions de $\lambda\mu$-calculs en appel par nom que nous avons rencontrées dans cette thèse ; $\lambda\mu$, $\lambda\mu\eta$, $\lambda\mu\epsilon$ et $\Lambda\mu$. Nous établirons des résultats comparant les théories équationnelles de ces calculs et étudierons de manière un peu plus détaillée le $\lambda\mu\epsilon$-calcul de Ph. de Groote sur lequel nous ne nous sommes pas encore penché en détail.

Streams et contrôle. Nous approfondirons ensuite l'interprétation de streams du $\Lambda\mu$-calcul et du parallèle avec le contrôle délimité via $\lambda\mu\widehat{\mathtt{tp}}$ en proposant d'abord quelques exemples de $\Lambda\mu$-termes dont le comportement opérationnel est intéressant du point de vue de l'interprétation de streams, nous définirons ensuite une machine abstraite pour $\Lambda\mu$ ainsi qu'une sémantique dénotationnelle pour $\Lambda\mu$ proposée par Streicher. Nous proposons une extension du calcul $\Lambda\mu$ à une hiérarchie de calculs qui semblent liés aux hiérarchies d'opérateurs $\mathtt{shift}_i/\mathtt{reset}_i$ de Danvy et Filinski [DF90] et nous terminerons en proposant un calcul de stream, $\Lambda\mathcal{S}$, dont les constructions principales sont une abstraction et une application de stream et qui a déjà été brièvement discuté aux chapitres où nous avons étudié le théorème de Böhm pour $\Lambda\mu$ et au chapitre sur les réseaux SANE.

8.2. Comparaison des $\lambda\mu$-calculs

8.2.1 Diverses variantes de $\lambda\mu$-calculs

On trouve dans la littérature plusieurs variantes de $\lambda\mu$-calculs. En s'en tenant aux $\lambda\mu$-calculs en appel par nom (sans considérer les systèmes en appel par valeur comme ceux introduits pas Ong et Stewart [OS97]), on a déjà vu au cours des précédents chapitres le $\lambda\mu$-calcul tel que présenté par Parigot [Par92], le $\lambda\mu\eta$-calcul introduit par Py pour étudier la confluence et la propriété de séparation [Py98, DP01a], le $\Lambda\mu$-calcul introduit par nos soins pour retrouver la séparation et dont cette thèse tente de mener une étude approfondie [Sau05, Sau07, Sau08a, PS08] et dont un sous-calcul très proche avait déjà été considéré par de Groote dans un article à propos des traductions CPS du $\lambda\mu$-calcul [dG94], ainsi qu'une autre variante avec une nouvelle réduction ϵ également due à de Groote et que nous avons brièvement mentionnée [dG98] ainsi que le $\lambda\mu\widehat{\mathtt{tp}}$-calcul de Herbelin et al. [HG08]. Si on met de côté le $\lambda\mu\widehat{\mathtt{tp}}$-calcul qui a été introduit explicitement comme une variante du $\lambda\mu$-calcul avec une variable de continuation dynamiquement liée de manière à observer de nouveaux effets calculatoires, les calculs précédents ont souvent été considérés comme étant

le $\lambda\mu$-calcul et les points communs et les divergences entre ces calculs n'ont que rarement été soulevés et n'ont pas très souvent été tirés au clair. C'est ainsi qu'il arrive que des auteurs utilisent diverses présentations du $\lambda\mu$-calcul dans différents travaux, et on ne sait plus forcément très bien quelle propriété est valable dans quelle «version de $\lambda\mu$».

Pourtant, ces calculs différents certainement ; le fait que la propriété de séparation soit valable en $\Lambda\mu$ mais pas dans $\lambda\mu$ ni $\lambda\mu\eta$ en est peut-être la meilleure illustration (les liens entre $\Lambda\mu$ et $\lambda\mu\widehat{\mathsf{tp}}$ en sont une autre).

Il nous semble important d'une part de souligner le fait qu'il y a donc ***des*** $\lambda\mu$-calculs et non pas un $\lambda\mu$-calcul et d'autre part de mettre en évidence les relations que ces calculs entretiennent les uns avec les autres. C'est ce que nous allons tenter de faire dans cette section.

Nous limiterons notre analyse aux quatre calculs précédemment cités : $\lambda\mu$, $\lambda\mu\eta$, $\Lambda\mu$ et $\lambda\mu\epsilon$. Avant de commencer, il nous faut définir $\lambda\mu\epsilon$ puisque ce calcul n'a été qu'évoqué jusqu'ici.

8.2.2 La syntaxe de Ph. de Groote avec ϵ-réduction : $\lambda\mu\epsilon$

Philippe de Groote a introduit une machine abstraite pour le $\lambda\mu$-calcul à la fin des années 90 [dG98], de manière contemporaine d'autres auteurs [Bie98, SR98][1]. Pour cela, il a considéré une extension du $\lambda\mu$-calcul de Parigot, le $\lambda\mu\epsilon$-calcul.

Les termes du $\lambda\mu\epsilon$-calcul sont les termes de $\Sigma_{\Lambda\mu}$, mais le $\lambda\mu\epsilon$-calcul contient une nouvelle règle, la ϵ-reduction :

$$\boxed{\mu\alpha.\mu\beta.t \rightarrow_\epsilon \mu\alpha.\left|t\right|_\beta}$$

où $|t|_\beta$ est le terme obtenu en effaçant de t toutes les occurrence libres de β dans t. Cette règle était justifiée, du point de vue du typage, par le fait que dans le terme $\mu\alpha.\mu\beta.t$, le sous-terme $\mu\beta.t$ doit recevoir le type $\perp$.

Une contrainte sur la réduction ρ doit être ajoutée de manière à ne pas perdre la confluence : la ρ-réduction est maintenant :

$$\mu\gamma.(\mu\alpha.t)\beta \longrightarrow_\rho \mu\gamma.t\left\{\beta/\alpha\right\}.$$

Sans cette contrainte, il y aurait un problème de confluence pour le terme $(\mu\gamma.\mu\beta.\mu\alpha.t)\delta\zeta$ selon que l'on applique ϵ à $\mu\alpha$ ou à $\mu\beta$:

$$|t|_\alpha\left\{\delta/\gamma\right\}\left\{\zeta/\beta\right\} \longleftarrow_\rho^2 \longleftarrow_\epsilon (\mu\gamma.\mu\beta.\mu\alpha.t)\delta\zeta \longrightarrow_\epsilon \longrightarrow_\rho^2 |t|_\beta\left\{\delta/\gamma\right\}\left\{\zeta/\alpha\right\}$$

En outre, on ne peut pas ajouter la η-réduction à $\lambda\mu\epsilon$ sans quoi la confluence est également perdue[2] :

$$\mu\alpha.\lambda x.\mu\beta.y \longleftarrow_\mu \mu\alpha.\lambda x.(\mu\beta.y)x \longrightarrow_\eta \mu\alpha.\mu\beta.y \longrightarrow_\epsilon \mu\alpha.y$$

On notera que cet exemple montre par ailleurs que l'ajout de la *fst*-réduction (ou ν-réduction dans la terminologie de Py) n'arrangerait rien à ce problème, au contraire, il y aurait vraiment une paire critique non confluente *fst*/ϵ sur le terme $\mu\alpha.\mu\beta.y$. Il en est de même avec l'η-expansion qui d'habitude se comporte pourtant bien vis-à-vis de la confluence.

[1] On pourra se référer à la section 2.3.4 du chapitre 2 pour des détails à ce sujet.

[2] On trouvera de nombreux détails sur les problèmes de confluence dans $\lambda\mu\epsilon$ avec η dans la thèse de doctorat de Py [Py98].

On peut maintenant définir la réduction de $\lambda\mu\epsilon$:

Définition 8.1 (*Réduction du $\lambda\mu\epsilon$-calcul*)

La réduction du $\lambda\mu\epsilon$-calcul, notée $\longrightarrow_{\lambda\mu\epsilon}$, est définie par les cinq règles de réduction suivantes :

$$\begin{array}{llll}
(\lambda x.t)u & \longrightarrow_\beta & t\{u/x\} & \\
(\mu\alpha.t)u & \longrightarrow_\mu & \mu\alpha.t\{(v)u\alpha/(v)\alpha\} & \\
\mu\gamma.(\mu\alpha.t)\beta & \longrightarrow_\rho & \mu\gamma.t\{\beta/\alpha\} & \\
\mu\alpha.(t)\alpha & \longrightarrow_\theta & t & \text{si } \alpha \notin FV(t) \\
\mu\alpha.\mu\beta.t & \longrightarrow_\epsilon & \mu\alpha.\,|t|_\beta &
\end{array}$$

où $|t|_\beta$ consiste à effacer toutes les occurrences libres de β dans t et est défini comme suit :

Définition 8.2 ($|t|_\beta$)

$|t|_\beta$ est défini comme suit :

$$\begin{array}{lll}
|x|_\beta & \triangleq & x \\
|\lambda x.t|_\beta & \triangleq & \lambda x.\,|t|_\beta \\
|(t)u|_\beta & \triangleq & (|t|_\beta)\,|u|_\beta \\
|\mu\beta.t|_\beta & \triangleq & \mu\beta.t \\
|\mu\alpha.t|_\beta & \triangleq & \mu\alpha.\,|t|_\beta \qquad \text{si } \alpha \neq \beta \\
|(t)\beta|_\beta & \triangleq & |t|_\beta \\
|(t)\alpha|_\beta & \triangleq & (|t|_\beta)\alpha \qquad \text{si } \alpha \neq \beta
\end{array}$$

Py [Py98] a prouvé le confluence de $\lambda\mu\epsilon$.

8.2.3 Comparaison des théories équationnelles des $\lambda\mu$-calculs

On compare rapidement les théories équationnelles de $\lambda\mu$, $\lambda\mu\eta$, $\lambda\mu\epsilon$ et $\Lambda\mu$ dans cette section.

Stabilité de $\lambda\mu/\lambda\mu\eta$ par $\lambda\mu\epsilon/\Lambda\mu$. On commence par établir le lemme suivant, dont une partie a déjà été présentée au chapitre 5, qui énonce la stabilité de $\Sigma_{\lambda\mu}$ ainsi que celle des réductions dans $\lambda\mu$ et $\lambda\mu\eta$.

Lemme 8.3

Le $\lambda\mu$-calcul est stable pour la $\Lambda\mu$-réduction et la $\lambda\mu\epsilon$-réduction :
- *si $t \in \Sigma_{\lambda\mu}$ et $t \longrightarrow^\star_{\Lambda\mu} u$, alors $u \in \Sigma_{\lambda\mu}$ et $t \longrightarrow^\star_{\lambda\mu\eta} u$;*
- *si $t \in \Sigma_{\lambda\mu}$ et $t \longrightarrow^\star_{\lambda\mu\epsilon} u$, alors $u \in \Sigma_{\lambda\mu}$ et $t \longrightarrow^\star_{\lambda\mu} u$.*

Démonstration : La preuve du lemme est immédiate : de chaque réduction de $\lambda\mu\epsilon$ (resp. $\Lambda\mu$) à partir d'un terme de $\Sigma_{\lambda\mu}$, on obtient un terme de $\Sigma_{\lambda\mu}$ et l'instance de réduction considérée peut être simulée dans $\lambda\mu$ (resp. $\lambda\mu\eta$). On pourra se reporter au lemme 5.28.

□

Extensions conservatives. Le $\Lambda\mu$-calcul est une extension du $\lambda\mu\eta$-calcul. On montre qu'il s'agit d'une extension conservative quand on considère les termes de $\Sigma^c_{\lambda\mu}$ pour $\Lambda\mu$ et n'importe quel terme de $\Sigma_{\lambda\mu}$ pour $\lambda\mu\epsilon$: si $t, u \in \Sigma^c_{\lambda\mu}$ (resp. $t, u \in \Sigma_{\lambda\mu}$), une égalité entre t et u est vraie dans $\lambda\mu\eta$ (resp. $\lambda\mu$) si, et seulement si, elle est vraie dans $\Lambda\mu$ (resp. dans $\lambda\mu\epsilon$) :

Proposition 8.4 ($\Lambda\mu$ *est une extension conservative de* $\lambda\mu\eta$)

Le $\Lambda\mu$-calcul est une **extension conservative** *du $\lambda\mu\eta$-calcul sur les termes clos : Soient $t, u \in \Sigma^c_{\lambda\mu}$, on a l'équivalence suivante : $t =_{\Lambda\mu} u$ si, et seulement si $t =_{\lambda\mu\eta} u$.*

Démonstration :

$\boxed{\Leftarrow}$ Cette implication est évidente puisque toute égalité dans $\lambda\mu$ est également une égalité dans $\Lambda\mu$.

$\boxed{\Rightarrow}$ Supposons $t =_{\Lambda\mu} u$. D'après le théorème 5.26, on sait qu'il existe $v \in \Sigma_{\Lambda\mu}$ tel que $t, u \longrightarrow^\star_{\Lambda\mu} v$. D'après le lemme 5.28, $v \in \Sigma_{\lambda\mu}$ et $t, u \longrightarrow^\star_{\lambda\mu} v$, ce qui entraîne immédiatement $t =_{\lambda\mu\eta} u$.

□

Proposition 8.5 ($\lambda\mu\epsilon$ *est une extension conservative de* $\lambda\mu$)

Si $t, u \in \Sigma_{\lambda\mu}$, alors $t =_{\lambda\mu\epsilon} u \Leftrightarrow t =_{\lambda\mu} u$.

Démonstration : La preuve est similaire à la preuve précédente si ce n'est que l'hypothèse de clôture est maintenant superflue du fait de la confluence en $\lambda\mu/\lambda\mu\epsilon$ qui ne requiert pas la μ-clôture.

□

Incomparabilité de $=_{\Lambda\mu}$ et $=_{\lambda\mu\epsilon}$.

Proposition 8.6 ($=_{\Lambda\mu}$ *et* $=_{\lambda\mu\epsilon}$ *sont incomparables.*)

Il existe $t, u, v \in \Sigma^c_{\Lambda\mu}$ tels que $t =_{\Lambda\mu} u$ mais $t \neq_{\lambda\mu\epsilon} u$ et tels que $t =_{\lambda\mu\epsilon} v$ mais $t \neq_{\Lambda\mu} v$.

Démonstration : Soit $t \in \Sigma^c_{\Lambda\mu}$ un terme de la forme $\mu\alpha.t'$, qui soit en forme normale canonique pour $\Lambda\mu$ et qui ne possède aucune paire de μ-abstractions consécutives. Soient alors $u = \lambda x.\mu\alpha.t' \{(w)x\alpha/(w)\alpha\}$ et $v = \mu\alpha.\mu\beta.t'$. On a $t =_{fst} u$ et $t =_\epsilon v$ de telle sorte que $t =_{\Lambda\mu} u$ et $t =_{\lambda\mu\epsilon} v$.

Par ailleurs, t est une $\lambda\mu\epsilon$-forme normale puisqu'il n'a pas de ϵ-redex puisque aucun sous-terme de la forme $\mu\alpha.\mu\beta.u$ et qu'étant en forme normale canonique il n'a donc ni β, ni μ, ni θ, ni ρ redex[3] et il en est de même de u (la variable x ne peut pas créer de $\lambda\mu\epsilon$-redex sans contredire le fait que t est une forme normale canonique) : il s'agit donc de deux formes normales distinctes du $\lambda\mu\epsilon$-calcul et par confluence de $\lambda\mu\epsilon$, $t \neq_{\lambda\mu\epsilon} u$.

En outre, t et v sont en forme normale canonique de telle sorte que $t =_{\Lambda\mu} v$ si, et seulement si, $t =_{fst} v$. Mais t et v contiennent un nombre différent de μ-abstractions alors que la *fst*-réduction préserve le nombre de μ dans un terme. En conclusion $t \neq_{fst} v$ et finalement on a $t \neq_{\Lambda\mu} v$.

On notera qu'on peut choisir, par exemple, $t = \mu\alpha.\lambda x.x$.

□

8.2.4 $\lambda\mu\epsilon$ et séparation

On sait ce qu'il en est de la séparation dans $\lambda\mu$, $\lambda\mu\eta$ et $\Lambda\mu$. Que peut-on dire de la séparabilité dans $\lambda\mu\epsilon$? On esquisse ici quelques pistes à ce sujet.

[3] Que ce soit pour la ρ-réduction de $\Lambda\mu$ ou de $\lambda\mu\epsilon$.

	Confluence	Séparation	Système de types	Subject reduction	SN
$\lambda\mu$	OUI	NON	CND	OUI	OUI
$\lambda\mu\eta$	OUI (μ-clos)	NON	CND + $\perp$	OUI	OUI
$\lambda\mu\epsilon$	OUI	?	CND + $\perp$	OUI	OUI
$\Lambda\mu$	OUI (μ-clos)	OUI	CND + $\perp$ et $\Lambda_{\mathcal{S}}$	OUI	OUI

FIG. 8.1 – Propriétés des quatre $\lambda\mu$-calculs.

Énoncer la séparation dans le $\lambda\mu\epsilon$-calcul nécessiterait de considérer les classes d'équivalence modulo les $\lambda\mu\epsilon$-régles plus η. Cette étude est complexe puisque $\lambda\mu\epsilon+\eta$ n'est pas confluent et qu'il est donc difficile de dire quoi que ce soit à propos du fait que deux termes ne soient pas égalisés par la théorie équationnelle.

On considérera simplement un exemple de deux termes qui ne sont pas séparables alors qu'ils ne sont pas équationnellement équivalents dans $\lambda\mu\epsilon$: $t_0 = \mu\alpha.0$ et $t_1 = \mu\alpha.1$ ne peuvent être séparés par aucun contexte :

$$
\begin{array}{llll}
(\mu\alpha.t)u & \longrightarrow_\mu & \mu\alpha.t & \text{si } \alpha \notin FV(t). \\
\mu\gamma.(\mu\alpha.t)\beta & \longrightarrow_\rho & \mu\gamma.t & \text{qui est } \alpha\text{-équivalent à } \mu\alpha.t. \\
\mu\beta.\mu\alpha.t & \longrightarrow_\epsilon & \mu\beta.\left|t\right|_\alpha = \mu\beta.t & \text{qui est } \alpha\text{-équivalent à } \mu\alpha.t.
\end{array}
$$

Les autres contextes possibles $(\lambda x.[], \lambda x.([])\beta, \mu\beta.\lambda x.[], \dots)$ n'interagissent pas avec les termes et donc a fortiori ne les distinguent pas. En fait, tout terme de la forme $\mu\alpha.t$ avec t un terme clos est observationnellement équivalent à t_0 : aucun contexte ne peut faire se réduire l'un sur une forme normale de tête sans que l'autre ne se réduise également sur une forme normale de tête.

Ceci ne clôt pas la question de la séparation dans $\lambda\mu\epsilon$: il n'est pas tout à fait exclu que l'ajout de la règle η égalise tous les termes de ce type (en tous les cas nous n'avons pas d'argument pour l'exclure en l'absence de confluence)... même si c'est peu probable.

8.2.5 Bilan des propriétés des quatre $\lambda\mu$-calculs

Les quatre $\lambda\mu$-calculs considérés dans cette section partagent donc certaines propriétés et diffèrent sur d'autres points. Nous faisons ici un bref bilan des propriétés syntaxiques connues pour ces calculs en résumant dans la figure 8.1 les propriétés des quatre $\lambda\mu$-calculs. Les quatre calculs satisfont la confluence [Par92, Py98, Sau08a][4]. Ils ont tous les quatre un système de typage qui correspond aux preuves de la déduction naturelle classique, avec ou sans $\perp$ explicite et $\Lambda\mu$ dispose en outre d'un système de types supplémentaire que nous avons étudié au chapitre 6. La normalisation forte est connue dans le cas simple pour $\lambda\mu\eta$, $\lambda\mu\epsilon$ et $\Lambda\mu$ tandis que Parigot l'a prouvé pour le second ordre également [Par97] pour $\lambda\mu$.

8.3. Streams et contrôle

8.3.1 Retour sur l'interprétation de streams

Nous allons maintenant développer l'analogie qui a sous-tendu le développement de la théorie du $\Lambda\mu$-calcul, l'analogie entre la μ-abstraction et une abstraction de stream,

[4]Voir aussi le chapitre 5 de cette thèse.

c'est-à-dire une sorte de λ-abstraction infinitaire.

Jusqu'ici, l'interprétation de streams a été une intuition opérationnelle utile pour le développement de résultats à propos de $\Lambda\mu$, elle a également pu être une notation synthétique pratique, mais nous ne lui avons pas donné de statut formel, si ce n'est dans les types de $\Lambda_{\mathcal{S}}$ et dans la règle d'associativité des réseaux SANE. Nous allons dans cette section développer cette approche et la relier aux opérateurs de contrôle délimité. Après avoir considéré quelques $\Lambda\mu$-termes dont le comportement calculatoire est significatif pour l'interprétation de streams, nous proposons une machine abstraite pour $\Lambda\mu$, la $\Lambda\mu$-KAM, qui nous conduit à proposer une hiérarchie de $\Lambda\mu$-calculs ; nous faisons ensuite état d'une sémantique dénotationnelle proposée par Thomas Streicher pour le $\Lambda\mu$-calcul [Str05] dans une communication privée et nous proposons finalement un calcul de streams, $\Lambda\mathcal{S}$, qui est construit autour d'un constructeur de stream, d'une abstraction de stream et d'une application de stream.

Entrelacer des streams. Dans cette section on considère un exemple de ce qu'il est possible de faire en $\Lambda\mu$-calcul en utilisant un opérateur de point fixe.

On rappelle les combinateurs de point fixe usuels du λ-calcul :

Définition 8.7 (*Combinateurs de point fixe de Church et de Turing*)

- *Le combinateur de point fixe de Church, noté $Y_{\mathcal{C}}$ est :*

$$Y_{\mathcal{C}} = \lambda f.(\lambda x.(f)(x)x)\lambda x.(f)(x)x$$

- *Le combinateur de point fixe de Turing, noté $Y_{\mathcal{T}}$ est :*

$$Y_{\mathcal{T}} = (\lambda x.\lambda y.(y)(x)xy)\lambda x.\lambda y.(y)(x)xy$$

Proposition 8.8

Soit t un λ-terme (resp. un $\Lambda\mu$-terme). On a :

$$\begin{array}{llll} (Y_{\mathcal{C}})t & =_\beta & (t)(Y_{\mathcal{C}})t & (\text{resp. } (Y_{\mathcal{C}})t =_{\beta_T} (t)(Y_{\mathcal{C}})t.) \\ (Y_{\mathcal{T}})t & \longrightarrow_\beta & (t)(Y_{\mathcal{T}})t & (\text{resp. } (Y_{\mathcal{T}})t \longrightarrow_{\beta_T} (t)(Y_{\mathcal{T}})t.) \end{array}$$

Définition 8.9

On considère le $\Lambda\mu$-terme suivant :

$$t = \lambda z.\lambda x.\mu\alpha.\lambda y.\mu\beta.\mu\gamma.(z)\alpha\beta xy\gamma.$$

En notation de stream, cela donne :

$$t = \lambda z.\Lambda(x,\alpha).\Lambda(y,\beta).\Lambda\gamma.(z)\alpha\beta(x,y,\gamma).$$

Quand on prend le point fixe de ce terme, on obtient un terme qui prend deux streams et empile sur une troisième stream, en les entrelaçant, les contenus des deux premières streams :

$$(Y_{\mathcal{T}})t \longrightarrow^{\star}_{\Lambda\mu} \Lambda(x_1,\ldots,x_n,\alpha).\Lambda(y_1,\ldots,y_n,\beta).\Lambda\gamma.(Y_{\mathcal{T}})t\alpha\beta(x_1,y_1,\ldots,x_n,y_n,\gamma).$$

Remarque 8.10

On peut considérer une variante plus simple avec le $\Lambda\mu$-terme t' suivant :

$$t' = \lambda z.\lambda x.\mu\alpha.\mu\beta.\mu\gamma.(z)\beta\alpha x\gamma.$$

En notation de stream, cela donne :

$$t' = \lambda z.\Lambda(x,\alpha).\Lambda\beta.\Lambda\gamma.(z)\beta\alpha(x,\gamma).$$

Explorer des streams. On montre sur quelques exemples comment on peut explorer les streams grâce aux termes du $\Lambda\mu$-calcul.

Définition 8.11 (*Terme* `First`)

$$\texttt{First} \triangleq \mu\alpha.(\lambda x.\mu\beta.x)\alpha$$

Définition 8.12 (*Terme* `Tail`)

$$\texttt{Tail} \triangleq \lambda u.\mu\alpha.(\lambda x.\mu\beta.(u)\beta)\alpha$$

On peut ainsi définir un terme qui retourne le n^{e} argument d'une stream (en comptant à partir de 0) :

$$(([n])\texttt{Tail})\texttt{First}$$

où $[n]$ est l'entier de Church n donné par : $[n] \triangleq \lambda f.\lambda x.(f)^n x$.

On a en effet :

$$\begin{array}{rll} (([n])\texttt{Tail})\texttt{First} & \longrightarrow^{\star}_{\Lambda\mu} & \mu\alpha.(\lambda x_0 \ldots x_n.\mu\beta.x_n)\alpha \\ & \longrightarrow^{\star}_{\Lambda\mu} & \lambda x_0 \ldots x_n.\mu\alpha.x_n \end{array}$$

en conséquence de quoi ce terme retourne le n^{e} élément d'une stream lorsqu'il arrive en position de tête :

$$((([n])\texttt{Tail})\texttt{First})\mathcal{S} \longrightarrow^{\star}_{\Lambda\mu} \mathcal{S}(i)$$

8.3.2 La $\Lambda\mu$-KAM, une machine abstraite pour $\Lambda\mu$

On a présenté une machine abstraite pour le $\lambda\mu$-calcul dans le chapitre 2. La $\lambda\mu$-KAM traitait indifféremment $\lambda\mu$-termes et termes nommés avec une instruction dédiée à chaque construction. Pourtant, la $\lambda\mu$-KAM ne peut pas traiter l'ensemble des $\Lambda\mu$-termes. En effet, cette machine suppose qu'un terme nommé est dans un état où la pile est vide de même que la transition d'un terme de la forme $\mu\alpha.c$ laisse la pile vide.

Une machine abstraite pour le $\Lambda\mu$-calcul doit avoir des états plus structurés : pour cela on va considérer une pile supplémentaire qui sera aux variables de stream ce que la première pile était aux variables de terme.

λ-abstractions et λ-applications empilaient et dépilaient sur la pile K tandis que μ-abstractions et μ-applications empileront et dépileront de la nouvelle pile, S.

Comme constaté dans la définition du système de typage pour $\Lambda\mu$, il faut prévoir que les constructions de stream et de terme puissent interagir ; notre présentation de la machine en tiendra compte.

$$
\begin{array}{llrcllrl}
\langle x & [e] & [S]\rangle & \longrightarrow & \langle t & [e'] & [S]\rangle & \text{si } e(x) = t[e'] \\
\langle \lambda x.t & [e] & [u[e'] \cdot S]\rangle & \longrightarrow & \langle t & [x = u[e']; e] & [S]\rangle & \\
\langle (t)u & [e] & [S]\rangle & \longrightarrow & \langle t & [e] & [u[e] \cdot S]\rangle & \\
&&&&&&& \\
\langle \mu\alpha.t & [e] & [K :: S]\rangle & \longrightarrow & \langle t & [\alpha = K; e] & [S]\rangle & \\
\langle (t)\alpha & [e] & [S]\rangle & \longrightarrow & \langle t & [e] & [K :: S]\rangle & \text{si } e(\alpha) = K
\end{array}
$$

$$\text{avec } S = K_1 :: K_2 :: \ldots :: \Yup \text{ et } K = u_1[e_1] \cdot u_2[e_2] \cdot \cdots \cdot \perp.$$

FIG. 8.2 – Présentation alternative de la $\Lambda\mu$-KAM.

$$
\begin{array}{lllrcllllr}
\langle x & [e] & K & [S]\rangle & \longrightarrow & \langle t & [e'] & K & [S]\rangle & \text{si } e(x) = t[e'] \\
\langle \lambda x.t & [e] & u[e'] \cdot K & [S]\rangle & \longrightarrow & \langle t & [x = u[e']; e] & K & [S]\rangle & \\
\langle (t)u & [e] & K & [S]\rangle & \longrightarrow & \langle t & [e] & u[e] \cdot K & [S]\rangle & \\
&&&&&&&&& \\
\langle \mu\alpha.c & [e] & K & [S]\rangle & \longrightarrow & \langle c & [\alpha = K; e] & \perp & [S]\rangle & \\
\langle [\alpha]t & [e] & \perp & [S]\rangle & \longrightarrow & \langle t & [e] & K & [S]\rangle & \text{si } e(\alpha) = K \\
&&&&&&&&& \\
\langle \mu\widehat{\mathsf{tp}}.c & [e] & K & [S]\rangle & \longrightarrow & \langle c & [e] & \perp & [K \cdot S]\rangle & \\
\langle [\widehat{\mathsf{tp}}]t & [e] & \perp & [K \cdot S]\rangle & \longrightarrow & \langle t & [e] & K & [S]\rangle &
\end{array}
$$

FIG. 8.3 – Machine abstraite pour $\lambda\mu\widehat{\mathsf{tp}}$ [HG08].

Définition 8.13 (*États de la $\Lambda\mu$-KAM*)

les **états de la $\Lambda\mu$-KAM** *sont de la forme $\langle t, [e], K, S\rangle$ où :*
- *t est un $\Lambda\mu$-terme ;*
- *e est un environnement de variables de terme et de stream : les variables de terme sont liées à des clôtures $u[e]$ tandis que les variables de stream sont liées à des piles de clôtures K ;*
- *K est une pile de clôtures, la pile vide est notée $\perp$;*
- *S est une pile de piles de clôtures, la pile vide étant notée $\Yup$.*

Un **état initial** *de la $\Lambda\mu$-KAM est de la forme $\langle t, [\emptyset], \perp, \Yup\rangle$.*

Définition 8.14 (*Transitions de la $\Lambda\mu$-KAM*)

$$
\begin{array}{lllrcllllr}
\langle x & [e] & K & S\rangle & \longrightarrow & \langle t & [e'] & K & S\rangle & \textit{si } e(x) = t[e'] \\
\langle \lambda x.t & [e] & u[e'] \cdot K & S\rangle & \longrightarrow & \langle t & [x = u[e']; e] & K & S\rangle & \\
\langle (t)u & [e] & K & S\rangle & \longrightarrow & \langle t & [e] & u[e] \cdot K & S\rangle & \\
&&&&&&&&& \\
\langle \mu\alpha.t & [e] & K & L \cdot S\rangle & \longrightarrow & \langle t & [\alpha = K; e] & L & S\rangle & \\
\langle (t)\alpha & [e] & L & S\rangle & \longrightarrow & \langle t & [e] & K & L \cdot S\rangle & \textit{si } e(\alpha) = K
\end{array}
$$

Les deux dernières transitions effectuent deux opérations :

1. Assigner la pile K à α dans la clôture (resp. charge le K correspondant à α) et
2. Empiler une nouvelle pile dans S (resp. la dépiler de S).

Il y a deux manières d'analyser cela :

– On peut d'un côté considérer que cette double opération provient de l'interaction entre les constructions de terme et de stream, interaction déjà évoquée plus haut ; dans cette perspective, on pourrait aussi travailler avec simplement une pile de piles, les λ-instructions n'agissant qu'au niveau de la première pile tandis que les μ-instructions peuvent empiler et dépiler au second niveau, on obtiendrait alors des μ-instructions primitives (mais on retrouverait un phénomène proche de l'associativité des types de $\Lambda_{\mathcal{S}}$ ou des SANE). On pourrait alors présenter la machine de manière alternative à la manière de la figure 8.2.
– D'un autre côté, on pourrait chercher à rendre plus primitives ces instructions en le reflétant dans le calcul ; on serait alors dans une approche à la $\lambda\mu\widehat{\mathsf{tp}}$, comme présenté en figure 8.3 : la machine abstraite pour $\lambda\mu\widehat{\mathsf{tp}}$ (en appel par nom) décompose en effet les deux instructions composites de la $\Lambda\mu$-KAM grâce aux opérations effectuées par les $\mu\widehat{\mathsf{tp}}$ et $[\widehat{\mathsf{tp}}]$.

On remarquera que la machine $\Lambda\mu$-KAM a une structure d'états similaire à celle de la machine abstraite pour shift/reset de Danvy et al [BBD05] et que les transitions ont des structures proches.

Analyse des états finaux de la $\Lambda\mu$-KAM. On étudie les configurations possibles des états finaux de la $\Lambda\mu$-KAM :

– Un état $\langle x[e], K, S\rangle$ est final si, et seulement si, la variable x n'est pas définie dans l'environnement e, ce qui ne peut arriver que si x était une variable libre du terme de départ. Cela signifie qu'on a atteint une variable de tête. Ce cas n'arrive jamais lorsque le terme initial était un $\Lambda\mu$-terme clos ;
– Un état $\langle \lambda x.t[e], K, S\rangle$ est un état final si, et seulement si, $K = \perp$. Cela signifie que l'on a atteint un λ-préfixe d'une forme de tête faible du terme calculé ou un pré-redex bloqué (ce dernier cas n'arrive jamais avec un terme μ-clos) ;
– Un état $\langle (t)u[e], K, S\rangle$ ne peut jamais être final ;
– Un état $\langle \mu\alpha.t[e], K, S\rangle$ est un état final si, et seulement si, $S = \curlywedge$. Cela signifie qu'on a atteint un μ-préfixe d'une forme de tête faible du terme calculé ;
– Un état $\langle (t)\alpha[e], K, S\rangle$ est final si, et seulement si, la variable α n'est pas définie dans l'environnement e, ce qui ne peut arriver que si α était une variable libre du terme de départ. Ce cas n'arrive jamais si on commence avec un terme μ-clos.

Au-delà de la réduction de tête faible. Pour aller au-delà de la réduction de tête faible de $\Lambda\mu$, il faut :

– pouvoir réduire sous les λ et les μ ;
– traiter le cas des variables qui ne seront plus liées dans l'environnement parce qu'on réduit sous les λ et les μ.

On gère ces deux cas en relançant la machine une fois qu'un λ ou μ-préfixe de tête faible a été atteint, et on introduit des constantes comme suit :

– on introduit des constantes simples pour les λ-variables correspondant aux λ-préfixes rencontrés auparavant, ces constantes ont pour effet de bloquer la machine (une variable de tête est atteinte) ;
– on introduit de nouvelles constantes de fond de pile pour les variables de stream, de la forme $\alpha(n)$ ainsi que des constantes de terme v_n^α. Lorsqu'un terme $(t)\alpha$ est atteint et

que la variable α n'est pas définie dans l'environnement parce que liée à un μ-préfixe rencontré plus haut, on ajoute une pile réduite au fond de pile $\alpha(0)$ et on ajoute la nouvelle transition :

$$\langle \lambda x.t \quad [e] \quad \alpha(n) \quad S\rangle \longrightarrow \langle t \quad [x = v_n^\alpha[\emptyset]; e] \quad \alpha(n+1) \quad S\rangle$$

Cette nouvelle transition permet de traiter le cas de la *fst*-réduction pour les pré-redex de type $(\mathcal{T})\mathcal{S}$. La reconstruction du $\Lambda\mu$-terme doit prendre en compte la mise en cohérence des variables de stream puisque la machine implémente une *fst*-réduction locale, à la manière de celle des réseaux SANE.

8.3.3 Une hiérarchie de $\Lambda\mu$-calculs

Une remarque assez immédiate, étant donné le parallélisme entre les abstractions λ et μ, serait d'envisager un calcul construit sur une hiérarchie de tels opérateurs : $\lambda, \mu, \nu, \ldots$ (que l'on pourrait noter $\Lambda^1, \Lambda^2, \Lambda^3 \ldots$ pour plus d'uniformité).

On ne présente que le cas de l'opérateur ν, et encore que très brièvement, car il n'y a pas de grand intérêt technique une fois que l'étude du $\Lambda\mu$-calcul a été menée, l'ensemble des résultats devant se transposer aisément à la hiérarchie entière, quel que soit l'indice considéré dans la hiérarchie.

le $\Lambda\mu\nu$-calcul, que l'on pourrait également appeler le Λ^3-calcul, est défini comme suit :

Définition 8.15 ($\Lambda\mu\nu$-*calcul*)

On considère un ensemble $\mathcal{V}_r$ de ν-variables $(\aleph, \beth, \gimel, \daleth, \ldots)$ en complément des ensembles $\mathcal{V}_s$ et $\mathcal{V}_t$ précédemment considérés. Les termes du $\Lambda\mu\nu$-calcul sont définis par la syntaxe :

$$t, s ::= x \mid \lambda x.t \mid (t)u \mid \mu\alpha.t \mid (t)\alpha \mid \nu\aleph.t \mid (t)\aleph$$

et les réductions du $\Lambda\mu\nu$-calcul sont données par :

$$\begin{array}{llll}
(\lambda x.t)u & \longrightarrow_{\beta_T} & t\{u/x\} & \\
\lambda x.(t)x & \longrightarrow_{\eta_T} & t & \text{si } x \notin FV(t) \\
(\mu\alpha.t)\beta & \longrightarrow_{\beta_S} & t\{\beta/\alpha\} & \\
\mu\alpha.(t)\alpha & \longrightarrow_{\eta_S} & t & \text{si } \alpha \notin FV(t) \\
(\nu\aleph.t)\beth & \longrightarrow_{\beta_R} & t\{\beth/\aleph\} & \\
\nu\aleph.(t)\aleph & \longrightarrow_{\eta_R} & t & \text{si } \aleph \notin FV(t) \\
\mu\alpha.t & \longrightarrow_{fst_S} & \lambda x.\mu\alpha.t\{(v)x\alpha/(v)\alpha\} & \text{si } x \notin FV(t) \\
\nu\aleph.t & \longrightarrow_{fst_R} & \mu\alpha.\nu\aleph.t\{(v)\alpha\aleph/(v)\aleph\} & \text{si } \alpha \notin FV(t)
\end{array}$$

On peut également définir ce qui correspondrait à la réduction μ :

$$\begin{array}{lll}
(\mu\alpha.t)u & \longrightarrow_{\mu_S} & \mu\alpha.t\{(v)u\alpha/(v)\alpha\} \\
(\nu\aleph.t)u & \longrightarrow_{\mu_R^T} & \nu\aleph.t\{(v)u\aleph/(v)\aleph\} \\
(\nu\aleph.t)\alpha & \longrightarrow_{\mu_R^S} & \nu\aleph.t\{(v)\alpha\aleph/(v)\aleph\}
\end{array}$$

Au-delà du côté naturel de cette extension, elle peut être pertinente en ce qu'elle semble correspondre à la hiérarchie des opérateurs de contrôle délimité $\mathtt{shift}_i/\mathtt{reset}_i$ [DF90, DY99, BBD05].

8.3.4 Sémantique dénotationnelle de Streicher pour $\Lambda\mu$

Streicher a proposé une sémantique dénotationnelle pour le $\Lambda\mu$-calcul [Str05] que nous présentons ici dans un souci de complétude et parce qu'elle illustre et donne une justification formelle supplémentaire à l'interprétation du $\Lambda\mu$-calcul comme calcul de streams.

Streicher interprète le $\Lambda\mu$-calcul dans un domaine D solution de l'équation de domaines :

$$D = \left(D^{\mathbb{N}}\right)^{\mathbb{N}} \to D.$$

où $\mathbb{N}$ est le domaine des entiers naturels et $D^{\mathbb{N}}$ le domaine des streams d'éléments de D pour un domaine D. On considère trois constructions supplémentaires :

- $\mathit{fst}(S)$ désigne le premier élément d'une stream, c'est-à-dire d'un élément de $D^{\mathbb{N}}$, ou encore $\mathit{fst}(S) = S(0)$;
- $\mathit{snd}(S)$ désigne le reste de la stream, ou encore $\mathit{snd}(S) = [n : \mathbb{N} \mapsto S(n+1)]$ et enfin
- $d :: S$ désigne la stream obtenue en mettant en tête de la stream S, l'élément $d \in D$, ou encore

$$d :: S = \begin{cases} 0 & \mapsto \quad d \\ n+1 & \mapsto \quad S(n) \end{cases}$$

On interprète $\Lambda\mu$ dans un domaine D tel que $D = \left(D^{\mathbb{N}}\right)^{\mathbb{N}} \to D$ en notant ρ pour un environnement, c'est-à-dire une fonction qui associe une élément de D aux variables de terme et un élément de $D^{\mathbb{N}}$ aux variables de stream :

$$\begin{array}{lcll}
[\![x]\!]\rho S & \triangleq & \rho(x) & \\
[\![\lambda x.t]\!]\rho S & \triangleq & [\![t]\!]\rho' S' & \text{où } \rho' = \rho\,\{\mathit{fst}(\mathit{fst}(S))/x\} \text{ et } S' = \mathit{snd}(\mathit{fst}(S)) :: \mathit{snd}(S) \\
[\![(t)u]\!]\rho S & \triangleq & [\![t]\!]\rho S' & \text{où } S' = ([\![u]\!]\rho :: \mathit{fst}(S)) :: \mathit{snd}(S) \\
[\![\mu\alpha.t]\!]\rho S & \triangleq & [\![t]\!]\rho' S' & \text{où } \rho' = \rho\,\{\mathit{fst}(S)/\alpha\} \text{ et } S' = \mathit{snd}(S) \\
[\![(t)\alpha]\!]\rho S & \triangleq & [\![t]\!]\rho S' & \text{où } S' = \rho(\alpha) :: S
\end{array}$$

8.3.5 Un calcul de stream : le $\Lambda\mathcal{S}$-calcul

On peut définir un calcul de stream comme suit :

Définition 8.16 ($\Lambda\mathcal{S}$-*calcul*)

On considère à nouveau un ensemble $\mathcal{V}_t$ de variables de terme et un ensemble $\mathcal{V}_s$ de variables de stream. Les termes du $\Lambda\mathcal{S}$-calcul sont définis comme suit :

$$\begin{array}{llcl}
\textit{Termes} & t, u, v & ::= & x \mid \Lambda\mathcal{S}_v.t \mid (t)\mathcal{S} \\
\textit{Streams} & \mathcal{S} & ::= & \alpha \mid t \cdot \mathcal{S} \\
\textit{Streams de variables} & \mathcal{S}_v & ::= & \alpha \mid x \cdot \mathcal{S}_v
\end{array}$$

On demande par ailleurs que dans une stream de variables, toutes les variables de terme soient différentes les unes des autres. L'ensemble des $\Lambda\mathcal{S}$-termes est noté $\Sigma_{\Lambda\mathcal{S}}$.

Définition 8.17 (*Réductions de* $\Lambda\mathcal{S}$)

Les réductions du $\Lambda\mathcal{S}$*-calcul sont définies comme suit :*

$$
\begin{array}{l|llll}
\boldsymbol{\beta} & (\Lambda\mathcal{S}_v.t)\mathcal{S} & \longrightarrow_{\Lambda\mathcal{S}} & t\,\{\mathcal{S}/\mathcal{S}_v\} & si\ \sharp(\mathcal{S}_v) \leq \sharp(\mathcal{S}) \\
\boldsymbol{\eta} & \Lambda\mathcal{S}_v.(t)\mathcal{S}_v & \longrightarrow_{\Lambda\mathcal{S}} & t & si\ \mathcal{V}(\mathcal{S}_v) \cap FV(t) = \emptyset \\
n^{th} & \Lambda x_1 \ldots x_n \cdot \alpha.t & \longrightarrow_{\Lambda\mathcal{S}} & \Lambda x_1 \ldots x_{n+1} \cdot \alpha.t\,\{x_{n+1} \cdot \alpha/\alpha\} & si\ x_{n+1} \notin FV(t) \\
 & & & & \cup\{x_1, \ldots, x_n\}
\end{array}
$$

où $\mathcal{V}(\mathcal{S}_v)$ *est l'ensemble des variables de* $\mathcal{S}_v$ *et où* $\sharp(\mathcal{S})$ *est la longueur de* $\mathcal{S}$ *définie par :*

- $\sharp(\alpha) = 0$;
- $\sharp(t \cdot \mathcal{S}) = 1 + \sharp(\mathcal{S})$.

La substitution de streams est définie comme suit :

Définition 8.18 (*Substitution de stream*)

La substitution de stream $t\,\{\mathcal{S}/\mathcal{S}_v\}$ *est définie comme suit (on suppose que les variables de* $\mathcal{S}$ *et de* $\mathcal{S}_v$ *sont deux à deux disjointes) :*

- $t\,\{u \cdot \mathcal{S}/x \cdot \mathcal{S}_v\} = t\,\{u/x\}\,\{\mathcal{S}/\mathcal{S}_v\}$;
- $t\,\{\mathcal{S}/\alpha\}$ *est la substitution sans capture de variables de* α *par la stream* $\mathcal{S}$.

Définition 8.19 (*Traduction de* $\Lambda\mu$ *dans* $\Lambda\mathcal{S}$)

Soit $t \in \Sigma_{\Lambda\mu}$. *On définit* $[\_]^{\mathcal{S}}$ *comme suit :*

$$
\begin{array}{lcl}
[x]^{\mathcal{S}} & = & x \\
[\lambda x.t]^{\mathcal{S}} & = & \Lambda x \cdot \alpha.([t]^{\mathcal{S}})\alpha \\
[\mu\alpha.t]^{\mathcal{S}} & = & \Lambda\alpha.[t]^{\mathcal{S}} \\
[(t)u]^{\mathcal{S}} & = & \Lambda\alpha.([t]^{\mathcal{S}})[u]^{\mathcal{S}} \cdot \alpha \\
[(t)\alpha]^{\mathcal{S}} & = & ([t]^{\mathcal{S}})\alpha
\end{array}
$$

Définition 8.20 (*Traduction de* $\Lambda\mathcal{S}$ *dans* $\Lambda\mu$)

Soit $t \in \Sigma_{\Lambda\mathcal{S}}$. *On définit* $[\_]^{\Lambda\mu}$ *comme suit :*

$$
\begin{array}{lcl}
[x]^{\Lambda\mu} & = & x \\
[\Lambda x_1 \ldots x_k \cdot \alpha.t]^{\Lambda\mu} & = & \lambda x_1 \ldots \lambda x_n.\mu\alpha.[t]^{\Lambda\mu} \\
[(t)u_1 \cdot \cdots \cdot u_n \cdot \alpha]^{\Lambda\mu} & = & ([t]^{\Lambda\mu})[u_1]^{\Lambda\mu} \ldots [u_n]^{\Lambda\mu}\alpha
\end{array}
$$

Du fait que la traduction de $\Lambda\mu$ dans $\Lambda\mathcal{S}$ fait des $\eta_{\mathcal{S}}$-expansions, on a :

Proposition 8.21

Soit $t \in \Sigma_{\Lambda\mu}$. *On a* $\left[[t]^{\mathcal{S}}\right]^{\Lambda\mu} =_{\eta_{\mathcal{S}}} t$.

Démonstration : Par une induction immédiate sur la structure de t.

□

et en conséquence, on a un résultat de simulation faible :

Proposition 8.22 (*Simulation de* $\Lambda\mu$ *par* $\Lambda\mathcal{S}$)

Soient $t, u \in \Sigma^c_{\Lambda\mu}$. *Si* $t \longrightarrow^{\star}_{\Lambda\mu} u$, *alors il existe* $u' \in \Sigma_{\Lambda\mathcal{S}}$ *tel que* $[t]^{\mathcal{S}} \longrightarrow^{\star}_{\Lambda\mathcal{S}} u'$ *et* $u \longrightarrow^{\star}_{\Lambda\mu} u_0 =_{\boldsymbol{\eta}} [u']^{\Lambda\mu}$.

En particulier, si $t \longrightarrow^{\star}_{\Lambda\mu} u$ avec u est en forme normale canonique, on a alors $[t]^{\mathcal{S}} \longrightarrow^{\star}_{\Lambda\mathcal{S}} u'$ avec $u =_{fst\boldsymbol{\eta}} [u']^{\Lambda\mu}$.

Dans l'autre sens en revanche, on peut simuler effectivement les réduction de $\Lambda\mathcal{S}$ par $\Lambda\mu$:

Proposition 8.23 (*Simulation de $\boldsymbol{\Lambda\mathcal{S}}$ par $\boldsymbol{\Lambda\mu}$*)

Soient $t, u \in \Sigma_{\Lambda\mathcal{S}}$, t un terme clos. Alors si $t \longrightarrow^{\star}_{\Lambda\mathcal{S}} u$ on a $[t]^{\Lambda\mu} \longrightarrow^{\star}_{\Lambda\mu} [u]^{\Lambda\mu}$.

Démonstration : Par induction sur la longueur de la réduction $t \longrightarrow^{\star}_{\Lambda\mathcal{S}} u$.

En effet, chaque réduction de $\Lambda\mathcal{S}$ peut être simulée par une séquence de réductions de $\Lambda\mu$ qui dépend de la longueur de la stream impliquée dans la réduction :

- la réduction n^{th} est directement simulée par la *fst*-réduction tandis que
- la $\boldsymbol{\eta}$-réduction est simulée par une séquence de η_T-réductions de la longueur de la stream $\mathcal{S}_v$ réduite puis par une η_S-réduction et qu'enfin
- la $\boldsymbol{\beta}$-réduction est simulée par une séquence de k *fst*-réductions où k est la différence entre les taille de $\mathcal{S}$ et $\mathcal{S}_v$, puis de $\sharp(\mathcal{S})$ β_T-réductions suivies d'une β_S-réduction.

$\square$

Deuxième partie

Focalisation en Logique linéaire

Chapitre 9

Une preuve modulaire de la focalisation

Résumé:

On donne dans ce chapitre une nouvelle preuve du résultat de focalisation d'Andreoli [And90, And92]. Notre méthode est modulaire et repose sur une analyse précise des propriétés de permutation des règles d'inférence de la logique linéaire. La preuve repose de manière essentielle sur la notion de ***graphe de focalisation*** dont l'étude met en évidence la possibilité de choisir un foyer sur lequel focaliser. Nous illustrons le caractère modulaire de la preuve de deux manières : nous commençons par démontrer le résultat pour MALL puis nous l'étendons à la logique linéaire entière, par ailleurs, nous donnons deux exemples d'extension du théorème de focalisation.

Références : Les résultats de ce chapitre ont été publiés dans l'article *From proofs to focused proofs : a modular proof of focalization in Linear Logic* [MS07].

Sommaire

9.1. Introduction

Le résultat de focalisation vient des travaux d'Andreoli au début des années 90 [And90, And92]. Le but initial d'Andreoli était de disposer d'un algorithme de recherche de preuve en logique linéaire qui éviterait des recherches redondantes dues à des différences de séquentialisation non pertinentes des règles d'inférences.

Dans ce sens, le résultat de focalisation se décompose en deux parties :

- l'inversibilité des négatifs : on décompose les formules négatives dès qu'on les rencontre ;
- focalisation sur les positifs : une fois qu'on a choisi de décomposer une formule positive, on continue à décomposer héréditairement ses sous-formules sans appliquer de règle d'inférence aux autres formules du séquent, jusqu'à rencontrer une sous-formule négative.

La focalisation a eu de très nombreuses conséquences en logique linéaire : en programmation en logique linéaire [AP90, Mil96] bien sûr, dans le développement de systèmes de preuve qui permettent à la fois du chaînage avant (forward-chaining) et du chaînage arrière (backward-chaining) [JNS05, LM07]. Plus fondamentalement, la structure des preuves focalisées est un ingrédient-clé du développement de la logique linéaire polarisée [Lau02] et de la Ludique [Gir01].

Le résultat d'Andreoli est à rapprocher des résultats concernant la prouvabilité uniforme de Miller et al. [MNPS91] (voir le chapitre 1) : les stratégies de recherche de preuve uniformes et focalisées présentent de grandes similarités. La focalisation a l'intérêt particulier de s'appliquer à toute la logique linéaire et pas seulement à l'un de ses fragments.

Nous souhaitons donner dans ce chapitre une preuve simple de la focalisation qui puisse facilement être adaptée à des extensions de la logique linéaire.

9.2. Retour sur le résultat d'Andreoli

La focalisation dans Σ_3. La propriété de focalisation de la logique linéaire a été découverte par Jean-Marc Andreoli dans sa thèse de doctorat [And90, And92]. Le but initial d'Andreoli était de disposer d'un algorithme de recherche de preuve en logique linéaire qui éviterait des recherches redondantes dues à des différences de séquentialisation non pertinentes des règles d'inférences.

Andreoli propose donc successivement des systèmes de preuve de plus en plus contraints pour aboutir au système focalisé Σ_3 que nous présentons en figure 9.1 et qui comprend deux sortes de séquents. Dans le séquent $\vdash \Psi\colon \Delta \Uparrow L$, les « zones » Ψ et Δ sont des multi-ensembles de formules et L est une liste de formules. Ce séquent encode le séquent usuel de LL à un côté $\vdash\ ?\Psi, \Delta, L$ (ici, on suppose que L est maintenant considéré comme un multi-ensemble et non plus une liste). Ce séquent devra aussi satisfaire un invariant qui demande que Δ ne contienne que des littéraux et des formules positives. Dans le séquent $\vdash \Psi\colon \Delta \Downarrow F$, la zone Ψ est un multi-ensemble de formules et Δ est un multi-ensemble de littéraux et de formules positives, et F est simplement une formule, positive ou négative.

Le théorème de focalisation d'Andreoli énonce la complétude de la prouvabilité dans Σ_3 par rapport à la prouvabilité de LL :

$$\frac{\vdash \Psi : \Delta \Uparrow L}{\vdash \Psi : \Delta \Uparrow \bot, L}\,[\bot] \qquad \frac{\vdash \Psi : \Delta \Uparrow F, G, L}{\vdash \Psi : \Delta \Uparrow F \parr G, L}\,[\parr] \qquad \frac{\vdash \Psi, F : \Delta \Uparrow L}{\vdash \Psi : \Delta \Uparrow ?F, L}\,[?]$$

$$\frac{}{\vdash \Psi : \Delta \Uparrow \top, L}\,[\top] \qquad \frac{\vdash \Psi : \Delta \Uparrow F, L \quad \vdash \Psi : \Delta \Uparrow G, L}{\vdash \Psi : \Delta \Uparrow F \,\&\, G, L}\,[\&] \qquad \frac{\vdash \Psi : \Delta \Uparrow B[y/x], L}{\vdash \Psi : \Delta \Uparrow \forall x.B, L}\,[\forall]$$

$$\frac{}{\vdash \Psi : \cdot \Downarrow \mathbf{1}}\,[\mathbf{1}] \qquad \frac{\vdash \Psi : \Delta_1 \Downarrow F \quad \vdash \Psi : \Delta_2 \Downarrow G}{\vdash \Psi : \Delta_1, \Delta_2 \Downarrow F \otimes G}\,[\otimes] \qquad \frac{\vdash \Psi : \cdot \Uparrow F}{\vdash \Psi : \cdot \Downarrow !F}\,[!]$$

$$\frac{\vdash \Psi : \Delta \Downarrow F_1}{\vdash \Psi : \Delta \Downarrow F_1 \oplus F_2}\,[\oplus_0] \qquad \frac{\vdash \Psi : \Delta \Downarrow F_2}{\vdash \Psi : \Delta \Downarrow F_1 \oplus F_2}\,[\oplus_1] \qquad \frac{\vdash \Psi : \Delta \Downarrow B[t/x]}{\vdash \Psi : \Delta \Downarrow \exists x.B}\,[\exists]$$

$$\frac{\vdash \Psi : \Delta, F \Uparrow L}{\vdash \Psi : \Delta \Uparrow F, L}\,R\Uparrow \qquad \frac{\vdash \Psi : \Delta \Uparrow F}{\vdash \Psi : \Delta \Downarrow F}\,R\Downarrow$$

$$\frac{}{\vdash \Psi : X^\perp \Downarrow X}\,I_1 \qquad \frac{}{\vdash \Psi, X^\perp : \cdot \Downarrow X}\,I_2$$

$$\frac{\vdash \Psi : \Delta \Downarrow F}{\vdash \Psi : \Delta, F \Uparrow \cdot}\,D_1 \qquad \frac{\vdash \Psi, F : \Delta \Downarrow F}{\vdash \Psi, F : \Delta \Uparrow \cdot}\,D_2$$

Fig. 9.1 – Le système Σ_3 des preuves focalisées de [And92] pour la logique linéaire. *Les contraintes sur les règles sont les suivantes : dans la règle $\forall$, la variable y n'est pas libre dans la conclusion. Dans $R\Uparrow$, F est positive ou bien un littéral négatif tandis que dans $R\Downarrow$, F est négative. Dans I_1 et I_2, X est un littéral positif. Dans D_1 et D_2, F n'est pas négative.*

Théorème 9.1

Étant donné Ψ un ensemble de formules, Γ un multi-ensemble de formules positives ou de littéraux et Δ une liste arbitraire de formules, $\vdash ?\Psi, \Gamma, \Delta$ est prouvable en LL si, et seulement si, le séquent $\vdash \Psi : \Gamma \Uparrow \Delta$ est prouvable dans le système Σ_3.

Les caractéristiques du système Σ_3 et des preuves de LL sous-jacentes au système sont les suivantes :

- ***Inversibilité des négatifs.*** On décompose une formule négative dès que celle-ci apparaît ;
- ***Focalisation sur les positifs.*** Une fois qu'on a commencé à décomposer une formule positive F, on continue à utiliser F comme formule principale dans les prémisses de la règle, jusqu'à atteindre une sous-formule négative.

On nuancera légèrement les deux affirmations précédentes lorsque l'on étudiera le cas des exponentiels dans la suite de ce chapitre.

Du point de vue de la recherche de preuve, la focalisation doit être comprise comme suit :

- La décomposition d'une formule négative n'est jamais source d'erreur dans la recherche de preuve : l'application d'une règle ne nécessite jamais de faire un choix et par ailleurs, le fait de sélectionner une formule négative ne peut jamais faire perdre la prouvabilité du séquent.
- La décomposition des formules positives est plus complexe puisque d'une part les règles impliquent généralement de faire un choix (choisir entre les règles $[\oplus_0]$ et $[\oplus_1]$,

choisir un partitionnement du contexte pour le tenseur, etc...) pour lequel on peut se tromper et que d'autre part, un séquent peut être prouvable sans que la règle appliquée au séquent racine de la preuve puisse forcément être sur une formule positive donnée. En revanche, la focalisation dit qu'on peut réduire le non-déterminisme des positifs en appliquant une stratégie entêtée consistant à sélectionner une formule positive comme foyer et à appliquer des inférences à cette formule et ses sous-formules jusqu'à arriver aux sous-formules négatives. Les éventuels backtrackings que l'on aura à faire concerneront le choix des règles d'inférence et la sélection du foyer, mais pas d'aller-retour d'une formule positive à une autre.

Commentaires sur la focalisation : quel intérêt à une autre approche ? Nous nous proposons dans ce chapitre de donner une autre preuve de la focalisation. Nous expliquons ici ce qui nous guide et nous semble important dans cette direction.

- La focalisation devrait être un résultat simple. La focalisation est un résultat essentiel de la théorie de la démonstration de la logique linéaire. Il ne s'agit pas d'un résultat complexe, mais il est souvent considéré comme tel car les preuves qui en sont données sont souvent complexes et ne sont pas évidentes à lire. Ainsi, l'utilisation du système Σ_3, par exemple, s'il apporte de nombreuses informations sur la manière dont on peut contraindre la recherche de preuve en logique linéaire, introduit une complexité dont il n'est pas certain qu'elle soit totalement nécessaire pour mettre à jour les éléments les plus importants de la focalisation. ***Nous souhaitons donc proposer une preuve aussi simple que possible, adaptable à d'autres cadres théoriques.***
- Relier les preuves et ne pas seulement parler de complétude de la prouvabilité. La focalisation est, à l'origine, une manière d'élaguer l'espace de recherche de preuve en ne considérant que des preuves dans un système contraint comme Σ_3. Il s'agit également d'une manière de quotienter l'espace des preuves de LL, de sélectionner des représentants de preuves habituelles. ***Nous souhaitons mettre en évidence cette relation entre preuves linéaires, en présentant la focalisation comme un processus de transformation de preuve plutôt qu'en mettant l'accent sur la complétude de la prouvabilité focalisée.***
- Enfin, nous souhaitons avoir une approche de la focalisation dans laquelle l'accent est mis d'avantage sur la focalisation comme propriété de certaines preuves plutôt que comme système à part entière, un peu à la manière des preuves uniformes déjà évoquées.

Nous présentons donc dans la section suivante les propriétés de permutabilité des inférences de MALL et établissons la réversibilité d'une partie des inférences de MALL, concernant les connecteurs dits négatifs. La section 9.4 mettra à profit les résultats de permutabilité étudiés en 9.3 pour établir le résultat de focalisation lui-même.

9.3. Propriétés de permutabilité des règles d'inférences de MALL

Nous avons vu au chapitre 1 les résultats de permutabilité de Kleene-Curry à propos du calcul des séquents LK. Les mêmes questions de permutabilité se posent pour la logique linéaire et sont à la base des résultats de focalisation.

Les propriétés de permutabilité auxquelles nous nous intéressons ici concernent des schémas d'inférence et non pas la permutabilité d'une instance donnée d'une règle d'inférence. Nous allons mettre en évidence deux types de comportement de permutabilité parmi les inférences de MALL :

Définition 9.2 (*Permutation des règles d'inférence*)

α et β désignent deux (schémas de) règles d'inférences.

- $\boxed{\frac{\beta}{\alpha}\downarrow}$**-permutabilité :** *on dit qu'il y a $\boxed{\frac{\beta}{\alpha}\downarrow}$-permutabilité si, étant donné un séquent $\mathcal{S}$ contenant deux formules A et B, pour toute preuve $\mathcal{D}$ de $\mathcal{S}$ commençant avec la règle α (de formule principale A) juste avant que la règle β (de formule principale B) ne soit appliquée, il existe une preuve $\mathcal{D}'$ de $\mathcal{S}$ dans laquelle l'ordre des deux règles a été interverti : la règle β est appliquée d'abord et est immédiatement suivie par la règle α. (il y a bien sûr un cas dégénéré pour les règles sans prémisse, comme $[\top]$, ainsi que des cas de duplication comme avec $[\&]$).*
- $\boxed{\frac{\beta}{\alpha}\updownarrow}$**-permutabilité :** *on parle de $\boxed{\frac{\beta}{\alpha}\updownarrow}$-permutabilité quand il y a à la fois $\boxed{\frac{\beta}{\alpha}\downarrow}$-permutabilité et $\boxed{\frac{\alpha}{\beta}\downarrow}$-permutabilité.*

Pour étudier les permutabilités étant donné un système d'inférence, il faut étudier les propriétés de permutabilité de sous-ensembles des règles d'inférence et non pas seulement les propriétés de permutabilité de deux règles d'inférence l'une avec l'autre, ce qui nous conduit aux notions de ***permutabilité faible*** et de ***permutabilité pleine*** :

Définition 9.3 (*Permutabilité faible, permutabilité pleine*)

Étant donnés deux ensembles complémentaires de règles d'inférence $\mathcal{N}$ et $\mathcal{P}$, nous disons que :

- *$\mathcal{P}$ a la propriété de* **permutabilité faible** *si, étant données deux règles α, β de $\mathcal{P}$ nous avons $\boxed{\frac{\beta}{\alpha}\updownarrow}$-permutabilité.*
- *$\mathcal{N}$ a la propriété de* **permutabilité pleine** *quand il a la permutabilité faible et que pour toute paire de règles $(\alpha, \beta) \in \mathcal{P} \times \mathcal{N}$, on a $\boxed{\frac{\beta}{\alpha}\downarrow}$-permutabilité.*

Les connecteurs de MALL peuvent être classifiés en deux ensembles en fonction des propriétés de permutabilité des règles d'inférence qui leur sont associées.

Définition 9.4 (*Polarité des connecteurs, des formules et des inférences*)

Les connecteurs de MALL peuvent être classifiés en deux ensembles :

- *les* **connecteurs positifs** *sont $\otimes, \oplus, \mathbf{1}, \mathbf{0}$;*
- *les* **connecteurs négatifs** *sont $⅋, \&, \bot, \top$.*

De même, les formules de MALL peuvent être classifiées en deux ensembles :

- *les* **formules positives** *sont les formules de la forme $A \otimes B, A \oplus B, \mathbf{1}, \mathbf{0}$;*
- *les* **formules négatives** *sont les formules de la forme $A ⅋ B, A \& B, \bot, \top$.*

On note en outre $\mathcal{N}$ l'ensemble des règles d'inférence correspondant aux connecteurs négatifs de MALL et $\mathcal{P}$ l'ensemble des règles d'inférence correspondant aux connecteurs positifs de MALL.

Remarque 9.5

On notera que les formules atomiques ne sont pas polarisées. La section 9.5 sera consacrée à cette question.

Proposition 9.6 (*Propriétés de Permutabilité des règles d'inférence de MALL*)

$\mathcal{N}$ a la propriété de permutabilité pleine tandis que $\mathcal{P}$ a la propriété de permutabilité faible.

Démonstration : La figure 9.1 présente les différentes permutations des connecteurs binaires.

Les unités ont également les permutabilités souhaitées, ce qui s'obtient du fait que

Permutations pos/pos :

$$\dfrac{\dfrac{\vdash A,C,\Gamma \quad \vdash D,\Delta}{\vdash A,C\otimes D,\Gamma,\Delta}\,[\otimes]}{\vdash A\oplus B,C\otimes D,\Gamma,\Delta}\,[\oplus_0] \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A,C,\Gamma}{\vdash A\oplus B,C,\Gamma}\,[\oplus_0] \quad \vdash D,\Delta}{\vdash A\oplus B,C\otimes D,\Gamma,\Delta}\,[\otimes]$$

$$\dfrac{\dfrac{\vdash A,C,\Gamma}{\vdash A,C\oplus D,\Gamma}\,[\oplus_0]}{\vdash A\oplus B,C\oplus D,\Gamma}\,[\oplus_0] \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A,C,\Gamma}{\vdash A\oplus B,C,\Gamma}\,[\oplus_0]}{\vdash A\oplus B,C\oplus D,\Gamma}\,[\oplus_0]$$

$$\dfrac{\dfrac{\vdash A,C,\Gamma \quad \vdash D,\Delta}{\vdash A,C\otimes D,\Gamma,\Delta}\,[\otimes] \quad \vdash B,\Sigma}{\vdash A\otimes B,C\otimes D,\Gamma,\Delta,\Sigma}\,[\otimes] \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A,C,\Gamma \quad \vdash B,\Sigma}{\vdash A\otimes B,C,\Gamma,\Sigma}\,[\otimes] \quad \vdash D,\Delta}{\vdash A\otimes B,C\otimes D,\Gamma,\Delta,\Sigma}\,[\otimes]$$

Permutations neg/neg :

$$\dfrac{\dfrac{\vdash A,B,C,D,\Gamma}{\vdash A,B,C\parr D,\Gamma}\,[\parr]}{\vdash A\parr B,C\parr D,\Gamma}\,[\parr] \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A,B,C,D,\Gamma}{\vdash A\parr B,C,D,\Gamma}\,[\parr]}{\vdash A\parr B,C\parr D,\Gamma}\,[\parr]$$

$$\dfrac{\dfrac{\vdash A,C,D,\Gamma}{\vdash A,C\parr D,\Gamma}\,[\parr] \quad \dfrac{\vdash B,C,D,\Gamma}{\vdash B,C\parr D,\Gamma}\,[\parr]}{\vdash A\,\&\,B,C\parr D,\Gamma}\,[\&] \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A,C,D,\Gamma \quad \vdash A,C,D,\Gamma}{\vdash A\,\&\,B,C,D,\Gamma}\,[\&]}{\vdash A\,\&\,B,C\parr D,\Gamma}\,[\parr]$$

$$\dfrac{\dfrac{\vdash A,C,\Gamma \quad \vdash A,D,\Gamma}{\vdash A,C\,\&\,D,\Gamma}\,[\&] \quad \dfrac{\vdash B,C,\Gamma \quad \vdash B,D,\Gamma}{\vdash B,C\,\&\,D,\Gamma}\,[\&]}{\vdash A\,\&\,B,C\,\&\,D,\Gamma}\,[\&] \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A,C,\Gamma \quad \vdash B,C,\Gamma}{\vdash A\,\&\,B,C,\Gamma}\,[\&] \quad \dfrac{\vdash A,D,\Gamma \quad \vdash B,D,\Gamma}{\vdash A\,\&\,B,D,\Gamma}\,[\&]}{\vdash A\,\&\,B,C\,\&\,D,\Gamma}\,[\&]$$

Permutations neg/pos :

$$\dfrac{\dfrac{\vdash A,C,D,\Gamma}{\vdash A,C\parr D,\Gamma}\,[\parr]}{\vdash A\oplus B,C\parr D,\Gamma}\,[\oplus_0] \quad\longrightarrow\quad \dfrac{\dfrac{\vdash A,C,D,\Gamma}{\vdash A\oplus B,C,D,\Gamma}\,[\oplus_0]}{\vdash A\oplus B,C\parr D,\Gamma}\,[\parr]$$

$$\dfrac{\dfrac{\vdash A,C,D,\Gamma}{\vdash A,C\parr D,\Gamma}\,[\parr] \quad \vdash B,\Delta}{\vdash A\otimes B,C\parr D,\Gamma,\Delta}\,[\otimes] \quad\longrightarrow\quad \dfrac{\dfrac{\vdash A,C,D,\Gamma \quad \vdash B,\Delta}{\vdash A\otimes B,C,D,\Gamma,\Delta}\,[\otimes]}{\vdash A\otimes B,C\parr D,\Gamma,\Delta}\,[\parr]$$

$$\dfrac{\dfrac{\vdash A,C,\Gamma \quad \vdash A,D,\Gamma}{\vdash A,C\,\&\,D,\Gamma}\,[\&]}{\vdash A\oplus B,C\,\&\,D,\Gamma}\,[\oplus_0] \quad\longrightarrow\quad \dfrac{\dfrac{\vdash A,C,\Gamma}{\vdash A\oplus B,C,\Gamma}\,[\oplus_0] \quad \dfrac{\vdash A,D,\Gamma}{\vdash A\oplus B,D,\Gamma}\,[\oplus_0]}{\vdash A\oplus B,C\,\&\,D,\Gamma}\,[\&]$$

$$\dfrac{\dfrac{\vdash A,C,\Gamma \quad \vdash A,D,\Gamma}{\vdash A,C\,\&\,D,\Gamma}\,[\&] \quad \vdash B,\Delta}{\vdash A\otimes B,C\,\&\,D,\Gamma,\Delta}\,[\otimes] \quad\longrightarrow\quad \dfrac{\dfrac{\vdash A,C,\Gamma \quad \vdash B,\Delta}{\vdash A\otimes B,C,\Gamma,\Delta}\,[\otimes] \quad \dfrac{\vdash A,D,\Gamma \quad \vdash B,\Delta}{\vdash A\otimes B,D,\Gamma,\Delta}\,[\otimes]}{\vdash A\otimes B,C\,\&\,D,\Gamma,\Delta}\,[\&]$$

TAB. 9.1 – Permutations des inférences de MALL.

chaque unité est une version d'arité nulle d'un connecteur binaire de MALL. On liste quelques permutations sans chercher à être exhaustif :

$$\dfrac{\dfrac{\vdash A,\Gamma}{\vdash A,\bot,\Gamma}\,[\bot] \quad \dfrac{\vdash B,\Gamma}{\vdash B,\bot,\Gamma}\,[\bot]}{\vdash A \,\&\, B,\bot,\Gamma}\,[\&] \qquad\longleftrightarrow\qquad \dfrac{\dfrac{\vdash A,\Gamma \quad \vdash A,\Gamma}{\vdash A \,\&\, B,\Gamma}\,[\&]}{\vdash A \,\&\, B,\bot,\Gamma}\,[\bot]$$

$$\dfrac{\dfrac{}{\vdash A,\top,\Gamma}\,[\top] \quad \dfrac{\mathcal{D}}{\vdash B,\Delta}}{\vdash A \otimes B,\top,\Gamma,\Delta}\,[\otimes] \qquad\longrightarrow\qquad \dfrac{}{\vdash A \otimes B,\top,\Gamma,\Delta}\,[\top]$$

$$\dfrac{\dfrac{\vdash A,\Gamma}{\vdash A,\bot,\Gamma}\,[\bot]}{\vdash A \oplus B,\bot,\Gamma}\,[\oplus_0] \qquad\longrightarrow\qquad \dfrac{\dfrac{\vdash A,\Gamma}{\vdash A \oplus B,\Gamma}\,[\oplus_0]}{\vdash A \oplus B,\bot,\Gamma}\,[\bot]$$

$$\dfrac{}{\vdash \top,\bot,\Gamma}\,[\top] \qquad\longleftrightarrow\qquad \dfrac{\dfrac{}{\vdash \top,\Gamma}\,[\top]}{\vdash \top,\bot,\Gamma}\,[\bot]$$

L'ensemble de ces permutations permet d'assurer les bonnes propriétés de permutabilité :

- Les permutations de type pos/pos assurent la permutabilité faible des positifs ;
- Les permutations de type neg/neg assurent la permutabilité faible des négatifs ;
- Les permutations de type neg/pos en complément des permutations de type neg/neg assurent la permutabilité pleine des négatifs.

□

Remarque 9.7

On notera que les positifs n'ont pas la permutabilité pleine : le séquent $\vdash a^\perp \parr b^\perp, a \otimes b$ *n'a pas de preuve sans coupure qui commence par une règle* $\otimes$*, c'est-à-dire qu'il est impossible de permuter le* $\parr$ *et le* $\otimes$ *de la preuve*

$$\dfrac{\dfrac{\dfrac{}{\vdash a^\perp, a}\,ini \quad \dfrac{}{\vdash b^\perp, b}\,ini}{\vdash a^\perp, b^\perp, a \otimes b}\,[\otimes]}{\vdash a^\perp \parr b^\perp, a \otimes b}\,[\parr]$$

Réversibilité des inférences négatives. La propriété de réversibilité (ou inversibilité) des inférences négatives de MALL est une conséquence immédiate de la permutabilité pleine des négatifs :

Proposition 9.8 (*Réversibilité des inférences négatives*)

Soit N *une formule négative. Si* $\vdash \Gamma, N$ *est prouvable dans MALL, alors il existe une preuve* $\mathcal{D}$ *de ce séquent qui se conclut par une inférence de formule principale* N*.*

Il s'agit d'une conséquence immédiate de la permutabilité et de la propriété de restriction de la règle initiale à sa version atomique. La réversibilité des négatifs peut également s'obtenir facilement par la propriété d'élimination des coupures : par exemple, si

$\dfrac{\mathcal{D}}{\vdash \Gamma, A \parr B}$, alors on peut constuire :

$$\mathcal{D}' = \dfrac{\dfrac{\dfrac{\mathcal{D}}{\vdash \Gamma, A \parr B} \quad \dfrac{\dfrac{}{\vdash A, A^{\perp}}\,ini \quad \dfrac{}{\vdash B, B^{\perp}}\,ini}{\vdash A, B, A^{\perp} \otimes B^{\perp}}\,[\otimes]}{\vdash \Gamma, A, B}\,cut}{\vdash \Gamma, A \parr B}\,[\parr]$$

et $\mathcal{D}'$ se réduit sur une preuve sans coupure de la forme souhaitée.

9.4. Graphes de Focalisation

9.4.1 Étude des troncs positifs

Nous avons étudié les propriétés de permutabilité des règles d'inférence de MALL dans la section précédente, ce qui nous donne la première moitié de la propriété de focalisation, à savoir l'inversibilité des formules négatives.

Nous étudions maintenant la propriété de focalisation des formules positives. Il s'agit donc d'étudier une propriété des preuves des séquents ne contenant que des formules positives ou des littéraux.

Définition 9.9 (*Séquents négatifs, positifs*)

- *Un séquent MALL contenant au moins une fomule négative non-littérale est* **négatif** *;*
- *un séquent est* **positif** *lorsqu'il ne contient aucune formule non-littérale négative et au moins une formule non-littérale positive ;*
- *on parlera sinon de séquent* **atomique***.*

Les seules règles qui peuvent être appliquées à un séquent positif sont des règles positives (le cas de la coupure sera traité plus tard, en section 9.6).

Pour prouver la propriété de focalisation des positifs, on va s'intéresser à la géométrie d'une partie seulement d'une preuve d'un séquent positif, son ***tronc positif*** :

Définition 9.10 (*Tronc positif*)

Étant donnée une preuve MALL $\mathcal{D}$ d'un séquent positif $\mathcal{S}$ on définit le **tronc positif** *$\mathcal{D}^+$ de $\mathcal{D}$ comme étant le préfixe maximal de l'arbre $\mathcal{D}$ ne contenant que des règles positives.*

$\mathcal{D}^+$ est donc l'arbre commençant à la racine de $\mathcal{D}$ et dont les feuilles sont les séquents-conclusions des premières règles non-positives rencontrées sur chacune des branches de l'arbre, si une telle règle existe.

Définition 9.11 (*Frontière d'un tronc positif*)

La **frontière** *d'un tronc positif $\mathcal{D}^+$ est l'ensemble des feuilles de $\mathcal{D}^+$.*

La frontière d'un tronc positif contient uniquement des séquents négatifs ou atomiques, mais jamais de séquent positif.

Définition 9.12 (*Formules actives dans un tronc positif*)

Les **formules actives** *dans un tronc positif $\mathcal{D}^+$ sont les formules du séquent-racine $\mathcal{S}$ auxquelles on a appliqué une règle d'inférence dans le tronc considéré.*

$$
\dfrac{\dfrac{\dfrac{\boxed{\vdash q \otimes r, q^{\perp} \parr r^{\perp}} \quad \boxed{\vdash s, s^{\perp}}}{\vdash q \otimes r, s \otimes (q^{\perp} \parr r^{\perp}), s^{\perp}}\,\otimes}{\vdash p \oplus (q \otimes r), s \otimes (q^{\perp} \parr r^{\perp}), s^{\perp}}\,\oplus \qquad \dfrac{}{\vdash \mathbf{1}}\,\mathbf{1}}{\vdash p \oplus (q \otimes r), s \otimes (q^{\perp} \parr r^{\perp}), s^{\perp} \otimes \mathbf{1}}\,\otimes
$$

FIG. 9.2 – Tronc positif et frontière associés à la preuve de la figure 1.17.

On donne en figure 9.2 un exemple de tronc positif. Les séquents de la frontière sont encadrés.

Remarque 9.13

Quand on traitera le cas des exponentiels en section 9.6, on verra que l'on peut ajouter une condition pour raccourcir une branche du tronc positif, cette condition peut aussi être considérée comme exprimant le fait que la règle pour ! *est bipolaire, c'est-à-dire à la fois positive et negative.*

Nous définissons une relation sur les occurrences de formules impliquées dans $\mathcal{D}$: $F \prec G$ si, et seulement si, G est sous-formule (ou sous-occurrence) de F dans le sens précis où l'occurrence G est obtenue à partir de la décomposition de F le long d'une branche de Π.

Définition 9.14 ($\prec$*-relation*)

La **relation de sous-occurrence** *(notée $\prec$) sur les occurrences de formules apparaissant dans $\mathcal{D}$ est la clôture réflexive et transitive de la relation binaire $\prec^1$ définie par $F \prec^1 G$ s'il existe dans $\mathcal{D}$ une règle α de séquent conclusion $\mathcal{S}$ et de prémisses $(\mathcal{S}_i)_{i \in I}$ telles que*
- *soit F est la formule principale de $\mathcal{S}$ et G est une sous-formule de F produite par la règle α dans l'un des $\mathcal{S}_i$.*
- *soit F n'est pas la formule principale de $\mathcal{S}$ et G est une occurrence de la formule F dans l'un des $\mathcal{S}_i$.*

Si $F \prec G$ on dit que G est une **$\prec$-sous-formule** *de F ou un* **descendant** *de F.*

Le lemme suivant nous aidera à prouver le résultat principal :

Lemme 9.15

Soit $\mathcal{D}^+$ un tronc positif de racine $\mathcal{S}$ et de fontière $\mathcal{F}$. Pour tout $\mathcal{S}' \in \mathcal{F}$ la relation $\prec$ définit une fonction injective de $\mathcal{S}'$ dans $\mathcal{S}$.

Démonstration : Nous prouvons en fait un résultat plus fort : le résultat tient pour tout séquent apparaissant dans le tronc, et pas seulement pour les séquents de $\mathcal{F}$. Le résultat est prouvé par induction sur la hauteur du séquent considéré dans $\mathcal{D}^+$:
- Le cas de base est trivial puisque le séquent considéré est $\mathcal{S}$ lui-même (on se rappelle que $\prec$ est réflexive) : on prend l'identité comme fonction.
- On suppose le résultat vrai pour une hauteur donnée $n \leq h(\mathcal{D}^+)$ et on suppose $n+1 \leq h(\mathcal{D}^+)$. Soit $\mathcal{S}_{n+1}$ un séquent de hauteur $n+1$ et soit α la règle dont $\mathcal{S}_{n+1}$ est une prémisse. On appelle $\mathcal{S}_n$ la conclusion de α.
  Par hypothèse d'induction $\mathcal{S}_n$ satisfait la condition.
  On peut par ailleurs définir une fonction injective ι_n de $\mathcal{S}_{n+1}$ (vu comme ensemble d'occurences de formules) dans $\mathcal{S}_n$ comme suit : $\iota_n(G) = F$ si $F \prec^1 G$.
  La fonction ainsi construite est injective grâce au fait que toute règle MALL positive produit au plus (et en fait exactement) une sous-formule de la formule principale dans chaque prémisse de la règle. En composant la fonction que l'on

vient de construire à la fonction injective qui nous est fournie par l'hypothèse d'induction, on voit que $\mathcal{S}_{n+1}$ satisfait la condition.

Par induction on obtient le résultat attendu.

□

Lemme 9.16

Soit $\vdash \Gamma$ un séquent positif et $\mathcal{D}$ une preuve de $\vdash \Gamma$.

Si $F \in \Gamma$ est telle qu'aucune règle $[\otimes]$ n'est appliquée à une $\prec$-sous-formule de F dans $\mathcal{D}^+$, alors il existe exactement un séquent de la frontière de $\mathcal{D}^+$ contenant une formule G telle que $F \prec G$, à moins que la frontière de $\mathcal{D}^+$ ne soit vide.

Remarque 9.17

Une formule satisfaisant la condition du lemme précédent est dite **non-branchante**.

Lorsqu'au contraire une règle $[\otimes]$ est appliquée à une $\prec$-sous-formule de F dans $\mathcal{D}^+$, alors il peut y avoir plusieurs séquents de la frontière contenant des formules G telles que $F \prec G$, comme illustré par la formule $s \otimes (q^\perp \parr r^\perp)$ de la figure 9.2.

Démonstration : On montre le résultat par induction sur le nombre de règles $[\otimes]$ apparaissant dans $\mathcal{D}^+$.

S'il n'y a aucune règle $[\otimes]$ dans $\mathcal{D}^+$, alors $\mathcal{D}^+$ est une simple séquence de séquents qui se termine en un séquent de la frontière (l'utilisation de la règle $[\mathbf{1}]$ est impossible car on suppose la frontière non-vide). Toutes les règles positives, à l'exception de $[\otimes]$ et $[\mathbf{1}]$, sont telles que pour chaque formules F de la conclusion de la règle, il y a une et une seule formule G de la prémisse telle que $F \prec G$.

Supposons que le tronc positif contienne $n+1$ règles $[\otimes]$ et que le résultat soit vrai pour $j \leq n$. Considérons la règle $[\otimes]$ la plus basse du tronc positif (c'est-à-dire le premier branchement de la preuve) qui a deux prémisses $\mathcal{S}_1$ et $\mathcal{S}_2$. Soit F une formule non-branchante de la conclusion du tronc positif, elle a forcément une $\prec$-sous-formule F' dans la conclusion de cette règle (même argument que ci-dessus) qui est dans l'une des deux portions du contexte. Il y a donc une formule F'', $F \prec F''$ dans l'une des deux prémisses exactement, par exemple $\mathcal{S}_1$. On considère la sous-preuve $\mathcal{D}_1$ de $\mathcal{D}$ enracinée en $\mathcal{S}_1$ et le tronc positif $\mathcal{D}_1^+$ de $\mathcal{D}_1$, qui est un sous-arbre de $\mathcal{D}^+$. La frontière de $\mathcal{D}_1^+$ est une partie de la frontière de $\mathcal{D}$ est elle est non-vide puisque $\mathcal{S}_1$ contient plus d'une formule. On peut donc appliquer l'hypothèse d'induction à $\mathcal{D}_1^+$ montrant qu'il existe exactement un séquent de la frontière de $\mathcal{D}_1^+$ contenant une formule G telle que $F \prec F' \prec F'' \prec G$. Par ailleurs, les séquents de la frontière au-dessus de $\mathcal{S}_2$ ne peuvent pas contenir de descendant de F puisque $\mathcal{S}_2$ ne contient lui-même aucun descendant de F. Ceci conclut la preuve du lemme.

□

9.4.2 Une abstraction des preuves : les graphes de focalisation

Maintenant que nous avons vu quelques propriétés des troncs positifs, nous introduisons un graphe construit à partir des tronc positifs qui est une abstraction de la preuve ne retenant que certaines informations relatives au possibilités de permutations des formules de la conclusion ; il s'agit du ***graphe de focalisation*** de la preuve.

À un tronc positif, on associe un graphe comme suit :

Définition 9.18 (*Graphe de Focalisation*)

Étant donné un tronc positif $\mathcal{D}^+$, on définit le **Graphe de Focalisation** $\mathcal{G}$ *comme le graphe dont*

- *les sommets du graphe sont les formules actives de $\mathcal{D}^+$;*
- *les arêtes du graphe sont définies comme suit : il y a une arête de F à G si, et seulement si, il y a un séquent $\mathcal{S}'$ dans la frontière contenant une $\prec$-sous-formule négative F' de F et une $\prec$-sous-formule positive G' de G.*

Exemple 9.19

Le Graphe de Focalisation associé à notre exemple de preuve de la figure 1.17 est :

$$s^\perp \otimes \mathbf{1} \qquad s \otimes (q^\perp \parr r^\perp) \longrightarrow p \oplus (q \otimes r)$$

Le graphe de l'exemple précédent est acyclique. On va montrer dans la suite que l'acyclicité est une propriété toujours vraie des graphes de focalisation, ce qui sera l'étape importante pour achever notre preuve de focalisation.

Lemme 9.20

Si $\mathcal{S}'$ et $\mathcal{S}''$ sont des séquents qui apparaissent dans des branches différentes de $\mathcal{D}^+$, alors il y a au plus une formule de la racine de $\mathcal{D}^+$ qui a des $\prec$-sous-formules à la fois dans $\mathcal{S}'$ et dans $\mathcal{S}''$.

Démonstration : Nous raisonnons par l'absurde et supposons que ce ne soit pas le cas : soient F, G deux formules de la racine de $\mathcal{D}^+$ et $F', G' \in \mathcal{S}'$ et $F'', G'' \in \mathcal{S}''$ avec $F \prec F', F''$ et $G \prec G', G''$.

Soit $\mathcal{S} = \mathcal{S}' \wedge \mathcal{S}''$ le plus haut prédécesseur commun aux deux séquents dans le tronc positif. Du fait de la maximalité de $\mathcal{S}$, il s'agit forcément d'une règle branchante, c'est-à-dire d'une règle $[\otimes]$ ayant deux prémisses $\mathcal{S}_0'$ et $\mathcal{S}_0''$, avec $\mathcal{S}'$ et $\mathcal{S}''$ se trouvant chacun au-dessus d'une prémisse différente de la règle. La formule principale H de la règle ne peut pas être $\prec$-sous-formule de F et de G, par exemple $F \not\prec H$. Pour que $F \prec F', F''$, il faut qu'il y ait une $\prec$-sous-formule de F dans chacune des prémisses $\mathcal{S}_0'$ et $\mathcal{S}_0''$, ce qui entraîne qu'il y a deux formules F_0' et F_0'' dans le séquent conclusion de la règle $[\otimes]$, contredisant le lemme 9.15.

□

9.4.3 Acyclicité des graphes de focalisation

Proposition 9.21

Soit $\mathcal{S}$ un séquent positif et $\mathcal{D}$ une preuve de $\mathcal{S}$.

Le Graphe de Focalisation de $\mathcal{D}^+$ est acyclique.

Démonstration : Nous prouvons le résultat par l'absurde.

Soit $\mathcal{G}$ le graphe de focalisation associé à $\mathcal{D}^+$. Supposons que $\mathcal{G}$ ait un cycle et considérons un tel cycle de longueur minimale $(F_1 \to F_2 \to \cdots \to F_n \to F_1)$ dans $\mathcal{G}$. Considérons $\mathcal{S}_1, \ldots, \mathcal{S}_n$ les séquents de la frontière qui justifient les flèches du cycle.

Grâce au lemme 9.20, ces séquents sont en fait définis de manière unique. Dans le même ordre d'idée, on peut noter immédiatement que le cycle est nécessairement de longueur $n \geq 2$: grâce au lemme 9.15, deux $\prec$-sous-formules de la même formule ne peuvent jamais être dans le même séquent de la frontière du tronc positif et par le lemme 9.20 il ne peut donc pas y avoir de cycle de longueur un.

Soit $\mathcal{S}_0 = \bigwedge_{i=1}^n \mathcal{S}_i$, le plus haut séquent dans $\mathcal{D}^+$ tel que les $\mathcal{S}_i$ soient toutes des feuilles du sous-arbre de $\mathcal{D}^+$ enraciné en $\mathcal{S}_0$. Nous allons obtenir la contradiction en étudiant $\mathcal{S}_0$ et en raisonnant par cas sur la dernière règle appliquée à ce séquent $\mathcal{S}_0$:

- cette règle ne peut pas être la règle [**1**] puisque cette règle ne produit pas de prémisse et donc qu'il y aurait un cycle vide ce qui n'a pas de sens. N'importe quelle autre règle sans prémisse conduirait à la même contradiction.
- Si la règle est l'une des règles $[\oplus_i]$, alors la prémisse $\mathcal{S}_0'$ de la règle satisfait également la condition requise pour $\mathcal{S}_0$ (à savoir que tous les $\mathcal{S}_i$ se trouvent dans la preuve enracinée en $\mathcal{S}_0'$) ce qui contredirait la maximalité de $\mathcal{S}_0$. Si la règle est n'importe quelle autre règle non-branchante, la maximalité de $\mathcal{S}_0$ produirait également la contradiction.
- En conséquence, la règle doit être branchante : cela ne peut donc être qu'une règle $[\otimes]$. On note respectivement $\mathcal{S}_L$ et $\mathcal{S}_R$ les prémisses de gauche et de droite de $\mathcal{S}_0$. Soit $G = G_L \otimes G_R$ la formule principale dans $\mathcal{S}_0$ et soit F la formule active du tronc telle que $F \prec G$. On a deux possibilités :

  (i) soit $F \in \{F_1, \ldots, F_n\}$ et F est la seule formule du cycle ayant à la fois des $\prec$-sous-formules dans la prémisse de droite et dans la prémisse de gauche,

  (ii) soit $F \notin \{F_1, \ldots, F_n\}$ et dans ce cas aucune formule du cycle n'a de $\prec$-sous-formule dans les deux prémisses.

  Soit alors I_L (resp. I_R) les ensembles d'indices des formules actives de la racine $\mathcal{S}$ ayant des sous-formules (en relation $\prec$) seulement dans la prémisse de gauche (resp. de droite). Clairement, ni I_L ni I_R n'est vide sinon cela contredirait la maximalité de $\mathcal{S}_0$. En effet, si $I_L = \emptyset$, alors $\mathcal{S}_R$ satisferait la condition d'être dominé par tous les $\mathcal{S}_i, 1 \leq i \leq n$ et $\mathcal{S}_0$ ne serait plus maximal. Par définition des deux ensembles d'indices, on a bien sûr $I_L \cap I_R = \emptyset$ et la seule formule du cycle qui peut ne pas se trouver dans $I_L \cup I_R$ est F si nous sommes dans le cas (i) : dans le cas (ii) toutes les formules du cycle ont leur indice soit dans I_L soit dans I_R.
  En conséquence, il doit y avoir une flèche dans le cycle (et donc dans le graphe) partant d'une formule de I_L vers une formule de I_R (ou le contraire) : en effet, le cycle peut passer une fois par la formule F dans le cas (i) mais pas deux ou alors cela contredirait la minimalité du cycle. Soient $i \in I_L$ et $j \in I_R$ des indices tels que, par exemple, $F_i \to F_j$ est dans $\mathcal{G}$ et soit $\mathcal{S}'$ le séquent de la frontière responsable de cette arête. $\mathcal{S}'$ contient F_i' et F_j' et par définition des ensembles I_L et I_R, $\mathcal{S}'$ ne peut pas être dans l'arbre enraciné en $\mathcal{S}_0$ ce qui est en contradiction avec la manière dont on a construit $\mathcal{S}_0$.

En conséquence, il ne peut y avoir aucun cycle dans le graphe.

□

9.4.4 Pré-focalisation

L'acyclicité du graphe de focalisation nous assure que le graphe a une source, c'est-à-dire une formule qui n'est pointée par aucune autre formule dans le graphe, c'est-à-dire une formule telle que lorsqu'un séquent de la frontière contient une de ses $\prec$-sous-occurrences F, la sous-formule en question n'est pas positive (elle est soit négative soit atomique).

En d'autres termes, cela signifie qu'il y a une formule active du séquent racine dont le niveau le plus élevé de connecteurs positifs est complètement décomposé au sein du tronc positif, et cela indépendamment de toute discipline de focalisation : il s'agit d'une propriété générale de la géométrie des preuves de MALL.

On peut voir cela comme une sorte de résultat de focalisation implicite. En quelque sorte, cela nous dit qu'il y a une formule qui est déjà implicitement focalisée dans le tronc positif puisque, du fait de la permutabilité des règles positives entre elles, l'ordre de deux règles positives n'importe pas.

$$\dfrac{\dfrac{\dfrac{}{\vdash s, s^\perp}\,ini \qquad \dfrac{}{\vdash \underline{\mathbf{1}}}\,\mathbf{1}}{\vdash s, \underline{s^\perp \otimes \mathbf{1}}}\otimes \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\vdash q,q^\perp}\,ini \quad \dfrac{}{\vdash r,r^\perp}\,ini}{\vdash \underline{q\otimes r}, q^\perp, r^\perp}\otimes}{\vdash \underline{p\oplus(q\otimes r)}, q^\perp, r^\perp}\oplus}{\vdash p\oplus(q\otimes r), q^\perp \parr r^\perp}\parr}{\vdash p \oplus (q\otimes r), \underline{s\otimes(q^\perp \parr r^\perp)}, s^\perp\otimes \mathbf{1}}\otimes$$

FIG. 9.3 – Preuve pré-focalisée correspondant à la preuve de la figure 1.17.

Grâce à la permutabilité pleine des négatifs, à la permutabilité faible des positifs et à l'acyclicité du graphe de focalisation, nous pouvons en déduire que, étant donnée une preuve MALL sans coupure $\mathcal{D}$ d'un séquent $\vdash \Gamma$, on peut transformer cette preuve en une autre preuve $\mathcal{D}'$ qui satisfait une discipline que l'on appelle ***pré-focalisation*** :

Définition 9.22 (*Preuve pré-focalisée*)

Une preuve $\mathcal{D}$ MALL sans coupure d'un séquent $\mathcal{S}$ est **pré-focalisée** *si, étant donné un séquent $\mathcal{S}'$ de $\mathcal{D}$:*
- *Si $\mathcal{S}'$ est un séquent négatif, $\mathcal{S}'$ est conclusion d'une règle négative ;*
- *Si $\mathcal{S}'$ est un séquent positif, prémisse d'une règle positive r de formule principale F, alors la règle positive r' dont $\mathcal{S}$ est conclusion a pour formule principale une sous-occurrence de F, à moins que cette sous-occurrence ne soit atomique ;*
- *Si $\mathcal{S}'$ est atomique, on lui applique une règle initiale.*

On montre en figure 9.3 un exemple de preuve pré-focalisée. Les formules positives qui sont les foyers sont mises en valeur.

Théorème 9.23 (*Pré-focalisation*)

Si $\mathcal{D}$ est une preuve MALL sans coupure d'un séquent $\mathcal{S}$, elle peut être transformée, par permutation de règles d'inférence, en une preuve pré-focalisée $\mathcal{D}'$.

Démonstration : On définit grâce aux résultats précédents la transformation suivante, dite de pré-focalisation, sur les preuves de LL :

1. **Phase négative :** Si $\mathcal{S}$ est négatif, grâce à la permutabilité pleine des négatifs, on peut permuter vers le bas toutes les règles négatives de telle sorte que $\mathcal{D}$ est transformée en une preuve $\mathcal{D}'$ où la partie inférieure de l'arbre n'est constituée que de règles négatives jusqu'au point où les branches de l'arbre atteignent des séquents positifs ou atomiques, racines de sous-preuves $\mathcal{D}_i$, on applique alors le processus de pré-focalisation aux $\mathcal{D}_i$;
2. **Phase positive :** Si $\mathcal{S}$ est positif, le graphe de focalisation associé nous assure de l'existence d'une source, disons P, que l'on sélectionne comme foyer. Grâce à la permutabilité faible des positifs, on peut amener les règles d'inférences positives dont les formules principales sont les $\prec$-sous-occurrences de P à permuter vers le bas de telle sorte que $\mathcal{D}$ est transformée en un arbre $\mathcal{D}'$ tel que le préfixe maximal ne contenant que des règles appliquées à P et à ses $\prec$-sous-formulas positives décompose P jusqu'à des sous-formules négatives ou atomiques. Il nous reste donc seulement des séquents négatifs ou atomiques ou des séquents positifs dans lesquels les sous-formules de P sont des atomes ;
3. **Cas atomique :** Si $\mathcal{S}$ est atomique, on termine puisque $\mathcal{D}$ étant une preuve sans coupure, elle ne peut que consister en une règle initiale qui est de la forme requise par la pré-focalisation.

Le processus de pré-focalisation d'une preuve MALL sans coupure est clairement terminant puisque les utilisations récursives du processus sont appliquées à des séquents possédant strictement moins de connecteurs que le séquent de départ.

□

Nous avons appelé le processus précédent ***processus de pré-focalisation*** car il ne correspond pas exactement à la discipline de focalisation considérée par Andreoli au niveau du traitement des atomes vis-à-vis de la focalisation : les atomes ne sont en effet par intégrés dans le processus de focalisation. Nous allons voir dans le paragraphe suivant comment adapter de manière très simple notre méthode pour obtenir précisément le système d'Andreoli.

9.5. Traitement des atomes

Traitement des littéraux dans Σ_3 La discipline de focalisation considérée par Andreoli grâce au système Σ_3 est légèrement différente de ce que nous avons considéré jusque là puisqu'il contraint également l'utilisation de la règle initiale, ce que la pré-focalisation ne fait pas.

Dans Σ_3 en effet, la règle initiale se trouve présentée en deux versions[1], $[I_1]$ et $[I_2]$. On remarque que la règle intiale ne peut être appliquée que durant une phase de focalisation et sur des ***littéraux positifs*** :

$$\frac{}{\vdash \Psi \colon X^\perp \Downarrow X}\,[I_1] \qquad\qquad \frac{}{\vdash \Psi, X^\perp \colon \cdot \Downarrow X}\,[I_2]$$

où X est un littéral positif.

Par ailleurs, la règle permettant de sortir d'une phase focalisée :

$$\frac{\vdash \Psi \colon \Delta \Uparrow F}{\vdash \Psi \colon \Delta \Downarrow F}\, R\Downarrow$$

n'est applicable que lorsque la formule F est négative.

En particulier, le séquent $\vdash a^\perp \oplus \mathbf{0}, \mathbf{0} \oplus a$ n'aurait qu'une seule preuve focalisée[2] :

$$\dfrac{\dfrac{\dfrac{}{\vdash a^\perp, a}\,ini}{\vdash a^\perp, \mathbf{0} \oplus a}\,[\oplus_1]}{\vdash a^\perp \oplus \mathbf{0}, \mathbf{0} \oplus a}\,[\oplus_0]$$

[1] La règle $[I_2]$ n'a de sens que dans le cas où l'on traite les exponentiels, ce que nous ferons dans la prochaine section.

[2] Dans Σ_3, cette preuve serait : $\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash \cdot \colon a^\perp \Downarrow a}\,[I_1]}{\vdash \cdot \colon a^\perp \Downarrow \mathbf{0} \oplus a}\,[\oplus_1]}{\vdash \cdot \colon \mathbf{0} \oplus a, a^\perp \Uparrow \cdot}\,D_1}{\vdash \cdot \colon \mathbf{0} \oplus a \Uparrow a^\perp}\,R\Uparrow}{\vdash \cdot \colon \mathbf{0} \oplus a \Downarrow a^\perp}\,R\Downarrow}{\vdash \cdot \colon \mathbf{0} \oplus a \Downarrow a^\perp \oplus \mathbf{0}}\,[\oplus_0]}{\vdash \cdot \colon a^\perp \oplus \mathbf{0}, \mathbf{0} \oplus a \Uparrow \cdot}\,D_1}{\vdash \cdot \colon a^\perp \oplus \mathbf{0} \Uparrow \mathbf{0} \oplus a}\,R\Uparrow}{\vdash \cdot \colon \cdot \Uparrow a^\perp \oplus \mathbf{0}, \mathbf{0} \oplus a}\,R\Uparrow$

alors que la pré-focalisation que nous avons vu précédemment permet de construire une deuxième preuve pré-focalisée en plus de la preuve ci-dessus :

$$\dfrac{\dfrac{\dfrac{}{\vdash a^{\perp}, a}\ ini}{\vdash a^{\perp} \oplus \mathbf{0}, a}\ [\oplus_0]}{\vdash a^{\perp} \oplus \mathbf{0}, \mathbf{0} \oplus a}\ [\oplus_1]$$

En effet, le graphe de focalisation de n'importe quelle preuve sans coupure de $\vdash a^{\perp} \oplus \mathbf{0}, \mathbf{0} \oplus a$ aurait deux sources et aucune arête, permettant dans le processus de pré-focalisation décrit par la preuve du théorème 9.23 de choisir n'importe quel foyer.

Le système d'Andreoli ajoute donc des contraintes supplémentaires à la recherche de preuve tout en conservant la complétude du système focalisé.

Nous allons maintenant voir que notre technique se généralise naturellement de manière à permettre de capturer exactement la discipline de focalisation d'Andreoli ainsi qu'une série de disciplines de focalisation qui ont des manières de traiter les littéraux différentes. Nous introduisons pour cela les assignations de biais de manière à traiter ce cas.

Quel intérêt à un traitement des littéraux ? Les littéraux sont par essence des formules inconnues, formules non encore instanciées ou formules dont la structure n'a pas vocation a être utilisée dans le cours de la preuve.

On peut se demander quel intérêt il y a à étudier la focalisation au sujet des littéraux. Nous tentons de donner quelques explications à ce sujet :

- Tout d'abord, il est certain que le cœur des résultats de focalisation ne réside pas dans la question des formules atomiques, mais bien dans les interactions entre les différentes règles du calcul des séquents de LL. En effet, une règle initiale peut toujours être expansée et, dans des formalismes comme la ludique, on peut même se passer de la règle initiale en considérant des η-expansions infinies de l'axiome ;
- Comme nous allons le voir, la contrainte d'Andreoli (focalisation sur les littéraux positifs) n'est que l'une des multiples contraintes que l'on peut appliquer à la règle initiale de manière à obtenir un système focalisé complet.
- Le fait d'imposer une contrainte sur l'utilisation de la règle initiale comme le fait Andreoli permet de réduire encore l'espace de recherche de preuve. Cela peut être intéressant du point de vue du quotient de l'espace des preuves de LL que l'on obtient ainsi et cela peut être intéressant pour rechercher des preuves de manière plus efficace ;
- Il faudra tout de même se demander, dans cette optique, si la discipline consistant à imposer la focalisation sur les atomes positifs est nécessairement la discipline optimale. Si l'on considère par exemple les clauses de Horn du programme logique spécifiant la suite de Fibonacci $(\mathit{fibo}_n)_{n\in\mathbb{N}}$:

$$\boxed{\begin{array}{ll} \mathit{fib}(0,0). & \\ \mathit{fib}(1,1). & \\ \forall n.\forall x.\forall y. & \mathit{fib}(n,x) \Rightarrow \\ & \mathit{fib}(n+1,y) \Rightarrow \mathit{fib}(n+2,x+y). \end{array}}$$

  On constate [LM07] que si tous les littéraux positifs reçoivent un biais négatif, alors $\mathit{fib}(n, \mathit{fibo}_n)$ ne possède qu'une preuve focalisée qui est exponentielle en n tandis que si les littéraux positifs reçoivent un biais positif, alors il existe plusieurs preuves focalisées, dont la plus courte est linéaire en n.

  Cette différence témoigne de la différence entre le raisonnement par chaînage arrière et le raisonnement par chaînage avant.

9.5.1 Assignation de biais

Dans la section où nous avons défini la pré-focalisation, les atomes n'étaient pas polarisés et n'intervenaient donc pas dans la construction du graphe de focalisation.

Nous allons maintenant étudier la manière dont le graphe de focalisation est affecté si nous assignons des polarités aux atomes, ce que nous appellerons dans ce cas des biais. Un atome qui reçoit une polarité positive (resp. négative) contribuera maintenant aux arêtes du graphe de focalisation comme toute formule positive (resp. négative).

La discipline d'Andreoli définie par le système Σ_3 correspond au fait de considérer comme positifs les littéraux positifs et comme négatifs les littéraux négatifs, mais nous allons considérer tout un ensemble d'autres possibilités dans lesquelles l'attribution d'une polarité à un atome peut éventuellement varier en fonction de l'occurrence de cet atome dans la preuve.

On commence en donnant une définition abstraite visant à définir l'ensemble des positions possibles pour un séquent dans un arbre de preuve d'une conclusion donnée :

Définition 9.24

Étant donné un séquent prouvable $\mathcal{S}$, on appelle $\mathcal{P}_\mathcal{S}$ (pour l'ensemble des positions disponibles de $\mathcal{S}$) l'ensemble contenant toutes les branches de tous les arbres de preuve possibles pour $\mathcal{S}$.

On écrira $\mathcal{O}_\mathcal{S}$ pour l'ensemble des occurrences de littéraux dans $\mathcal{S}$.

Définition 9.25 (*Assignation de biais $\mathcal{B}_\mathcal{S}$*)

Une assignation de biais pour un séquent prouvable $\mathcal{S}$, noté $\mathcal{B}_\mathcal{S}$, est une fonction partielle de $\mathcal{P}_\mathcal{S} \times \mathcal{O}_\mathcal{S}$ vers $\{-;+\}$.

On dira qu'une occurrence de littéral l a un biais positif (resp. négatif) dans un séquent d'une preuve $\mathcal{D}$ si l'assignation de biais est définie pour l dans ce séquent et vaut $+$ (resp. $-$). Si l'assignation de biais n'est pas définie, on dira que le littéral est neutre dans ce séquent.

Nous donnons ici quelques exemples d'assignations de biais typiques :

$$
\dfrac{\dfrac{\dfrac{}{\vdash \underline{s}, s^{\perp}}\,ini \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\vdash \underline{q}, q^{\perp}}\,ini \qquad \dfrac{}{\vdash \underline{r}, r^{\perp}}\,ini}{\vdash \underline{q \otimes r}, q^{\perp}, r^{\perp}}\,\otimes}{\vdash \underline{p \oplus (q \otimes r)}, q^{\perp}, r^{\perp}}\,\oplus}{\vdash p \oplus (q \otimes r), q^{\perp} \parr r^{\perp}}\,\parr}{\vdash p \oplus (q \otimes r), \underline{s \otimes (q^{\perp} \parr r^{\perp})}, s^{\perp}}\,\otimes \qquad \dfrac{}{\vdash \underline{\mathbf{1}}}\,\mathbf{1}}{\vdash p \oplus (q \otimes r), s \otimes (q^{\perp} \parr r^{\perp}), \underline{s^{\perp} \otimes \mathbf{1}}}\,\otimes
$$

FIG. 9.4 – Preuve focalisée correspondant à la preuve de la figure 1.17.

Exemple 9.26

- *L'assignation de biais qui n'est définie* **nulle part** *correspond à la situation de la partie précédente.*
- **L'assignation de biais d'Andreoli**, *$\mathcal{B}^{\Sigma_3}$, est la fonction définie comme : pour chaque atome a, $\mathcal{B}^{\Sigma_3}(\_, a) = +$ et $\mathcal{B}^{\Sigma_3}(\_, a^{\perp}) = -$.*
  *Plus généralement, l'assignation de biais peut ne pas dépendre de la première composante et donner la même polarité à différentes occurrences du même littéral. On parlera dans ce cas d'une assignation de biais «***atom-based***».*
- *On peut considérer des assignations de biais qui dépendent de la position de littéral considéré dans l'arbre. Pour de tels assignations de biais b, $b(p, a)$ peut être différent de $b(q, a)$. Dans ce cas on parle d'une assignation de biais «***occurrence-based***».*
  On peut considérer diverses conditions de cohérence pour les assignations.
  Par exemple, on peut demander qu'en traversant l'arbre de démonstration le long d'une branche à partir de la racine, la polarité ne change pas une fois qu'elle est définie : si p et q sont deux branches, p étant une extension de q et si $b(q, a) \searrow$ alors $b(p, a) \searrow$ alors $b(p, a) = b(q, a)$.
  On peut aussi considérer des assignations de biais totalement arbitraires, mais il faudra faire attention à leur cohérence vis-à-vis de la permutation de règles d'inférence au cours du processus de focalisation.

Définition 9.27 (*Les $\mathcal{B}$-Graphes de Focalisation*)

Étant donné un séquent positif $\mathcal{S}$, une preuve $\mathcal{D}$ de $\mathcal{S}$ et une assignation de biais $\mathcal{B}$ pour $\mathcal{S}$, on définit le **$\mathcal{B}$-Graphe de Focalisation** *$\mathcal{G}_{\mathcal{S}}^{\mathcal{B}}$ comme dans la sous-section précédente (les sommets sont les formules actives du tronc positif et les arêtes sont définies grâce aux formules apparaissant dans un séquent de la frontière) mais en considérant comme négatifs les littéraux à biais négatif et comme positifs les littéraux à biais positif.*

Les littéraux neutres sont traités comme auparavant : ils ne contribuent pas au graphe.

L'assignation de biais a pour conséquence qu'il y a plus d'arêtes dans le $\mathcal{B}$-Graphe de Focalisation. Par exemple, avec $\mathcal{B}^{\Sigma_3}$, le graphe de focalisation de notre exemple de la figure 1.17 est le suivant :

$$s^{\perp} \otimes \mathbf{1} \longrightarrow s \otimes (q^{\perp} \parr r^{\perp}) \longrightarrow p \oplus (q \otimes r)$$

Nous allons étudier dans la section suivante l'acyclicité des $\mathcal{B}$-graphes de focalisation.

9.5.2 Focalisation à la Andreoli

Proposition 9.28

Étant donné un séquent positif $\mathcal{S}$ et une preuve $\mathcal{D}$ de $\mathcal{S}$, le $\mathcal{B}^{\Sigma_3}$-Graphe de Focalisation $\mathcal{G}_{\mathcal{S}}^{\mathcal{B}^{\Sigma_3}}$ est acyclique.

Démonstration : On vérifie simplement que l'ajout d'arêtes supplémentaires n'a aucun effet sur les arguments qu'on utilisait pour la preuve précédente de la proposition 9.21 puisqu'ils ne concernaient que la structure de partitionnement des contextes pour les règles branchantes. Dès que les arêtes ne relient que des formules du séquent conclusion dont des sous-occurrences apparaissent dans un même séquent de la frontière, on est assuré de l'acyclicité.

□

La proposition précédente peut même s'étendre comme suit, la démonstration ci-dessus étant valide dans ce cas plus général :

Proposition 9.29

Étant donné un séquent positif $\mathcal{S}$ et une preuve $\mathcal{D}$ de $\mathcal{S}$, quelle que soit l'assignation de biais $\mathcal{B}$ que l'on choisit, le $\mathcal{B}$-Graphe de Focalisation $\mathcal{G}_{\mathcal{S}}^{\mathcal{B}}$ est acyclique.

Nous pouvons maintenant définir les preuves focalisées de MALL et énoncer le résultat principal concernant la focalisation :

Définition 9.30 (*Preuve Focalisée de MALL*)

Une preuve $\mathcal{D}$ MALL sans coupure d'un séquent $\mathcal{S}$ est **focalisée** *si, pour tout séquent $\mathcal{S}'$ de $\mathcal{D}'$:*
- *Si $\mathcal{S}'$ est un séquent négatif, $\mathcal{S}$ est conclusion d'une règle négative ;*
- *Si $\mathcal{S}'$ est un séquent prémisse d'une règle positive r de formule principale F, alors la règle positive r' dont $\mathcal{S}'$ est conclusion a pour formule principale une sous-occurrence de F, à moins que cette sous-occurrence ne soit un littéral négatif.*
- *Si $\mathcal{S}'$ est atomique, on lui applique une règle initiale.*

Remarque 9.31

La définition précédente impose en particulier que si $\mathcal{S}$ est un séquent prémisse d'une règle positive r de formule principale F telle que la sous-occurrence de F dans $\mathcal{S}$ est un littéral positif, alors $\mathcal{S}$ est un séquent atomique conclusion d'une règle initiale.

Remarque 9.32

On présente en figure 9.5 une adaptation du système focalisé d'Andreoli pour MALL. À condition de traiter la portion L de droite des séquents de type $\vdash \Delta \Uparrow L$ comme une liste non-ordonnée de formules, les preuves focalisées de la définition précédente peuvent être annotées (en insérant des inférences de la forme de type $R \Uparrow$ ou $R \Downarrow$ quand c'est nécessaire) de manière à satisfaire les contraintes du système de la figure 9.5.

Théorème 9.33 ($\mathcal{B}^{\Sigma_3}$*-Focalisation pour MALL*)

Si $\mathcal{D}$ est une preuve MALL sans coupure d'un séquent $\mathcal{S}$, elle peut être transformée, par un processus de permutation de règles d'inférence appelé focalisation, en une preuve focalisée $\mathcal{D}'$.

Démonstration : Le processus de focalisation est identique au processus de pré-focalisation de la section précédente, à ceci près qu'on utilise maintenant les $\mathcal{B}^{\Sigma_3}$-graphes de focalisation pour choisir une source.

$$\frac{\vdash \Delta \Uparrow L}{\vdash \Delta \Uparrow \bot, L}\ \bot \qquad \frac{\vdash \Delta \Uparrow F, G, L}{\vdash \Delta \Uparrow F \parr G, L}\ \parr \qquad \frac{}{\vdash \Delta \Uparrow \top, L}\ \top \qquad \frac{\vdash \Delta \Uparrow F, L \quad \vdash \Delta \Uparrow G, L}{\vdash \Delta \Uparrow F \,\&\, G, L}\ \&$$

$$\frac{}{\vdash \cdot \Downarrow \mathbf{1}}\ \mathbf{1} \qquad \frac{\vdash \Delta_1 \Downarrow F \quad \vdash \Delta_2 \Downarrow G}{\vdash \Delta_1, \Delta_2 \Downarrow F \otimes G}\ \otimes \qquad \frac{\vdash \Delta \Downarrow F_0}{\vdash \Delta \Downarrow F_0 llplus F_1}\ [\oplus_0] \qquad \frac{\vdash \Delta \Downarrow F_1}{\vdash \Delta \Downarrow F_0 \oplus F_1}\ [\oplus_1]$$

$$\frac{\vdash \Delta, F \Uparrow L}{\vdash \Delta \Uparrow F, L}\ R\Uparrow \qquad \frac{\vdash \Delta \Uparrow F}{\vdash \Delta \Downarrow F}\ R\Downarrow \qquad \frac{}{\vdash X^{\bot} \Downarrow X}\ I \qquad \frac{\vdash \Delta \Downarrow F}{\vdash \Delta, F \Uparrow \cdot}\ D$$

FIG. 9.5 – Le système Σ_3 pour MALL. *Les contraintes sur les règles sont les suivantes : dans $R \Uparrow$, F est positive ou bien un littéral négatif tandis que dans $R \Downarrow$, F est négative. Dans I, X est un littéral positif. Dans D, F n'est pas négative.*

La polarisation des atomes de $\mathcal{B}^{\Sigma_3}$ assure que la contrainte sur les règles initiales est vérifiée. En effet, supposons qu'une formule F d'un séquent positif $\mathcal{S}$ soit une source du $\mathcal{B}^{\Sigma_3}$-graphe de focalisation $\mathcal{G}$ et ait une sous-occurrence a dans un séquent $\mathcal{S}'$ de la frontière du graphe qui est un littéral positif alors dans ce cas, le séquent $\mathcal{S}'$ est forcément atomique (sinon, il est conclusion d'une règle négative et en conséquence il y a une formule négative en conclusion ce qui contredit le fait que F soit une source de $\mathcal{G}$) et de plus, la formule $a^{\bot}$ est déjà présente dans la conclusion du tronc positif (sinon, elle serait sous-occurrence d'une formule active G du tronc positif et on aurait donc une arête de G vers F dans le graphe).

Il s'agit enfin simplement de vérifier que la permutation des positifs préserve la condition de polarité des positifs. Ceci n'est pas forcément le cas avec une assignation de biais arbitraire. Comme il s'agit d'une assignation de biais atom-based, la permutation vers le bas ne pose pas de problème et la commutation vers le bas de toutes les règles du tronc positif dont la formule principale est une sous-occurrence de F aboutit au fait que la règle initiale est bien appliquée au cours d'une phase focalisée produisant la formule a.

$\square$

Comme on l'a vu dans la preuve, l'acyclicité du graphe qui est vraie quelle que soit l'assignation de biais ne suffit pas pour avoir une procédure de focalisation qui prenne en compte les atomes : encore faut-il que la phase de permutation vers le bas des règles positives ne modifie pas les polarités des atomes. Certaines assignations de biais peuvent mener à des incohérences. Nous étudions maintenant quelques assignations de biais qui permettent, ou non, de définir une discipline de focalisation.

Assignations de biais atom-based. Toutes les assignations de biais aux atomes (c'est-à-dire qui donne toujours la même polarité à un atome quelle que soit sa position) permettent de définir une focalisation par la méthode des graphes de focalisation : la preuve du théorème 9.33 fonctionne directement dans ce cadre.

Assignation d'un biais négatif au premier atome rencontré. On s'intéresse à l'assignation de biais qui assigne un biais négatif au premier biais rencontré lors de la construction de preuve.

Définition 9.34 ($\mathcal{B}^-$)

Étant donné un séquent $\mathcal{S}$, on définit l'assignation de biais $\mathcal{B}^-$ comme suit :

- *$\mathcal{B}^-(\epsilon, a)$ n'est défini nulle part ;*
- *si $p \in \mathcal{P}_\mathcal{S}$, $\mathcal{B}^-(p,a)$ est défini si, et seulement si, $\mathcal{B}^-(p,a^\perp)$ est défini avec la polarité opposée ;*
- *si $p \cdot \mathcal{S}' \in \mathcal{P}_\mathcal{S}$, si $\mathcal{B}^-(p,a) = +$ (resp. $\mathcal{B}^-(p,a) = -$), alors $\mathcal{B}^-(p \cdot \mathcal{S}', a) = +$ (resp. $\mathcal{B}^-(p \cdot \mathcal{S}', a) = -$) ;*
- *si $p \cdot \mathcal{S}' \in \mathcal{P}_\mathcal{S}$, $\mathcal{B}^-(p,a)$ et $\mathcal{B}^-(p,a^\perp)$ ne sont pas définis :*
  - *si $a^\perp \in \mathcal{S}'$, alors $\mathcal{B}^-(p \cdot \mathcal{S}', a) = +$;*
  - *sinon, si a a une occurrence dans $\mathcal{S}'$, alors $\mathcal{B}^-(p \cdot \mathcal{S}', a) = -$;*
  - *sinon, les atomes $a, a^\perp$ n'ont pas d'occurrence dans $\mathcal{S}'$, et $\mathcal{B}^-(p \cdot \mathcal{S}', a)$ et $\mathcal{B}^-(p \cdot \mathcal{S}', a^\perp)$ ne sont pas définis.*

On sait par la proposition 9.29 que le $\mathcal{B}^-$-graphe de focalisation est acyclique.

Nous vérifions maintant que la procédure de focalisation produit une preuve satisfaisant les contraintes souhaitées :

Proposition 9.35

Soit $\mathcal{S}$ un séquent positif, $\mathcal{D}$ une preuve de $\mathcal{S}$ et F une source du graphe de focalisation associé à $\mathcal{D}^+$. La preuve obtenue en faisant permuter vers le bas les règles d'inférences de $\mathcal{D}^+$ dont les formules principales sont des sous-occurrences de F produit une preuve qui focalise sur F.

Démonstration : On doit simplement vérifier que la condition sur les règles initiales est vérifiée. Supposons qu'un séquent $\mathcal{S}_{ini} = \vdash a, a^\perp$ est dans la frontière de $\mathcal{D}^+$ et supposons, par exemple, que a est une sous-occurrence de F (le cas où ni a, ni $a^\perp$ n'est sous-occurrence de F ne nous intéresse pas). Puisque F est une source du graphe, de deux choses l'une :

- soit $\mathcal{B}^-(p \cdot \mathcal{S}_{ini}, a) = -$;
- soit $\mathcal{B}^-(p \cdot \mathcal{S}_{ini}, a) = +$ et $a^\perp \in \mathcal{S}$.

Dans le premier cas, cela signifie que l'atome a est apparu dans un séquent de $\mathcal{D}^+$ plus bas que $a^\perp$ (en effet, puisqu'on est dans un tronc positif, il n'est pas possible que les deux atomes apparaissent dans le même séquent) auquel cas, la permutation vers le bas des règles appliquées aux sous-occurrences de F maintient cet invariant de la preuve. Dans le second cas, l'inférence de formule principale la sous-occurrence de F qui crée a, quelles que soient les permutations qu'on leur applique, ne permutera jamais sous l'inférence créant la formule $a^\perp$ (puisque cette dernière est déjà présente dans la racine du tronc positif), et l'invariant est également maintenu.

□

Cette assignation de biais nous amène à un système de preuve de la forme de la figure 9.6. Les assignations $\mathcal{B}_1^-$ et $\mathcal{B}_2^-$ sont obtenues en suivant les conditions de la définition 9.34.

Ce système présente l'intérêt, en recherche de preuve, de ne pas avoir à backtracker lorsque le seul élément problématique pour obtenir une preuve dans Σ_3 est la polarité des atomes : en effet, tant que la négation d'un atome a n'apparaît pas dans un séquent, une phase focalisée qui aboutirait sur un a ne peut pas utiliser la règle initiale. Attribuer la polarité négative au premier atome introduit dans le séquent permet de limiter cet effet de mauvais choix de foyer.

Assignation de biais qui attribue un biais positif au permier atome rencontré. Ce cas est l'inverse du précédent et nous le mentionnons pour illustrer un cas où l'on n'obtient pas une discipline de focalisation.

$$\frac{\vdash_{\mathcal{B}^-} \Delta \Uparrow L}{\vdash_{\mathcal{B}^-} \Delta \Uparrow \bot, L}\ \bot \qquad \frac{\vdash_{\mathcal{B}_1^-} \Delta \Uparrow F, G, L}{\vdash_{\mathcal{B}^-} \Delta \Uparrow F \parr G, L}\ \parr$$

$$\frac{}{\vdash_{\mathcal{B}^-} \Delta \Uparrow \top, L}\ \top \qquad \frac{\vdash_{\mathcal{B}_1^-} \Delta \Uparrow F, L \quad \vdash_{\mathcal{B}_2^-} \Delta \Uparrow G, L}{\vdash_{\mathcal{B}^-} \Delta \Uparrow F \,\&\, G, L}\ \&$$

$$\frac{}{\vdash_{\mathcal{B}^-} \cdot \Downarrow \mathbf{1}}\ \mathbf{1} \qquad \frac{\vdash_{\mathcal{B}_1^-} \Delta_1 \Downarrow F \quad \vdash_{\mathcal{B}_2^-} \Delta_2 \Downarrow G}{\vdash_{\mathcal{B}^-} \Delta_1, \Delta_2 \Downarrow F \otimes G}\ \otimes$$

$$\frac{\vdash_{\mathcal{B}_1^-} \Delta \Downarrow F_0}{\vdash_{\mathcal{B}^-} \Delta \Downarrow F_0 \oplus F_1}\ [\oplus_0] \qquad \frac{\vdash_{\mathcal{B}_1^-} \Delta \Downarrow F_1}{\vdash_{\mathcal{B}^-} \Delta \Downarrow F_0 \oplus F_1}\ [\oplus_1]$$

$$\frac{\vdash_{\mathcal{B}^-} \Delta, F \Uparrow L}{\vdash_{\mathcal{B}^-} \Delta \Uparrow F, L}\ R\Uparrow \qquad \frac{\vdash_{\mathcal{B}^-} \Delta \Uparrow F}{\vdash_{\mathcal{B}^-} \Delta \Downarrow F}\ R\Downarrow \qquad \frac{}{\vdash_{\mathcal{B}^-} X^\perp \Downarrow X}\ I \qquad \frac{\vdash_{\mathcal{B}^-} \Delta \Downarrow F}{\vdash_{\mathcal{B}^-} \Delta, F \Uparrow \cdot}\ D$$

FIG. 9.6 – Le système Σ_3 pour MALL avec assignation de biais négatif au premier atome rencontré. *Les contraintes sur les règles sont les suivantes : dans $R \Uparrow$, F est positive ou bien un littéral négatif tandis que dans $R \Downarrow$, F est négative. Dans I, X est un littéral positif. Dans D, F n'est pas négative.*

Définition 9.36 ($\mathcal{B}^+$)

Étant donné un séquent $\mathcal{S}$, on définit l'assignation de biais $\mathcal{B}^+$ comme suit :
- *$\mathcal{B}^+(\epsilon, a)$ n'est défini nulle part ;*
- *si $p \in \mathcal{P}_\mathcal{S}$, $\mathcal{B}^+(p, a)$ est défini si, et seulement si, $\mathcal{B}^+(p, a^\perp)$ est défini avec la polarité opposée ;*
- *si $p \cdot \mathcal{S}' \in \mathcal{P}_\mathcal{S}$, si $\mathcal{B}^+(p, a) = +$ (resp. $\mathcal{B}^+(p, a) = -$), alors $\mathcal{B}^+(p \cdot \mathcal{S}', a) = +$ (resp. $\mathcal{B}^+(p \cdot \mathcal{S}', a) = -$) ;*
- *si $p \cdot \mathcal{S}' \in \mathcal{P}_\mathcal{S}$, $\mathcal{B}^-(p, a)$ et $\mathcal{B}^+(p, a^\perp)$ ne sont pas définis :*
  - *si $a^\perp \in \mathcal{S}'$, alors $\mathcal{B}^+(p \cdot \mathcal{S}', a) = -$;*
  - *sinon, si a a une occurrence dans $\mathcal{S}'$, alors $\mathcal{B}^+(p \cdot \mathcal{S}', a) = +$;*
  - *sinon, les atomes $a, a^\perp$ n'ont pas d'occurrence dans $\mathcal{S}'$, ainsi que $\mathcal{B}^+(p \cdot \mathcal{S}', a)$ et $\mathcal{B}^+(p \cdot \mathcal{S}', a^\perp)$ ne sont pas définis.*

Un contre-exemple à la focalisation dans ce cas est le séquent considéré au début de cette section $\vdash a^\perp \oplus \mathbf{0}, \mathbf{0} \oplus a$ qui n'a pas de preuve focalisée pour $\mathcal{B}^+$: en effet, aucune des deux preuves :

$$\mathcal{D}_1 = \dfrac{\dfrac{\dfrac{}{\vdash a^\perp, a}\ ini}{\vdash a^\perp, \mathbf{0} \oplus a}\ [\oplus_1]}{\vdash a^\perp \oplus \mathbf{0}, \mathbf{0} \oplus a}\ [\oplus_0] \qquad \text{et} \qquad \mathcal{D}_2 = \dfrac{\dfrac{\dfrac{}{\vdash a^\perp, a}\ ini}{\vdash a^\perp \oplus \mathbf{0}, a}\ [\oplus_0]}{\vdash a^\perp \oplus \mathbf{0}, \mathbf{0} \oplus a}\ [\oplus_1]$$

ne vérifie la discipline de focalisation puisqu'après la première instance d'une règle $[\oplus_i]$, on obtient un atome de polarité positive et qu'aucune règle initiale n'est applicable à ce point.

Ce dernier fait est mis en évidence en considérant le graphe de focalisation associé à chacune des deux preuves considérées :

$$\mathcal{G}_{\mathcal{D}_1} : \quad a^\perp \oplus \mathbf{0} \longleftarrow \mathbf{0} \oplus a \qquad \text{et} \qquad \mathcal{G}_{\mathcal{D}_2} : \quad a^\perp \oplus \mathbf{0} \longrightarrow \mathbf{0} \oplus a$$

$$\dfrac{\dfrac{\vdash A, C[F/X], \Gamma}{\vdash A, \exists^2 X.C, \Gamma}\,\exists^2}{\vdash A \oplus B, \exists^2 X.C, \Gamma}\,\oplus \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A, C[F/X], \Gamma}{\vdash A \oplus B, C[F/X], \Gamma}\,\oplus}{\vdash A \oplus B, \exists^2 X.C, \Gamma}\,\exists^2$$

$$\dfrac{\dfrac{\vdash A, C[F/X], \Gamma}{\vdash A, \exists^2 X.C, \Gamma}\,\exists^2 \quad \vdash B, \Delta}{\vdash A \otimes B, \exists^2 X.C, \Gamma, \Delta}\,\otimes \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A, C[F/X], \Gamma \quad}{\vdash A \otimes B, C[F/X], \Gamma}\,\otimes \quad \vdash B, \Delta}{\vdash A \otimes B, \exists^2 X.C, \Gamma}\,\exists^2$$

$$\dfrac{\dfrac{\vdash A[F/X], B[G/Y], \Gamma}{\vdash A[F/X], \exists^2 Y.B, \Gamma}\,\exists^2}{\vdash \exists^2 X.A, \exists^2 Y.B, \Gamma}\,\exists^2 \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A[F/X], B[G/Y], \Gamma}{\vdash \exists^2 X.A, B[G/Y], \Gamma}\,\exists^2}{\vdash \exists^2 X.A, \exists^2 Y.B, \Gamma}\,\exists^2$$

FIG. 9.7 – Permutations parmi les connecteurs positifs $\exists/\oplus$, $\exists/\otimes$ et $\exists/\exists$.

Le choix de la seule source de $\mathcal{G}_{\mathcal{D}_1}$ comme foyer conduit à réécrire $\mathcal{D}_1$ en $\mathcal{D}_2$ tandis que le choix de la seule source de $\mathcal{G}_{\mathcal{D}_2}$ conduit à tranformer $\mathcal{D}_2$ en $\mathcal{D}_1$ et le processus ne termine donc pas.

9.6. Extension du résultat à tout LL

Notre analyse est pour le moment restreinte au cas des preuves sans coupure pour MALL propositionnelle.

Nous étendons dans cette section notre méthode des graphes de focalisation à toute la logique linéaire en montrant comment traiter les quantificateurs, la coupure et les exponentiels avant de présenter des extensions du résultat de focalisation dans la section suivante.

9.6.1 Les quantificateurs

La preuve que nous avons proposée dans la première partie de ce chapitre peut être directement étendue au cas des quantificateurs du premier ordre comme du second ordre :

Propriétés de permutabilité des quantificateurs. $\forall$ et $\forall^2$ sont ajoutés aux connecteurs négatifs tandis que $\exists$ et $\exists^2$ sont ajoutés aux connecteurs positifs. On conserve les mêmes propriétés de permutabilité :

- Nous montrons en figure 9.7 les permutations additionnelles entre les inférences positives provoquées par l'introduction des existentiels : $\exists/\oplus$, $\exists/\otimes$ et $\exists/\exists$. Les permutations sont identiques pour $\exists$ et $\exists^2$, nous ne montrons donc que le cas du second ordre ;
- Nous montrons en figure 9.8 les permutations additionnelles entre les inférences négatives provoquées par l'introduction des universels. Les permutations sont identiques pour $\forall$ et $\forall^2$, nous ne montrons donc que le cas du second ordre ;
- Nous montrons en figure 9.9 les permutations de l'universel sous les positifs (on peut, si besoin, renommer la variable X en une variable qui n'est libre ni dans Γ, ni dans A, ni dans B) ;
- Nous montrons en figure 9.10 les permutations des négatifs sous l'existentiel ;

La définition des troncs positifs, des graphes de focalisation et des preuves focalisées s'étend sans problème. L'acyclicité du graphe de focalisation est préservée par le fait que

$$\dfrac{\dfrac{\vdash A, B, C, \Gamma}{\vdash A, B, \forall^2 X.C, \Gamma}\ \forall^2}{\vdash A \parr B, \forall^2 X.C, \Gamma}\ \parr \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A, B, C, \Gamma}{\vdash A \parr B, C, \Gamma}\ \parr}{\vdash A \parr B, \forall^2 X.C, \Gamma}\ \forall^2$$

$$\dfrac{\dfrac{\vdash A, C, \Gamma}{\vdash A, \forall^2 X.C, \Gamma}\ \forall^2 \quad \dfrac{\vdash B, C, \Gamma}{\vdash B, \forall^2 X.C, \Gamma}\ \forall^2}{\vdash A \,\&\, B, \forall^2 X.C, \Gamma}\ \& \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A, C, \Gamma \quad \vdash A, C, \Gamma}{\vdash A \,\&\, B, C, , \Gamma}\ \&}{\vdash A \,\&\, B, \forall^2 X.C, \Gamma}\ \forall^2$$

$$\dfrac{\dfrac{\vdash A, B, \Gamma}{\vdash A, \forall^2 Y.B, \Gamma}\ \forall^2}{\vdash \forall^2 X.A, \forall^2 Y.B, \Gamma}\ \forall^2 \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A, B, \Gamma}{\vdash \forall^2 X.A, B, \Gamma}\ \forall^2}{\vdash \forall^2 X.A, \forall^2 Y.B, \Gamma}\ \forall^2$$

FIG. 9.8 – Permutations parmi les connecteurs négatifs.

$$\dfrac{\dfrac{\vdash A, C, \Gamma}{\vdash A, \forall^2 X.C, \Gamma}\ \forall^2\ (\star)}{\vdash A \oplus B, \forall^2 X.C, \Gamma}\ \oplus \quad\longrightarrow\quad \dfrac{\dfrac{\vdash A, C, \Gamma}{\vdash A \oplus B, C, \Gamma}\ \oplus}{\vdash A \oplus B, \forall^2 X.C, \Gamma}\ \forall^2\ (\star)$$

$$\dfrac{\dfrac{\vdash A, C, \Gamma}{\vdash A, \forall^2 X.C, \Gamma}\ \forall^2\ (\star) \quad \vdash B, \Delta}{\vdash A \otimes B, \forall^2 X.C, \Gamma, \Delta}\ \otimes \quad\longrightarrow\quad \dfrac{\dfrac{\vdash A, C, \Gamma}{\vdash A \otimes B, C, \Gamma}\ \otimes \quad \vdash B, \Delta}{\vdash A \otimes B, \forall^2 X.C, \Gamma}\ \forall^2\ (\star)$$

$$\dfrac{\dfrac{\vdash A[F/X], B, \Gamma}{\vdash A[F/X], \forall^2 Y.B, \Gamma}\ \forall^2\ (\star)}{\vdash \exists^2 X.A, \forall^2 Y.B, \Gamma}\ \exists^2 \quad\longrightarrow\quad \dfrac{\dfrac{\vdash A[F/X], B, \Gamma}{\vdash \exists^2 X.A, B, \Gamma}\ \exists^2}{\vdash \exists^2 X.A, \forall^2 Y.B, \Gamma}\ \forall^2\ (\star)$$

FIG. 9.9 – Permutations de l'universel sous les positifs.

---

les règles pour les quantificateurs sont non-branchantes, ce qui nous permet de conclure l'énoncé suivant :

Théorème 9.37 (*Focalisation pour MALL avec quantificateurs*)

Si $\mathcal{D}$ est une preuve MALL avec quantificateurs sans coupure d'un séquent $\mathcal{S}$, elle peut être transformée, par un processus de permutation de règles d'inférence, en une preuve focalisée $\mathcal{D}'$.

9.6.2 La coupure

Nous obtenons également facilement le résultat de focalisation pour la coupure. La simplicité de cette extension provient de la souplesse des graphes de focalisation qui nous permettent d'ajouter facilement une information utile à notre démarche.

Coupure et focalisation. Traiter la règle de coupure dans une analyse de la focalisation n'est pas une question essentielle lorsqu'on est simplement intéressé par les questions de complétude et de recherche de preuve. Mais puisque nous présentons une approche dynamique du processus de focalisation, il est naturel et important de prendre au sérieux la règle de coupure. Par exemple, on pourra s'intéresser à l'étude de la manière dont la réduction de coupure et le processus de focalisation interagissent.

$$\dfrac{\dfrac{\vdash A, B, C[F/X], \Gamma}{\vdash A \parr B, C[F/X], \Gamma}\ \parr}{\vdash A \parr B, \exists^2 X.C, \Gamma}\ \exists^2 \quad \longrightarrow \quad \dfrac{\dfrac{\vdash A, B, C[F/X], \Gamma}{\vdash A, B, \exists^2 X.C, \Gamma}\ \exists^2}{\vdash A \parr B, \exists^2 X.C, \Gamma}\ \parr$$

$$\dfrac{\dfrac{\vdash A, C[F/X], \Gamma \quad \vdash A, C[F/X], \Gamma}{\vdash A \,\&\, B, C[F/X], \Gamma}\ \&}{\vdash A \,\&\, B, \exists^2 X.C, \Gamma}\ \exists^2 \quad \longrightarrow \quad \dfrac{\dfrac{\vdash A, C[F/X], \Gamma}{\vdash A, \exists^2 X.C, \Gamma}\ \exists^2 \quad \dfrac{\vdash B, C[F/X], \Gamma}{\vdash B, \exists^2 X.C, \Gamma}\ \exists^2}{\vdash A \,\&\, B, \exists^2 X.C, \Gamma}\ \&$$

FIG. 9.10 – Permutations des négatifs sous l'existentiel.

La coupure dans Σ_3. Andreoli introduit donc une règle dans le système Σ_3 pour la coupure :

$$\dfrac{\vdash \Delta \Downarrow F \quad \vdash \Delta' \Downarrow F^{\perp}}{\vdash \Delta, \Delta' \Uparrow \cdot}\ cut$$

La coupure initie donc deux phases de focalisation, l'une sur F, l'autre sur $F^{\perp}$. Bien sûr, si l'une des deux formules est positive, l'autre est négative et la phase de focalisation la concernant s'achève immédiatement grâce à une règle $R \Downarrow$.

En introduisant cette règle, l'idée d'Andreoli consiste à remarquer que la règle de coupure est très similaire à une règle $\otimes$: remplacer une règle de coupure sur A dans une preuve $\mathcal{D}$ de $\vdash \Gamma$ par une règle $\otimes$ produit *presque* une preuve de $\vdash \Gamma, A \otimes A^{\perp}$. Il ne s'agit pas exactement d'une preuve puisque la règle $[\&]$, la règle $[!]$ et la règle $[\forall]$ et $\left[\forall^2\right]$ peuvent poser problème. Andreoli résout ce problème en considérant la formule $?A \otimes A^{\perp}$ au lieu de $A \otimes A^{\perp}$ ce qui résoud le problème pour $\&$ et $!$ mais n'est pas suffisant pour le quantificateur $\forall$: il faudrait en effet clore la formule universellement pour avoir une solution tout à fait complète.

Coupure et graphes de focalisation. Dans notre cadre, nous obtiendrons bien une preuve de $\vdash \Gamma, A \otimes A^{\perp}$: nous ne sommes en effet intéressés que par les règles de coupure qui s'appliquent au sein d'un tronc positif. On vérifiera facilement que si $\mathcal{D}^{+}$ est un tronc positif pour $\vdash \Gamma$ contenant une règle de coupure sur A alors le remplacement de la coupure par une règle tenseur sur $A \otimes A^{\perp}$ conduit immédiatement à un tronc positif $\mathcal{D}'^{+}$ pour $\vdash \Gamma, A \otimes A^{\perp}$ puisque les trois inférences problématiques du paragraphe précédent ne peuvent pas apparaître dans un tronc positif.

Nous n'avons même pas besoin d'utiliser la preuve elle-même de $\vdash \Gamma, A \otimes A^{\perp}$[3]. Nous allons simplement utiliser cette analogie de manière à trouver comment adapter le Graphe de Focalisation aux preuves avec coupures.

L'analogie entre la coupure et la règle $[\otimes]$ suggère simplement de traiter la règle de coupure comme une règle positive. En particulier, on constate que la coupure satisfait les mêmes propriétés de permutation avec les règles positives que la règle $[\otimes]$. En conséquence, les troncs positifs peuvent contenir des règles de coupure et le graphe de focalisation aura de nouveaux sommets de la forme $Cut(A)$. On adapte de manière évidente la relation de sous-occurrence de manière à ce que les deux formules de coupure A et $A^{\perp}$ soient des sous-occurrence de $Cut(A)$. Les arêtes sont créées dans les mêmes conditions que précédemment.

La preuve de l'acyclicité des graphes de focalisation de la première section s'adapte immédiatement du fait du partitionnement du contexte dans la règle *cut* comme dans la règle $[\otimes]$.

[3]Ou plutôt le fait qu'il s'agisse bien d'un preuve.

On étend la notion de preuve focalisée :

Définition 9.38 (*Preuve focalisée avec coupure*)

Une preuve $\mathcal{D}$ de MALL avec coupure d'un séquent $\mathcal{S}$ est focalisée si elle remplit les conditions des définitions 9.22 et 9.30 et qu'en outre si $\mathcal{S}'$ est un séquent positif de $\mathcal{D}$, prémisse d'une coupure, il est nécessairement conclusion d'une règle d'inférence appliquée à la formule de coupure sauf s'il s'agit d'un littéral polarisé négativement.

et on a donc, grâce aux permutations des positifs :

Théorème 9.39

La méthode des graphes de focalisation produit des preuves focalisées à partir de preuves MALL avec coupures.

9.6.3 Les exponentiels

Nous complétons le tableau de la méthode des graphes de focalisation en expliquant comment traiter les exponentiels. La question de la focalisation pour les exponentiels soulève de nouvelles questions.

Toutes les inférences que nous avons considérées jusque là avaient des propriétés de permutabilité de deux sortes : permutabilités faibles pour les inférences de formule principale positive ainsi que pour la coupure et permutabilité pleine pour les inférences de formule principale négative. En particulier, toutes les règles associées à un connecteur donné avaient les mêmes propriétés de permutabilité. La situation des exponentiels est plus complexe car d'une part des règles structurelles peuvent s'appliquer aux formules qui ont «?» pour connecteur principal et d'autre part la règle de promotion suppose une synchronisation avec les formules contextuelles qui doivent être toutes préfixées par des «?».

Cela a pour conséquence qu'il n'est pas évident d'adapter directement le résultat de focalisation en utilisant la structure des séquents de LL. Nous avons alors la possibilité de suivre deux approches : d'une part considérer un calcul dyadique selon les termes d'Andreoli [And90] ou calcul mixte selon la terminologie de Girard [Gir06] dans lequel une partie du séquent est gérée de manière classique et non pas linéaire. L'autre approche consiste à rester dans les séquents de LL avec une caractérisation des preuves focalisées de LL qui prenne en compte les exponentiels.

Nous commençons par traiter l'approche dyadique avant de présenter les preuves focalisées du système linéaire usuel.

Discussion de la règle de promotion. La promotion a une forme particulière parce que, contrairement aux autres règles d'inférence du calcul des séquents, elle dépend de la structure de toutes les formules du contexte : l'une des formules du séquent doit avoir un «!» tandis que toutes les autres doivent avoir un «?» :

$$\frac{\vdash\ ?\Gamma, F}{\vdash\ ?\Gamma, !F}\ [!]$$

Cela indique qu'il y a un niveau particulier de connaissance à propos de la structure du séquent pour pouvoir appliquer la règle de promotion qui n'est pas ce qu'on utilise d'habitude en calcul des séquents (il s'agit de l'une des raisons pour laquelle on parle de modalités à propos des connecteurs exponentiels). C'est également reflété par la manière dont la règle du «!» est implémentée dans les systèmes de programmation en logique linéaire ou par la structure des boîtes exponentielles dans les réseaux de démonstration.

On voit en fait que deux sortes d'opérations sont effectuées avec la règle [!] :
– la classification de F comme une formule portant un «?» d'un côté,
– le retrait du «!» de $!G$ quand $!G$ est la seule formule qui n'a pas de «?».

Ceci peut être mis en évidence avec l'illustration suivante : *considérons* $\vdash ?\Gamma, F, !G$*, peut-on appliquer une règle «!» à ce séquent ?* Il y a deux manières de répondre à cette question : soit «cela dépend de F», soit «non, pas encore, il faut encore effectuer une action sur F». Les deux réponses témoignent de cette idée que la promotion ne peut être appliquée que si F est de la forme $?F'$ mais elle sont différentes d'un point de vue opérationnel : la seconde réponse suggère qu'il y a des actions à accomplir avant de pouvoir appliquer la règle de promotion. Ainsi, F devrait d'abord être reconnue comme étant de la forme $?F'$. Par ailleurs, cela suggère que $?\Gamma$ a déjà été reconnu comme portant le «?» : le travail a été effectué et cette information a été stockée quelque part dans la structure du séquent.

Cette remarque suggère d'introduire un contexte séparé qui stockera les formules que l'on a reconnues comme ayant un «?» comme connecteur principal : $\vdash \Gamma \mid \Delta$. Les deux opérations que l'on a discutées précédemment et la déréliction deviennent alors les règles d'inférence suivantes :

$$\frac{\vdash \Gamma, A \mid \Delta}{\vdash \Gamma \mid ?A, \Delta}[?] \qquad \frac{\vdash \Gamma \mid A}{\vdash \Gamma \mid !A}[!] \qquad \frac{\vdash \Gamma, A \mid A, \Delta}{\vdash \Gamma, A \mid \Delta}\, der$$

On doit bien sûr adapter de manière évidente les autres règles de MALL à la nouvelle structure des séquents pour obtenir le calcul dyadique.

Calcul dyadique. Le calcul dyadique profite de la structure hétérogène des séquents pour incorporer certaines règles structurelles applicables aux formules de la forme $?F$ lorsque ces formules ont été stockées dans la portion distinguée du séquent, dite à gestion classique, et qui n'est constituée que de formules préfixées d'un ? tandis que l'autre portion du séquent est traitée de manière habituelle (il y a une gestion linéaire de cette partie du séquent).

Ainsi, dans ce système, la règle initiale et la règle [**1**] incorporent l'affaiblissement pour le contexte à gestion classique :

$$\frac{}{\vdash \Gamma \mid a, a^{\perp}}\, ini \qquad \frac{}{\vdash \Gamma \mid \mathbf{1}}[\mathbf{1}]$$

tandis que la règle $[\otimes]$ et la coupure incorporent la contraction pour le contexte à gestion classique :

$$\frac{\vdash \Gamma \mid \Delta, F \quad \vdash \Gamma \mid \Delta', F'}{\vdash \Gamma \mid \Delta, \Delta', F \otimes F'}[\otimes] \qquad \frac{\vdash \Gamma \mid \Delta, F \quad \vdash \Gamma \mid \Delta', F^{\perp}}{\vdash \Gamma \mid \Delta, \Delta'}\, cut$$

Le système dyadique pour LL est présenté en figure 9.11.

Focalisation des exponentiels. Dans le système dyadique, on a donc décomposé les règles qui s'appliquent au connecteur «?» en deux : la nouvelle règle [?] sera considérée comme négative tandis que la déréliction sera considérée comme règle positive. La promotion a également un caractère bi-polaire : connecteur positif, il devra être appliqué dans une phase de focalisation, mais son caractère négatif fera cesser la phase de focalisation

$$\frac{}{\vdash \Sigma \mid A, A^{\perp}}\ ini \qquad \frac{\vdash \Sigma \mid \Gamma, A \quad \vdash \Sigma \mid \Delta, A^{\perp}}{\vdash \Sigma \mid \Gamma, \Delta}\ cut$$

$$\frac{\vdash \Sigma \mid \Gamma}{\vdash \Sigma \mid \Gamma, \perp}\ [\perp] \quad \frac{\vdash \Sigma \mid \Gamma, A, B}{\vdash \Sigma \mid \Gamma, A \parr B}\ [\parr] \quad \frac{}{\vdash \Sigma \mid \Gamma, \top}\ [\top] \quad \frac{\vdash \Sigma \mid \Gamma, A \quad \vdash \Sigma \mid \Gamma, B}{\vdash \Sigma \mid \Gamma, A \,\&\, B}\ [\&]$$

$$\frac{\vdash \Sigma \mid \Gamma, A \quad \vdash \Sigma \mid \Delta, B}{\vdash \Sigma \mid \Gamma, \Delta, A \otimes B}\ [\otimes] \quad \frac{}{\vdash \Sigma \mid \mathbf{1}}\ [\mathbf{1}] \quad \frac{\vdash \Sigma \mid \Gamma, A_1}{\vdash \Sigma \mid \Gamma, A_1 \oplus A_2}\ [\oplus_0] \quad \frac{\vdash \Sigma \mid \Gamma, A_2}{\vdash \Sigma \mid \Gamma, A_1 \oplus A_2}\ [\oplus_1]$$

$$\frac{\vdash \Sigma \mid \Gamma, A}{\vdash \Sigma \mid \Gamma, \forall x.A}\ [\forall] \qquad \frac{\vdash \Sigma \mid \Gamma, A[t/x]}{\vdash \Sigma \mid \Gamma, \exists x.A}\ [\exists]$$

$$\frac{\vdash \Sigma, A \mid \Gamma}{\vdash \Sigma \mid\ ?A, \Gamma}\ [?] \qquad \frac{\vdash \Sigma \mid A}{\vdash \Sigma \mid\ !A}\ [!] \qquad \frac{\vdash \Sigma, A \mid A, \Gamma}{\vdash \Sigma, A \mid \Gamma}\ der$$

Fig. 9.11 – Calcul des séquents dyadiques pour LL

immédiatement après. En effet, le séquent $\vdash\ ?(a \parr b), !(a^{\perp} \otimes b^{\perp})$ est un bon exemple de ce fait : la promotion doit être appliquée avant d'effectuer la déréliction sur $?(a \parr b)$ (sinon, on ne peut plus enlever le «!»), mais le tenseur ne peut pas être décomposé immédiatement après : il faut d'abord effecteur la déréliction et la règle $[\parr]$ avant de pouvoir appliquer $[\otimes]$.

La règle [?] permute au-dessous de toutes les règles du calcul pour lesquelles cela a du sens (c'est-à-dire les règles ayant un contexte linéaire, contrairement à $[\mathbf{1}], [!], ini$). La déréliction, *der*, permute avec toutes les règles positives considérées précédemment dans ce chapitre. La règle [!] a par contre des propriétés de permutabilité particulièrement limitées qui ne posent pourtant pas de problème. Son application requiert en effet que le contexte linéaire ne soit constitué que de la formule promue et elle conclut une phase focalisée. Nous donnons à la fin de cette section quelques éléments supplémentaires pour expliquer ceci.

On étend de manière évidente la notion de séquent positif, négatif ou neutre aux séquents dyadiques :

Définition 9.40 (*Types de séquents dyadiques*)

- *Un séquent dyadique* $\vdash \Sigma \mid \Gamma$ *est négatif si* $\vdash \Gamma$ *est négatif ;*
- *Un séquent dyadique* $\vdash \Sigma \mid \Gamma$ *est positif si* $\vdash \Gamma$ *est positif ou si* $\vdash \Gamma$ *est atomique et* $\Sigma \neq \emptyset$ *;*
- *Un séquent dyadique* $\vdash \Sigma \mid \Gamma$ *est atomique si* $\vdash \Gamma$ *est atomique.*

Remarque 9.41

Un séquent dyadique peut donc être à la fois atomique et positif, ce qui est naturel puisque le séquent $\vdash a \otimes a^\perp \mid a, a^\perp$ *possède les deux preuves sans coupure suivantes :*

$$\dfrac{}{\vdash a \otimes a^\perp \mid a, a^\perp}\, ini \qquad \dfrac{\dfrac{\dfrac{}{\vdash a \otimes a^\perp \mid a^\perp, a}\, ini \quad \dfrac{}{\vdash a \otimes a^\perp \mid a, a^\perp}\, ini}{\vdash a \otimes a^\perp \mid a, a^\perp, a \otimes a^\perp}\,[\otimes]}{\vdash a \otimes a^\perp \mid a, a^\perp}\, der$$

On peut alors étendre la définition des troncs positifs aux exponentiels :

Définition 9.42 (*Tronc positif exponentiel*)

Étant donné un séquent positif $\mathcal{S}$ *et une preuve* $\mathcal{D}$ *de* $\mathcal{S}$*, un tronc positif exponentiel (ou simplement tronc positif) pour* $\mathcal{S}$ *est un sous-arbre maximal* $\mathcal{D}^+$ *de* $\mathcal{D}$ *ne contenant que des règles positives et tel que si une promotion apparaît dans* $\mathcal{D}^+$*, alors la prémisse de la promotion est une feuille du tronc (les branches sont coupées aussitôt qu'une promotion est appliquée).*

Le fait de couper les branches du tronc dès qu'une promotion est rencontrée fait écho à la remarque précédente sur la bipolarité du connecteur «!». Nous reviendrons sur ce point à la fin de cette section.

Graphe de focalisation exponentiel. La règle de contraction qui est implémentée par la déréliction du système dyadique produit diverses occurrences de la même formule. Il nous faut pouvoir distinguer ces diverses occurrences dans la construction du graphe de focalisation puisque chacune de ces occurrences est suceptible de servir de foyer de focalisation.

On définit ainsi les formules actives d'un tronc positif comme suit :

Définition 9.43 (*Formules actives d'un tronc exponentiel*)

Étant donné un tronc positif exponentiel $\mathcal{D}^+$ *de conclusion* $\vdash \Sigma \mid \Gamma$*, on attribue à chaque occurrence de déréliction de* $\mathcal{D}^+$ *de manière unique un indice* i *et on désigne l'occurrence de la formule* A *ainsi créée par* (A, i)*. On étend de manière évidente la relation de sous-occurrence aux formules indexées.*

Les **formules actives** *de* $\mathcal{D}^+$ *sont les formules actives de* Γ *ainsi que les* (A, i) *comme défini ci-dessus.*

Exemple 9.44

La preuve suivante possède deux formules actives : $(a \otimes a^\perp, 1)$ *et* $(a \otimes a^\perp, 2)$

$$\boxed{\mathcal{D} = \dfrac{\dfrac{\dfrac{\dfrac{}{\vdash a \otimes a^\perp \mid a^\perp, a}\, ini \quad \dfrac{\dfrac{}{\vdash a \otimes a^\perp \mid a, a^\perp}\, ini \quad \dfrac{}{\vdash a \otimes a^\perp \mid a, a^\perp}\, ini}{\vdash a \otimes a^\perp \mid a, \underline{a \otimes_1 a^\perp}, a^\perp}\,[\otimes]}{\vdash a \otimes a^\perp \mid a, a^\perp, a \otimes_1 a^\perp, \underline{a \otimes_2 a^\perp}}\,[\otimes]}{\vdash \underline{a \otimes a^\perp} \mid a, a^\perp, a \otimes_1 a^\perp}\, der^2}{\vdash \underline{a \otimes a^\perp} \mid a, a^\perp}\, der^1}$$

Pour clarifier la preuve ci-dessus, nous avons attribué des indices aux diverses occurrences de la formule $a \otimes a^\perp$ *dans les parties linéaires des séquents de* $\mathcal{D}$*.*

Définition 9.45 (*Graphe de Focalisation Exponentiel*)

Soient $\vdash \Sigma \mid \Gamma$ un séquent dyadique positif, $\mathcal{D}$ une preuve de ce séquent et $\mathcal{D}^+$ le tronc positif (exponentiel) associé. Le **Graphe de Focalisation Exponentiel** *étend la définition des graphes de focalisation standards comme suit :*

- *Les sommets du graphe sont les formules actives de $\vdash \Sigma \mid \Gamma$ dans $\mathcal{D}^+$;*
- *Les arcs sont donnés par les séquents de la frontière de la manière habituelle (en prenant en compte l'assignation de biais s'il y en a une).*

Remarque 9.46

On notera que si $\vdash \Sigma' \mid \Gamma'$ est un séquent de la frontière, seules les formules de Γ' peuvent être sous-occurrence d'une formule active et donc elle seules contribuent aux arêtes du graphe.

Par ailleurs, on n'a pas besoin de prendre en compte les prémisses des promotions puisque ces séquents contiennent exactement une sous-formule d'une formule active de la racine : A est la seule formule dans la partie linéaire de ce séquent de la frontière.

La suite nous permet d'étendre le résultat de focalisation au cas exponentiel :

Proposition 9.47

Les graphes de focalisation exponentiels sont acycliques.

Démonstration : L'acyclicité des graphes de focalisation exponentiels est obtenue de la même manière que dans les cas précédents. Les deux seuls éléments nouveaux sont la déréliction et la promotion ; il nous faut donc simplement vérifier que ces deux règles ne créent pas de cycle :

- La promotion ne crée pas de problème puisque la prémisse d'une promotion est une feuille du tronc positif exponentiel mais qui ne contient qu'une formule active du séquent racine et n'est responsable d'aucune arête du graphe.
- Le fait d'avoir attribué un indice différent à chaque déréliction, et donc un sommet différent à chaque formule produite par une déréliction, assure qu'on ne peut pas avoir de cycle causé par une déréliction. En effet, considérant un tronc positif $\mathcal{D}^+$, on peut faire permuter toutes les dérélictions vers le bas et obtenir un tronc positif de la forme :

$$\mathcal{D'}^+ = \quad \dfrac{\dfrac{\mathcal{E}^+}{\vdash A_1, \dots A_k \mid A_1^{i_1}, \dots A_k^{i_k}, \Gamma}}{\vdash A_1, \dots A_k \mid \Gamma}\ \vdots\ der$$

tel que $\mathcal{E}^+$ est un tronc positif sans déréliction, le graphe de focalisation de $\mathcal{E}^+$ ayant un cycle si, et seulement si, celui de $\mathcal{D}^+$ contient un cycle.

□

L'acyclicité des graphes de focalisation exponentiels nous permet d'obtenir la propriété de focalisation pour les preuves de LL. Nous donnons une version du résultat sans prise en compte des biais atomiques mais l'extension à la focalisation sur les atomes ne poserait pas de problème :

Définition 9.48 (*Preuve focalisée de LL*)

Soit $\mathcal{D}$ une preuve d'un séquent $\mathcal{S}$. $\mathcal{D}$ est **focalisée** *si, étant donné un séquent $\mathcal{S}'$ de $\mathcal{D}$:*

- *si $\mathcal{S}'$ est un séquent négatif, alors il est conclusion d'une règle négative ;*
- *si $\mathcal{S}'$ est un séquent positif, prémisse d'une règle positive r et conclusion d'une règle r', et que F est la sous-occurrence de la formule principale de r (ou la formule de coupure dans le cas où $r = cut$) et n'est pas un littéral, on est dans l'un des cas suivants :*
  - *si $r \in \{[\oplus_i], [\exists], der, [\otimes], cut\}$ et si F est de la forme $A \oplus B$, $!A$, $\exists x.A$, $\exists X.A$ alors F est la formule principale de r' ;*
  - *si $r \in \{[\oplus_i], [\exists], der, [\otimes], cut\}$ et si $F = \mathbf{1}$, alors $\mathcal{S}'$ est conclusion d'un certain nombre d'affaiblissements suivis d'une règle $[\mathbf{1}]$ de formule principale F ;*
  - *si $r \in \{[\oplus_i], [\exists], der, [\otimes], cut\}$ et si F est de la forme $A \otimes B$, alors $\mathcal{S}'$ est conclusion d'un certain nombre de contractions au-dessus desquelles est appliquée une règle $[\otimes]$ de formule principale F ;*
  - *si $r = [!]$, alors on n'a pas de contrainte sur la règle r' ni sur sa formule principale.*

Remarque 9.49

Pour prendre en compte les biais atomiques, il suffirait d'ajouter la condition suivante à la définition précédente :

- *si $r \in \{[\oplus_i], [\exists], der, [\otimes], cut\}$ et* **si F est un littéral positif, alors $\mathcal{S}'$ est conclusion d'un certain nombre d'affaiblissements suivis d'une règle** *ini ;*

Théorème 9.50 (*Focalisation pour LL*)

Si $\mathcal{D}$ est une preuve d'un séquent $\mathcal{S}$ dans LL, alors on peut transformer $\mathcal{D}$ en une preuve focalisée $\mathcal{D}'$ par une série de permutations d'inférences par la méthode des graphes de focalisation.

Démonstration : La preuve ne pose pas de problème particulier. Il faut dans un premier temps passer de la preuve $\mathcal{D}$ à une preuve du système dyadique ce qui n'est pas problématique : les contractions peuvent être anticipées (permutées vers le bas de la preuve) et les affaiblissements retardés jusqu'aux axiomes ou aux règles $[\mathbf{1}]$.

Ensuite, on utilise un processus de focalisation similaire à celui qui a été utilisé plus tôt. La règle de contraction ne pose pas de problème puisqu'on part d'une preuve déjà construite $\mathcal{D}$ et que les permutations ne vont pas dupliquer les contractions au sein d'un même tronc positif. On peut ainsi raisonner sur le nombre d'alternances entre des phases positives et des phases négatives dans la preuve en transformation et, dans le cas d'une racine positive conclusion d'un tronc positif $\mathcal{D}^+$, on prend en compte le nombre de formules actives du tronc.

□

Commentaires sur la promotion. La bipolarité du connecteur «?» est manifestée par le fait que le système dyadique possède deux règles pour «?», la règle de classification dans la partie à gestion classique [?] et la déréliction *der*.

Le caractère bipolaire du «!» est moins évident à mettre en évidence. Le fait qu'une phase de focalisation soit intérrompue dès qu'une promotion est effectuée (c'est-à-dire qu'on ne peut pas s'assurer d'utiliser la sous-occurrence comme prochain foyer, il faudra peut-être plutôt faire une déréliction) témoigne de cette bipolarité. Nous donnons une autre justification de cette bipolarité en étudiant la forme de la règle promotion. Cette approche tendrait à renforcer l'idée selon laquelle la bipolarité du «!» ne serait pas intrinsèque au «!», mais qu'elle aurait plutôt sa source dans une interaction du «!» avec son contexte.

On peut considérer la variante suivante des règles exponentielles du calcul dyadique :

$$\frac{\vdash \Sigma\,?\,A \mid \Delta}{\vdash \Sigma \mid\ ?\,A, \Delta}\,[?]' \qquad \frac{\vdash \Sigma\,?\,A \mid A, \Delta}{\vdash \Sigma\,?\,A \mid \Delta}\,der' \qquad \frac{\vdash \cdot \mid \Sigma, A}{\vdash \Sigma \mid\ !\,A}\,[!]'$$

Comme nous l'avons remarqué plus haut, la règle $[?]'$ (comme la règle [?]) a pour rôle opérationnel de noter qu'une formule de la portion linéaire du séquent a une modalité «?». La règle der' joue le rôle de la déréliction couplée avec la contraction tandis que la règle $[!]'$ permet de retirer le connecteur ! «au prix» de l'information accumulée grâce aux règles $[?]'$. Le système dyadique ainsi obtenu est légèrement plus souple que le système dyadique précédent (il contient plus de preuves), mais surtout, les règles $[?]'$ et $[!]'$ rappellent la construction des boîtes exponentielles des réseaux de démonstration : la règle $[?]'$ correspond à la construction des ports auxiliaires d'une boîte promotion tandis que la règle $[!]'$ correspond au port principal de cette boîte. En outre, pour prouver $\vdash\ ?\,\Gamma, !\,A$, on doit construire deux boîtes exponentielles imbriquées.

La règle $[!]'$ permet de mettre en évidence que, si sa conclusion est un séquent positif, sa prémisse ne l'est qu'à la condition que A soit une formule positive et que Σ soit vide. Dans le cas où nous utilisons une telle règle $[!]'$, nous pouvons donner une définition des preuves focalisées qui ne nécessite pas de cas particulier pour abandonner la phase focalisée après une règle promotion puisque c'est le fait d'obtenir un séquent négatif (parce que $\Sigma \neq \emptyset$) qui contraint à abandonner cette phase négative.

Remarque 9.51

On notera que notre approche de la focalisation pour les exponentiels met en évidence que la règle I_2 du système Σ_3 d'Andreoli n'est pas nécessaire à la complétude du système focalisé.

Remarque 9.52

Notre traitement des exponentiels nous permet d'étendre le système focalisé pour traiter la règle MIX : En effet, on sait que $\vdash \Gamma$ est prouvable dans $LL + MIX$ si, et seulement si, $\vdash \Gamma, ?(\bot \otimes \bot)$ est prouvable dans LL, les occurrences de la règle MIX correspondant aux diverses instances de la règle $[\otimes]$ dont la formule principale est une formule $\bot \otimes \bot$ obtenue à partir de $?(\bot \otimes \bot)$ par une contraction suivie d'une déréliction ce qui, dans le système Σ_3, donne de la forme :

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdash \Phi, ?(\bot\otimes\bot) \colon \Gamma \Uparrow \cdot}{\vdash \Phi, ?(\bot\otimes\bot) \colon \Gamma \Uparrow \bot}\,[\bot]}{\vdash \Phi, ?(\bot\otimes\bot) \colon \Gamma \Downarrow \bot}\,R\Downarrow \quad \dfrac{\dfrac{\vdash \Phi, ?(\bot\otimes\bot) \colon \Delta \Uparrow \cdot}{\vdash \Phi, ?(\bot\otimes\bot) \colon \Delta \Uparrow \bot}\,[\bot]}{\vdash \Phi, ?(\bot\otimes\bot) \colon \Delta \Downarrow \bot}\,R\Downarrow}{\vdash \Phi, ?(\bot\otimes\bot) \colon \Gamma, \Delta \Downarrow \bot\otimes\bot}\,\otimes}{\vdash \Phi, ?(\bot\otimes\bot) \colon \Gamma, \Delta \Uparrow \cdot}\,D_2$$

Ce qui nous suggère la règle suivante, dans le système avec annotation :

$$\frac{\vdash \Phi \colon \Gamma \Uparrow \cdot \quad \vdash \Phi \colon \Delta \Uparrow \cdot}{\vdash \Phi \colon \Gamma, \Delta \Uparrow \cdot}\,[MIX]$$

Dans le système focalisé, la règle MIX ne peut donc être appliquée qu'à la fin d'une phase négative, juste avant de débuter une phase focalisée.

$$\frac{\vdash \Gamma, A}{\vdash ?\,\Gamma, !\,A}\ !\,fonct \qquad \frac{\vdash \Gamma}{\vdash \Gamma, ?\,A}\ ?\,w \qquad \frac{\vdash \Gamma, ?\,A, ?\,A}{\vdash \Gamma, ?\,A}\ ?\,c$$

FIG. 9.12 – Règles exponentielles de ELL

9.7. La focalisation au-delà de LL

La preuve que nous avons présentée est modulaire dans le sens où elle s'appuie sur une série de résultats simples qui peuvent être adaptés à des cadres plus riches. C'est ainsi que Baelde et Miller [BM07] sont parvenus à étendre la focalisation à une extension de la logique linéaire avec des points fixes (inductifs et coinductifs) en utilisant notre méthode.

Nous montrons pour notre part dans la section suivante comment notre preuve s'adapte très simplement pour obtenir un résultat de focalisation pour ELL.

9.7.1 Elementary Linear Logic

Nous allons montrer comment la méthode de ce chapitre permet d'obtenir presque immédiatement une propriété de focalisation pour une version de la logique linéaire (dont l'élimination des coupures est) à complexité bornée, ELL.

On distingue principalement trois systèmes de logiques allégées : ELL (Elementary linear logic), LLL (Light linear logic), SLL (Soft linear logic). Nous traitons ici le cas de ELL qui est le cas le plus simple des trois.

Présentation des systèmes allégés. Ces systèmes tels que ELL sont obtenus en modifiant les règles du fragment exponentiel de la logique linéaire, le reste de la logique n'étant pas forcément modifié[4].

L'idée de base est que, outre les règles habituelles des exponentiels :

$$\frac{\vdash ?\,\Gamma, A}{\vdash ?\,\Gamma, !\,A}\ ! \qquad \frac{\vdash \Gamma}{\vdash \Gamma, ?\,A}\ ?\,w \qquad \frac{\vdash \Gamma, ?\,A, ?\,A}{\vdash \Gamma, ?\,A}\ ?\,c \qquad \frac{\vdash \Gamma, A}{\vdash \Gamma, ?\,A}\ ?\,d$$

il existe une autre présentation des règles d'inférence du fragment exponentiel où on remplace la règle de promotion par une règle de «promotion fonctorielle» et une règle appelée «digging» :

$$\frac{\vdash \Gamma, A}{\vdash ?\,\Gamma, !\,A}\ !\,fonct \qquad \frac{\vdash \Gamma, ?\,?\,A}{\vdash \Gamma, ?\,A}\ !\,dig$$

Les logiques allégées peuvent souvent être présentées comme le résultat de restrictions sur les règles exponentielles dans cette dernière présentation avec promotion fonctorielle et digging ; c'est par exemple le cas de ELL.

ELL : Logique linéaire élémentaire. Les règles des connecteurs exponentiels de ELL sont données en figure 9.12. On obtient ce système en remplaçant la promotion par la promotion fonctorielle (mais en n'ajoutant pas le digging) et en enlevant la déréliction.

Le calcul des séquents complet de ELL est donné en figure 9.13. Dans ce système, on ne peut dériver ni la règle de déréliction ni la règle de digging.

[4]Dans la littérature on considère souvent des variantes affines, sans additifs, intuitionnistes, *etc.*

$$\frac{}{\vdash A, A^\perp}\ ini \qquad \frac{\vdash \Gamma, A \quad \vdash \Delta, A^\perp}{\vdash \Gamma, \Delta}\ cut$$

$$\frac{\vdash \Gamma}{\vdash \Gamma, \bot}\ [\bot] \quad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}\ [\parr] \qquad \frac{\vdash \Gamma, A \quad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B}\ [\otimes] \quad \frac{}{\vdash \mathbf{1}}\ [\mathbf{1}]$$

$$\frac{}{\vdash \Gamma, \top}\ [\top] \quad \frac{\vdash \Gamma, A \quad \vdash \Gamma, B}{\vdash \Gamma, A \,\&\, B}\ [\&] \qquad \frac{\vdash \Gamma, A_1}{\vdash \Gamma, A_1 \oplus A_2}\ [\oplus_0] \quad \frac{\vdash \Gamma, A_2}{\vdash \Gamma, A_1 \oplus A_2}\ [\oplus_1]$$

$$\frac{\vdash \Gamma, A[c/x]}{\vdash \Gamma, \forall x.A}\ [\forall] \quad \text{si } c \notin FV(\Gamma, \forall x.A). \qquad \frac{\vdash \Gamma, A[t/x]}{\vdash \Gamma, \exists x.A}\ [\exists]$$

$$\frac{\vdash \Gamma, B}{\vdash ?\,\Gamma, !\,B}\ [!] \qquad \frac{\vdash \Gamma}{\vdash \Gamma, ?\,B}\ [?\,w] \qquad \frac{\vdash \Gamma, ?\,B, ?\,B}{\vdash \Gamma, ?\,B}\ [?\,c]$$

Fig. 9.13 – Calcul des séquents pour ELL

$$\frac{}{\vdash \Sigma \mid A, A^\perp}\ ini \qquad \frac{\vdash \Sigma \mid \Gamma, A \quad \vdash \Sigma \mid \Delta, A^\perp}{\vdash \Sigma \mid \Gamma, \Delta}\ cut$$

$$\frac{\vdash \Sigma \mid \Gamma}{\vdash \Sigma \mid \Gamma, \bot}\ [\bot] \quad \frac{\vdash \Sigma \mid \Gamma, A, B}{\vdash \Sigma \mid \Gamma, A \parr B}\ [\parr] \qquad \frac{\vdash \Sigma \mid \Gamma, A \quad \vdash \Sigma \mid \Delta, B}{\vdash \Sigma \mid \Gamma, \Delta, A \otimes B}\ [\otimes] \quad \frac{}{\vdash \Sigma \mid \mathbf{1}}\ [\mathbf{1}]$$

$$\frac{}{\vdash \Sigma \mid \Gamma, \top}\ [\top] \quad \frac{\vdash \Sigma \mid \Gamma, A \quad \vdash \Sigma \mid \Gamma, B}{\vdash \Sigma \mid \Gamma, A \,\&\, B}\ [\&] \qquad \frac{\vdash \Sigma \mid \Gamma, A_1}{\vdash \Sigma \mid \Gamma, A_1 \oplus A_2}\ [\oplus_0] \quad \frac{\vdash \Sigma \mid \Gamma, A_2}{\vdash \Sigma \mid \Gamma, A_1 \oplus A_2}\ [\oplus_1]$$

$$\frac{\vdash \Sigma \mid \Gamma, A[c/x]}{\vdash \Sigma \mid \Gamma, \forall x.A}\ [\forall] \quad \text{si } c \notin FV(\Sigma, \Gamma, \forall x.A). \qquad \frac{\vdash \Sigma \mid \Gamma, A[t/x]}{\vdash \Sigma \mid \Gamma, \exists x.A}\ [\exists]$$

$$\frac{\vdash \Sigma, ?\,B \mid \Gamma}{\vdash \Sigma \mid\ ?\,B, \Gamma}\ [?] \qquad \frac{\vdash \emptyset \mid F_1^{i_1}, \ldots, F_k^{i_k}, B}{\vdash ?\,F_1, \ldots, ?\,F_k \mid\ !\,B}\ [!] \quad \text{avec } i_1, \ldots, i_k \geq 0.$$

Fig. 9.14 – Calcul des séquents dyadiques pour ELL

Le principal résultat concernant ELL est que l'élimination des coupures des réseaux de preuve de EAL (la version de ELL avec affaiblissement) a une complexité élémentaire (bornée par une tour d'exponentiels), ce qui a pour conséquence que les fonctions représentables dans EAL sont exactement les fonctions élémentairement récursives.

Focalisation dans ELL. Comme d'habitude pour traiter les exponentiels de manière focalisée, on passe par un système dyadique de manière à traiter l'affaiblissement de manière implicite en le retardant le plus possible et à incorporer la contraction dans les autres règles branchantes (le tenseur et la coupure). Cette étape permet de décomposer les règles du «?» en une partie positive et une partie négative.

Le système dyadique obtenu est donné en figure 9.14.

On constate que, dans le système dyadique, on a les propriétés de permutabilité suivantes : $[\bot], [\parr], [\&], [\top], [\forall], [?]$ sont négatives (permutabilité pleine) et $[\mathbf{1}], [\otimes], [\oplus_i], [\exists], [!]$ ont la propriété de permutabilité faible et sont donc considérées comme positives. Séquents positifs et négatifs sont définis comme auparavant.

La réversibilité des négatifs ne pose pas de problème, il nous faut vérifier que l'on peut définir une discipline de focalisation sur les positifs.

On définit donc troncs positifs et graphes de focalisation comme suit :

Définition 9.53 (*ELL-Troncs positifs*)

Soit $\mathcal{S} = \vdash \Sigma \mid \Gamma$ un séquent positif et soit $\mathcal{D}$ une preuve de ce séquent. Le tronc positif $\mathcal{D}^+$ associé à $\mathcal{D}$ est le sous-arbre maximal de $\mathcal{D}$ enraciné en $\mathcal{S}$ tel que :
- *$\mathcal{D}^+$ ne contient que des règles positifs ;*
- *si un séquent apparaissant dans $\mathcal{D}^+$ est conclusion d'une occurrence de la règle de promotion fonctorielle alors il s'agit d'une feuille de $\mathcal{D}^+$.*

Définition 9.54 (*ELL-Graphe de focalisation*)

Le graphe de focalisation d'une preuve dans ELL d'un séquent positif $\vdash \Sigma \mid \Gamma$ est construit comme suit :
- *Les sommets du graphe sont les formules de Γ actives dans $\mathcal{D}^+$;*
- *Il y a une arête de A vers B s'il existe un séquent, $\mathcal{S}$, de la frontière de $\mathcal{D}^+$ contenant une sous-occurrence négative de A et une sous-occurrence positive de B.*

Proposition 9.55

Les ELL-graphes de focalisation sont acycliques.

Démonstration : La proposition se démontre facilement en constatant que tout ELL-tronc positif est également un tronc positif dans le sens de LL et que tout ELL-graphe de focalisation est également un graphe de focalisation dan le sens de LL et donc en particulier est acyclique.

□

On peut donc combiner le tout pour obtenir un résultat de focalisation pour ELL.

$$\frac{\vdash \Psi : \Delta \Uparrow L}{\vdash \Psi : \Delta \Uparrow \bot, L}\ [\bot] \qquad \frac{\vdash \Psi : \Delta \Uparrow F, G, L}{\vdash \Psi : \Delta \Uparrow F \parr G, L}\ [\parr] \qquad \boxed{\frac{\vdash \Psi, F : \Delta \Uparrow L}{\vdash \Psi : \Delta \Uparrow ?F, L}\ [?]}$$

$$\frac{}{\vdash \Psi : \Delta \Uparrow \top, L}\ [\top] \qquad \frac{\vdash \Psi : \Delta \Uparrow F, L \quad \vdash \Psi : \Delta \Uparrow G, L}{\vdash \Psi : \Delta \Uparrow F \mathbin{\&} G, L}\ [\&] \qquad \frac{\vdash \Psi : \Delta \Uparrow B[y/x], L}{\vdash \Psi : \Delta \Uparrow \forall x.B, L}\ [\forall]$$

$$\frac{}{\vdash \Psi : \cdot \Downarrow \mathbf{1}}\ [\mathbf{1}] \qquad \frac{\vdash \Psi : \Delta_1 \Downarrow F \quad \vdash \Psi : \Delta_2 \Downarrow G}{\vdash \Psi : \Delta_1, \Delta_2 \Downarrow F \otimes G}\ [\otimes]$$

$$\frac{\vdash \Psi : \Delta \Downarrow F_1}{\vdash \Psi : \Delta \Downarrow F_1 \oplus F_2}\ [\oplus_0] \qquad \frac{\vdash \Psi : \Delta \Downarrow F_2}{\vdash \Psi : \Delta \Downarrow F_1 \oplus F_2}\ [\oplus_1] \qquad \frac{\vdash \Psi : \Delta \Downarrow B[t/x]}{\vdash \Psi : \Delta \Downarrow \exists x.B}\ [\exists]$$

$$\frac{\vdash \Psi : \Delta \Uparrow F}{\vdash \Psi : \Delta \Downarrow F}\ R\Downarrow \qquad \boxed{\frac{\vdash \cdot : \cdot \Uparrow \Psi', F}{\vdash \Psi : \cdot \Downarrow !F}\ [!] \quad \begin{array}{ll} \text{où} & \Psi = F_1, \ldots, F_k \\ \text{et} & \Psi' = F_1^{i_1}, \ldots, F_k^{i_k} \\ \text{avec} & i_1, \ldots, i_k \geq 0 \end{array}}$$

$$\frac{\vdash \Psi : \Delta, F \Uparrow L}{\vdash \Psi : \Delta \Uparrow F, L}\ R\Uparrow \qquad \frac{}{\vdash \Psi : X^\perp \Downarrow X}\ I \qquad \frac{\vdash \Psi : \Delta \Downarrow F}{\vdash \Psi : \Delta, F \Uparrow \cdot}\ D$$

FIG. 9.15 – Système focalisé pour ELL. *Les contraintes sur les règles sont les suivantes : dans la règle $\forall$, la variable y n'est pas libre dans la conclusion. Dans $R \Uparrow$, F n'est pas négative tandis que dans $R \Downarrow$, F est soit négative soit un littéral négatif. Dans I_1, K est un littéral positif. Dans D_1, F n'est pas un littéral négatif.*

Chapitre 10

Multifocalisation

Résumé:

Nous étudions dans ce chapitre une extension des systèmes de preuves focalisés, la multifocalisation. Nous commençons par présenter le système multifocalisé qui trouve son origine dans les graphes de focalisation du précédent chapitre. Nous montrons ensuite qu'il est possible de définir une notion de maximalité parmi les preuves multifocalisée ce qui fournit une propriété de canonicité des preuves. Nous illustrons ensuite cette propriété de canonicité en présentant la relation qu'entretiennent les preuves maximalement multifocalisées et les réseaux de démonstration.

Références : Certains résultats de ce chapitre ont été publiés dans l'article *Canonical Sequent Proofs via Multi-Focusing* [CMS08] ainsi que dans la dernière section de *From proofs to focused proofs : a modular proof of focalization in Linear Logic* [MS07]

Sommaire

10.1. Origines de la Multifocalisation

L'une des objections à l'approche focalisée (ou polarisée) de la logique linéaire, objection que suggère notamment Abramsky [Abr03] (ou encore Girard [Gir06] au chapitre 12 du *Point aveugle*), est que celle-ci renforce la séquentialisation des preuves du calcul des séquents là où il faudrait au contraire chercher à «dé-séquentialiser» les preuves, à la manière des réseaux de démonstration ou des jeux concurrents.

Une question d'associativité. Le problème mentionné par Abramsky [Abr03] trouve sa source dans les interprétations de jeux de la logique où apparaissent des problèmes d'associativité lors de la composition des stratégies [AJ94][1], du fait de la séquentialisation des règles du calcul des séquents. Selon la manière dont on compose des stratégies, on peut obtenir des résultats différents car chaque composition induit une «première règle» différente dans le résultat. Cet élément peut être directement observé dans le calcul des séquents de MALL.

Considérons les trois preuves suivantes :

$$\mathcal{D}_2 = \dfrac{\dfrac{\dfrac{}{\vdash a^\perp, a}\,ini \quad \dfrac{}{\vdash a^\perp, a}\,ini}{\vdash a^\perp \,\&\, a^\perp, a}\,[\&] \quad \dfrac{\dfrac{}{\vdash a^\perp, a}\,ini \quad \dfrac{}{\vdash a^\perp, a}\,ini}{\vdash a^\perp \,\&\, a^\perp, a}\,[\&]}{\vdash a^\perp \,\&\, a^\perp, a \,\&\, a}\,[\&]$$

$$\mathcal{D}_1 = \dfrac{\dfrac{\dfrac{}{\vdash a^\perp, a}\,ini}{\vdash a^\perp, a \oplus a}\,[\oplus_1]}{\vdash a^\perp \oplus a^\perp, a \oplus a}\,[\oplus_1] \qquad \mathcal{D}_3 = \dfrac{\dfrac{\dfrac{}{\vdash a^\perp, a}\,ini}{\vdash a^\perp \oplus a^\perp, a}\,[\oplus_1]}{\vdash a^\perp \oplus a^\perp, a \oplus a}\,[\oplus_1]$$

On peut les composer à l'aide de la règle de coupure de deux manières :

$$\mathcal{E}_1 = \dfrac{\dfrac{\mathcal{D}_1}{\vdash a^\perp \oplus a^\perp, a \oplus a} \quad \dfrac{\dfrac{\mathcal{D}_2}{\vdash a^\perp \,\&\, a^\perp, a \,\&\, a} \quad \dfrac{\mathcal{D}_3}{\vdash a^\perp \oplus a^\perp, a \oplus a}}{\vdash a^\perp \,\&\, a^\perp, a \oplus a}\,cut}{\vdash a^\perp \oplus a^\perp, a \oplus a}\,cut$$

ou

$$\mathcal{E}_3 = \dfrac{\dfrac{\dfrac{\mathcal{D}_1}{\vdash a^\perp \oplus a^\perp, a \oplus a} \quad \dfrac{\mathcal{D}_2}{\vdash a^\perp \,\&\, a^\perp, a \,\&\, a}}{\vdash a^\perp \oplus a^\perp, a \,\&\, a}\,cut \quad \dfrac{\mathcal{D}_3}{\vdash a^\perp \oplus a^\perp, a \oplus a}}{\vdash a^\perp \oplus a^\perp, a \oplus a}\,cut$$

La preuve $\mathcal{E}_1$ se réduit, par élimination des coupures, en $\mathcal{D}_1$ tandis que la réduction des coupures de $\mathcal{E}_3$ nous conduit à la preuve $\mathcal{D}_3$. Ce qui témoigne de la non-associativité de la coupure : $\langle\langle \mathcal{D}_1, \mathcal{D}_2\rangle, \mathcal{D}_3\rangle \neq \langle \mathcal{D}_1, \langle \mathcal{D}_2, \mathcal{D}_3\rangle\rangle$.

Ce problème peut être résolu de deux manières comme l'explique Abramsky [Abr03] :

- soit en contraignant la syntaxe de manière à forcer la composition à s'effectuer dans un sens et réaliser ainsi l'associativité, ce qui correspond à la focalisation[2] ;
- soit, au contraire, en relâchant la séquentialité de manière à ce que l'égalité soit possible, ce qui correspond aux réseaux de démonstration (ou aux jeux concurrents).

[1] Abramsky mentionne que cet élément avait été indépendamment noté par Blass.

[2] Ainsi, le théorème d'associativité de la Ludique (voir le théorème 11.30) repose-t-il sur cet élément.

Relâcher la séquentialité de la focalisation. L'un des points sur lesquels nous souhaitons nous pencher dans le présent chapitre est la question de la séquentialité de la focalisation : peut-on concevoir une focalisation qui soit moins séquentielle ?

Cette question trouve également sa source dans le fait que même si la focalisation quotiente un certain nombre de preuves, elle est loin d'effectuer le quotient des réseaux de preuves : un séquent positif peut avoir des preuves focalisées distinctes qui ont pourtant toutes deux le même réseau de démonstration sous-jacent comme le montrent les deux preuves suivantes[3] :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash a, a^{\perp}}\;ini}{\vdash a, a^{\perp}, \bot}\;[\bot]}{\vdash a, a^{\perp} \parr \bot}\;[\parr]}{\vdash a, \mathbf{0} \oplus (a^{\perp} \parr \bot)}\;[\oplus_1]}{\vdash a, \bot, \mathbf{0} \oplus (a^{\perp} \parr \bot)}\;[\bot]}{\vdash a \parr \bot, \mathbf{0} \oplus (a^{\perp} \parr \bot)}\;[\parr]}{\vdash (a \parr \bot) \oplus \mathbf{0}, \mathbf{0} \oplus (a^{\perp} \parr \bot)}\;[\oplus_0] \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash a, a^{\perp}}\;ini}{\vdash a, \bot, a^{\perp}}\;[\bot]}{\vdash a \parr \bot, a^{\perp}}\;[\parr]}{\vdash (a \parr \bot) \oplus \mathbf{0}, a^{\perp}}\;[\oplus_0]}{\vdash (a \parr \bot) \oplus \mathbf{0}, a^{\perp}, \bot}\;[\bot]}{\vdash (a \parr \bot) \oplus \mathbf{0}, a^{\perp} \parr \bot}\;[\parr]}{\vdash (a \parr \bot) \oplus \mathbf{0}, \mathbf{0} \oplus (a^{\perp} \parr \bot)}\;[\oplus_1]$$

Les deux preuves précédentes ont des foyers différents et sont pourtant équivalentes par permutation et égalisées par les réseaux de démonstration.

Si la véritable signification du résultat de focalisation de la logique linéaire est la mise en évidence des polarités et la possibilité de définir des connecteurs synthétiques, on peut se demander s'il n'est pas possible de définir des variantes de la focalisation qui conservent ces propriétés tout en étant moins contraignantes du point de vue de la séquentialité.

Multifocalisation : retour aux sources. La multifocalisation que nous allons étudier dans ce chapitre trouve quant à elle son origine dans les graphes de focalisation qui nous ont servi au chapitre précédent. L'argument principal nous permettant d'établir la propriété de focalisation résidait dans l'acyclicité des graphes de focalisation : acyclique, le graphe possédait une source, et toute source du graphe pouvait servir de foyer pour la focalisation puisqu'on sait que la permutation vers le bas des règles positives appliquées à cette formule ne rencontrera que d'autres formules positives ; ce qui n'est jamais bloquant du fait de la permutabilité faible des positifs.

Cet argument est suffisant pour obtenir la focalisation, mais il dit en fait bien plus : on peut non seulement décomposer le niveau positif de n'importe quelle source du graphe, mais on peut en outre décomposer les niveaux positifs de toutes les sources en même temps. Ceci nous amène à considérer la règle suivante :

$$\boxed{\dfrac{\vdash F_1, \ldots F_k \colon \Delta \Downarrow F_1^{i_1}, \ldots F_k^{i_k}, F'_1, \ldots F'_l}{\vdash F_1, \ldots F_k \colon \Delta, F'_1, \ldots F'_l \Uparrow \cdot}\;[MF] \text{ où } i_1 \ldots i_k \geq 0}$$

Avec cette nouvelle règle de ***multifocalisation***, quand on décidera de focaliser, on pourra focaliser sur plus d'une formule positive à la fois, c'est-à-dire que nos séquents annotés au cours d'une phase (multi)focalisée auront maintenant la forme $\vdash \Delta \Downarrow \Gamma$ (avec Γ non-vide a priori). Lorsque l'on est dans une phase de focalisation, les formules focalisées

[3]On remarquera que la question de la focalisation sur les atomes est tout à fait étrangère à cet exemple alors que les preuves $\mathcal{D}_1$ et $\mathcal{D}_3$ considérées plus haut ne fourniraient pas toutes les deux des preuves focalisées si l'on considère une assignation de biais à la manière d'Andreoli.

$$\frac{\vdash \Delta \Uparrow L}{\vdash \Delta \Uparrow \bot, L}\,[\bot] \qquad \frac{\vdash \Delta \Uparrow F, G, L}{\vdash \Delta \Uparrow F \parr G, L}\,[\parr] \qquad \frac{}{\vdash \Delta \Uparrow \top, L}\,[\top]$$

$$\frac{\vdash \Delta \Uparrow F, L \quad \vdash \Delta \Uparrow G, L}{\vdash \Delta \Uparrow F \,\&\, G, L}\,[\&] \qquad \frac{\vdash \Delta \Uparrow F, L}{\vdash \Delta \Uparrow \forall X.F, L}\,[\forall]$$

$$\frac{}{\vdash \cdot \Downarrow \mathbf{1}}\,[\mathbf{1}] \qquad \frac{\vdash \Delta_1 \Downarrow F, \Gamma_1 \quad \vdash \Delta_2 \Downarrow G, \Gamma_2}{\vdash \Delta_1, \Delta_2 \Downarrow F \otimes G, \Gamma_1, \Gamma_2}\,[\otimes]$$

$$\frac{\vdash \Delta \Downarrow F_1, \Gamma}{\vdash \Delta \Downarrow F_1 \oplus F_2, \Gamma}\,[\oplus_0] \qquad \frac{\vdash \Delta \Downarrow F_2, \Gamma}{\vdash \Delta \Downarrow F_1 \oplus F_2, \Gamma}\,[\oplus_1] \qquad \frac{\vdash \Delta \Downarrow F[G/X], \Gamma}{\vdash \Delta \Downarrow \exists X.F, \Gamma}\,[\exists]$$

$$\frac{\vdash \Delta, F \Uparrow L}{\vdash \Delta \Uparrow F, L}\,[R \Uparrow] \qquad \frac{\vdash \Delta \Uparrow \Gamma}{\vdash \Delta \Downarrow \Gamma}\,[R \Downarrow] \qquad \frac{}{\vdash X^{\bot}, X \Uparrow \cdot}\,ini \qquad \frac{\vdash \Delta \Downarrow \Gamma}{\vdash \Delta, \Gamma \Uparrow \cdot}\,[MF]$$

FIG. 10.1 – Système multifocalisé pour MALL.

sont décomposées en n'utilisant que des inférences positives jusqu'à ce que seules des formules négatives (ou atomiques) restent dans le foyer. C'est alors seulement que le foyer est relâché et que les formules négatives peuvent être à nouveau décomposées dans la phase négative.

Ainsi, dans le calcul multifocalisé, nous avons deux différences par rapport à la focalisation habituelle :

– La focalisation peut sélectionner un ***ensemble de formules positives***, un ensemble de foyers et non pas seulement un singleton ;
– La phase de focalisation ***gèle les formules négatives*** qui apparaîtraient durant la phase focalisée jusqu'à ce que tous les foyers soient négatifs ou atomiques. Dans le système usuel de la focalisation, ce dernier élément n'apparaît pas puisque une seule formule est focalisée pendant toute la phase positive et qu'en conséquence la phase focalisée s'achève dès que le foyer devient négatif.

Nous nous bornerons dans ce chapitre à étudier le cas de MALL mais l'extension à tout LL ne poserait pas de problème en considérant la version générale de la règle introduite ci-dessus.

10.2. Multifocalisation pour MALL

Le système multifocalisé est construit autour de la règle $[MF]$ en conservant le reste du système focalisé :

Définition 10.1 (*Calcul des séquents MALL multifocalisé, Σ_{MF}*)

Le **système multifocalisé** Σ_{MF} *est présenté en figure 10.1. Les conditions sur les règles d'inférence sont les suivantes :*

– *dans* $[R \Uparrow]$, F *est positive ou atomique ;*
– *dans* $[R \Downarrow]$, Γ *n'est constitué que de formules négatives ou atomiques ;*
– *dans* $[MF]$, Γ *n'est pas vide ;*
– *Dans* $[\forall]$, X *n'est pas libre dans la conclusion de la règle.*

Remarque 10.2

Le système Σ_{MF} correspond au cas sans polarisation des atomes, mais nous pourrions également considérer un système avec une règle initiale focalisée qui, dans le système multifocalisé, pourrait s'exprimer comme suit :

$$\dfrac{}{\vdash \cdot \Downarrow X^{\perp}, X}\ ini$$

Définition 10.3 ($\lfloor \mathcal{D} \rfloor$)

Si $\mathcal{D} \vdash \Delta \Uparrow \Gamma$ ou si $\mathcal{D} \vdash \Delta \Downarrow \Gamma$, on écrit $\lfloor \mathcal{D} \rfloor$ pour la preuve de $\vdash \Delta, \Gamma$ qui remplace tout séquent de la forme $\vdash \Delta' \Uparrow \Gamma'$ ou $\vdash \Delta' \Downarrow \Gamma'$ dans $\mathcal{D}$ par le séquent $\vdash \Delta', \Gamma'$ en effaçant toutes les instances de $[R \Uparrow], [R \Downarrow]$ et $[MF]$ et en laissant telles quelles toutes les autres règles.

Une preuve de la forme $\lfloor \mathcal{D} \rfloor$ sera également appelée **preuve multifocalisée**.

On vérifie immédiatement que l'arbre obtenu par la procédure décrite dans la définition précédente est bien une preuve de MALL.

Remarque 10.4

On pourrait donner une définition directe des preuves de MALL qui sont multifocalisées à la manière des définitions 9.22 et 9.30. La condition serait alors de la forme :

- *si $\mathcal{S}$ est un séquent positif dans une preuve $\mathcal{D}$, soit $\mathcal{D}^{+}_{\mathcal{S}}$ le tronc positif de la preuve enracinée en $\mathcal{S}$, alors toutes les sous-occurrences des formules actives dans $\mathcal{D}^{+}$ apparaissant dans une feuille de $\mathcal{D}^{+}_{\mathcal{S}}$ sont négatives ou atomiques.*

Théorème 10.5 (*Correction et complétude de la multifocalisation*)

- **Correction :** *Si $\mathcal{D} \vdash \Delta \Uparrow \Gamma$ ou si $\mathcal{D} \vdash \Delta \Downarrow \Gamma$, alors $\lfloor \mathcal{D} \rfloor \vdash \Delta, \Gamma$.*
- **Complétude :** *Si $\vdash \Gamma$, alors $\vdash \cdot \Uparrow \Gamma$.*

Démonstration :

- La correction est immédiate.
- La complétude vient de l'observation que le calcul focalisé pour MALL (avec ou sans focalisation sur les atomes selon la version considérée) est contenu dans Σ_{MF} en restreignant le contexte Γ dans les règles $[MF]$ à un singleton et en utilisant le théorème de complétude du système simplement focalisé du chapitre précédent.

$\square$

Définition 10.6 (*Racines*)

Supposons que $\mathcal{D} = \dfrac{\vdash \Delta \Downarrow \Gamma}{\vdash \Gamma, \Delta \Uparrow \cdot}\ [MF]$.

On appelle alors les formules de Γ les **racines** *de $\mathcal{D}$ et on le notera $Root(\mathcal{D})$.*

On utilisera également cette définition pour les preuves multifocalisées présentées dans le système MALL usuel, sans annotation de phase $\Uparrow$ ou $\Downarrow$.

Remarque 10.7

Pour la définition de $Root(\mathcal{D})$ lorsque $\mathcal{D}$ est une preuve dans le système MALL usuel (c'est-à-dire sans les annotations $\Uparrow$ ou $\Downarrow$), il est utile de préciser certains éléments :

- *toutes les formules actives du tronc positif seront dans $Root(\mathcal{D})$;*
- *pour déterminer l'appartenance des atomes à $Root(\mathcal{D})$, on prendra la convention adaptée en fonction du système considéré :*
  - *si on ne considère pas les polarités des atomes, on inclura tous les atomes dans la racine,*
  - *si on considère une polarisation comme dans la remarque 10.2, on inclura tous les atomes négatifs dans $Root(\mathcal{D})$ ainsi que les atomes positifs qui interviennent dans une règle initiale dont la conclusion est une feuille du tronc positif de $\mathcal{D}$.*

10.2.1 Relations entre preuves multifocalisées

La correction et la complétude du système multifocalisé sont des propriétés évidentes et ne sont pas ce qui fait l'intérêt du système multifocalisé.

Le système multifocalisé trouve son intérêt dans l'étude des relations entre preuves multifocalisées : nous voulons dans la suite de ce chapitre étudier les relations entre les preuves multifocalisées qui sont équivalentes par permutation de règles d'inférence. En particulier, nous nous demandons si l'ensemble des preuves multifocalisées équivalentes à une preuve donnée $\mathcal{D} \vdash \Gamma$ est clos par union des racines, c'est-à-dire s'il existe une preuve qui soit maximale du point de vue de la taille des ensembles de foyers $Root(\mathcal{D})$.

L'intérêt d'une telle notion de maximalité vient de ce qu'elle traduit un plus grand parallélisme dans la preuve multifocalisée puisque l'ordre des règles positives entre différents foyers n'importe pas.

Dans un article en collaboration avec Chaudhuri et Miller [CMS08], nous avons étudié la multifocalisation pour les preuves sans coupure de MALL propositionnel et avons en particulier étudié les preuves maximalement multifocalisées sur lesquelles nous allons nous pencher dans la suite de ce chapitre.

Nous nous limiterons dans ce chapitre au calcul des séquents MALL sans coupure et sans unité mais avec quantification du second ordre, que l'on notera $MALL^{-}$. L'extension de l'analyse de la multifocalisation aux exponentiels et à la coupure ne devrait pas poser de problème en reprenant les techniques des graphes de focalisation du chapitre précédent. Pour le traitement des unités, nous renvoyons à [CMS08] et à la fin de la section 10.4 où nous en discutons certains éléments.

Relation entre les résultats de ce chapitre et les résultats de [CMS08]. Les seuls résultats de [CMS08] que nous présenterons ici tels qu'ils sont abordés dans l'article sont ceux concernant l'analyse de la relation entre preuves maximalement multifocalisées et réseaux de démonstrations MLL car nous avons développé ces résultats de manière autonome, le reste des résultats de l'article étant véritablement des résultats établis en collaboration avec Chaudhuri et Miller (en particulier, le calcul $[PMF]$ de [CMS08] est dû à Chaudhuri). L'une des principales différences entre l'approche adoptée dans [CMS08] et celle que nous présentons ici est que dans [CMS08] nous travaillions directement dans le système multifocalisé avec annotations de phase tandis que nous allons considérer ici les preuves de LL qui pourraient être annotées (voir définition 10.1 et remarque 10.4).

$$\frac{\vdash \Psi : \Delta \Uparrow L}{\vdash \Psi : \Delta \Uparrow \bot, L}\,[\bot] \qquad \frac{\vdash \Psi : \Delta \Uparrow F, G, L}{\vdash \Psi : \Delta \Uparrow F \parr G, L}\,[\parr]$$

$$\frac{}{\vdash \Psi : \Delta \Uparrow \top, L}\,[\top] \qquad \frac{\vdash \Psi : \Delta \Uparrow F, L \quad \vdash \Psi : \Delta \Uparrow G, L}{\vdash \Psi : \Delta \Uparrow F \,\&\, G, L}\,[\&]$$

$$\frac{\vdash \Psi : \Delta \Uparrow F, L}{\vdash \Psi : \Delta \Uparrow \forall X.F, L}\,[\forall] \qquad \boxed{\frac{\vdash \Psi, F : \Delta \Uparrow L}{\vdash \Psi : \Delta \Uparrow ?F, L}\,[?]}$$

$$\frac{}{\vdash \Psi : \cdot \Downarrow \mathbf{1}}\,[\mathbf{1}] \qquad \frac{\vdash \Psi : \Delta_1 \Downarrow F, \Gamma_1 \quad \vdash \Psi : \Delta_2 \Downarrow G, \Gamma_2}{\vdash \Psi : \Delta_1, \Delta_2 \Downarrow F \otimes G, \Gamma_1, \Gamma_2}\,[\otimes]$$

$$\frac{\vdash \Psi : \Delta \Downarrow F_1, \Gamma}{\vdash \Psi : \Delta \Downarrow F_1 \oplus F_2, \Gamma}\,[\oplus_0] \qquad \frac{\vdash \Psi : \Delta \Downarrow F_2, \Gamma}{\vdash \Psi : \Delta \Downarrow F_1 \oplus F_2, \Gamma}\,[\oplus_1]$$

$$\frac{\vdash \Psi : \Delta \Downarrow F[G/X], \Gamma}{\vdash \Psi : \Delta \Downarrow \exists X.F, \Gamma}\,[\exists] \qquad \boxed{\frac{\vdash \Psi : \cdot \Uparrow F}{\vdash \Psi : \cdot \Downarrow\, !F}\,[!]}$$

$$\frac{}{\vdash \Psi : X^\perp, X \Uparrow \cdot}\,\mathit{ini} \qquad \boxed{\frac{\vdash \Psi : \Delta_1 \Downarrow F, \Gamma_1 \quad \vdash \Psi : \Delta_2 \Downarrow F^\perp, \Gamma_2}{\vdash \Psi : \Delta_1, \Delta_2 \Downarrow \Gamma_1, \Gamma_2}\,\mathit{cut}}$$

$$\frac{\vdash \Psi : \Delta, F \Uparrow L}{\vdash \Psi : \Delta \Uparrow F, L}\,[R \Uparrow] \quad \frac{\vdash \Psi : \Delta \Uparrow \Gamma}{\vdash \Psi : \Delta \Downarrow \Gamma}\,[R \Downarrow] \quad \boxed{\frac{\vdash F_1, \ldots, F_k : \Delta \Downarrow \Gamma, F_1^{i_1}, \ldots, F_k^{i_k}}{\vdash F_1, \ldots, F_k : \Delta, \Gamma \Uparrow \cdot}\,[MF]}$$

FIG. 10.2 – Système multifocalisé pour LL avec coupure.

Nous verrons brièvement à la fin de la section 10.4 en quoi la technique d'anti-réversibilité développée ici trouve ses limites avec les unités et pourquoi on doit s'appuyer sur d'autres techniques comme celles présentées dans [CMS08].

10.2.2 Système Multifocalisé pour LL avec Coupure

Dans un souci de complétude, nous présentons un calcul des séquents multifocalisés, Σ_{MF}^{LL}, pour toute la logique linéaire avant de passer à l'étude plus spécifique de la multifocalisation dans MALL.

On a les conditions suivantes sur les règles d'inférences :
- dans $[R \Uparrow]$, F est positive ou atomique ;
- dans $[R \Downarrow]$, Γ n'est constitué que de formules négatives ou atomiques ;
- dans $[MF]$, $i_1, \ldots, i_k \geq 0$ et $F_1^{i_1}, \ldots, F_k^{i_k}, \Gamma$ n'est pas vide ;
- Dans $[\forall]$, X n'est pas libre dans la conclusion de la règle.

10.3. Équivalences de preuves par permutation

$$\dfrac{\dfrac{\dfrac{\mathcal{C}}{\vdash A, C, \Gamma} \quad \dfrac{\mathcal{D}}{\vdash B, D, \Delta}}{\vdash A, B, C \otimes D, \Gamma, \Delta}[\otimes]}{\vdash A \parr B, C \otimes D, \Gamma, \Delta}[\parr] \qquad \dfrac{\dfrac{\dfrac{\mathcal{C}}{\vdash A, C[F/X], \Gamma}}{\vdash A, \exists X.C, \Gamma}[\exists] \quad \dfrac{\dfrac{\mathcal{C}'}{\vdash B, C[G/X], \Gamma}}{\vdash B, \exists X.C, \Gamma}[\exists]}{\vdash A \,\&\, B, \exists X.C, \Gamma}[\&] \qquad \dfrac{\dfrac{\dfrac{\mathcal{D}}{\vdash A, B[X/Y], \Gamma}}{\vdash A, \exists Y.B, \Gamma}[\exists]}{\vdash \forall X.A, \exists Y.B, \Gamma}[\forall]$$

$$\dfrac{\dfrac{\dfrac{\mathcal{C}}{\vdash A, C, \Gamma}}{\vdash A, C \oplus D, \Gamma}[\oplus_0] \quad \dfrac{\dfrac{\mathcal{D}}{\vdash B, D, \Gamma}}{\vdash B, C \oplus D, \Gamma}[\oplus_1]}{\vdash A \,\&\, B, C \oplus D, \Gamma}[\&] \qquad \dfrac{\dfrac{\dfrac{\mathcal{C}_1}{\vdash A, C, \Gamma} \quad \dfrac{\mathcal{D}_1}{\vdash D, \Delta}}{\vdash A, C \otimes D, \Gamma, \Delta}[\otimes] \quad \dfrac{\dfrac{\mathcal{C}_2}{\vdash C, \Gamma} \quad \dfrac{\mathcal{D}_2}{\vdash B, D, \Delta}}{\vdash B, C \otimes D, \Gamma, \Delta}[\otimes]}{\vdash A \,\&\, B, C \otimes D, \Gamma, \Delta}[\&]$$

FIG. 10.3 – Différents cas de non-permutation pos/neg.

Pour mener à bien notre étude de la multifocalisation, il nous faut commencer par préciser la relation de permutation d'inférences que nous considérons pour définir une équivalence par permutation entre les preuves de $MALL^-$.

Nous considérons deux équivalences : la permutation usuelle d'inférences de MALL et la permutation dite ***isopolaire*** qui est une restriction de la première équivalence. L'équivalence par permutation d'inférences est tout à fait standard :

Définition 10.8 (*Équivalence par permutation dans $MALL^-$*)

L'équivalence $\sim$ est définie comme la plus petite relation d'équivalence contenant les permutations

- **pos/pos :** $\boxed{\frac{[\otimes]}{[\otimes]}\updownarrow}$, $\boxed{\frac{[\otimes]}{[\oplus_i]}\updownarrow}$, $\boxed{\frac{[\otimes]}{[\exists]}\updownarrow}$, $\boxed{\frac{[\oplus_i]}{[\oplus_j]}\updownarrow}$, $\boxed{\frac{[\oplus_i]}{[\exists]}\updownarrow}$, $\boxed{\frac{[\exists]}{[\exists]}\updownarrow}$, $\boxed{\frac{[\otimes]}{[\oplus_i]}\updownarrow}$;
- **neg/neg :** $\boxed{\frac{[\parr]}{[\parr]}\updownarrow}$, $\boxed{\frac{[\parr]}{[\&]}\updownarrow}$, $\boxed{\frac{[\parr]}{[\forall]}\updownarrow}$, $\boxed{\frac{[\&]}{[\&]}\updownarrow}$, $\boxed{\frac{[\&]}{[\forall]}\updownarrow}$, $\boxed{\frac{[\forall]}{[\forall]}\updownarrow}$;
- **neg/pos :** $\boxed{\frac{[\parr]}{[\otimes]}\downarrow}$, $\boxed{\frac{[\parr]}{[\oplus_i]}\downarrow}$, $\boxed{\frac{[\parr]}{[\exists]}\downarrow}$, $\boxed{\frac{[\&]}{[\otimes]}\downarrow}$, $\boxed{\frac{[\&]}{[\oplus_i]}\downarrow}$, $\boxed{\frac{[\&]}{[\exists]}\downarrow}$, $\boxed{\frac{[\forall]}{[\otimes]}\downarrow}$, $\boxed{\frac{[\forall]}{[\oplus_i]}\downarrow}$, $\boxed{\frac{[\forall]}{[\exists]}\downarrow}$.

qui ont été présentées au chapitre précédent (table 9.1 et figures 9.7, 9.8, 9.9 et 9.10).

Cas de non-permutation pos/neg. Comme on l'a vu lors de l'étude de la focalisation, les permutations pos/neg ne sont pas toujours possibles. En particulier, les cas suivants de règles adjacentes peuvent ne pas être permutables : $\boxed{\frac{[\otimes]}{[\parr]}\downarrow}$, $\boxed{\frac{[\otimes]}{[\&]}\downarrow}$, $\boxed{\frac{[\oplus_i]}{[\&]}\downarrow}$, $\boxed{\frac{[\exists]}{[\&]}\downarrow}$, $\boxed{\frac{[\exists]}{[\forall]}\downarrow}$ comme illustré en figure 10.3.

La permutation des règles $[\otimes]/[\&]$ pose un autre problème. Lorsqu'il s'agit de faire permuter la règle $[\&]$ au-dessous de la règle $[\otimes]$, on duplique l'une des preuves prémisse de $[\otimes]$:

$$\dfrac{\dfrac{\dfrac{\mathcal{C}_1}{\vdash A, C, \Gamma} \quad \dfrac{\mathcal{C}_2}{\vdash B, C, \Gamma}}{\vdash A \,\&\, B, C, \Gamma, \Delta}[\&] \quad \dfrac{\mathcal{D}}{\vdash D, \Delta}}{\vdash A \,\&\, B, C \otimes D, \Gamma, \Delta}[\otimes] \quad \longrightarrow \quad \dfrac{\dfrac{\dfrac{\mathcal{C}_1}{\vdash A, C, \Gamma} \quad \dfrac{\mathcal{D}}{\vdash D, \Delta}}{\vdash A, C \otimes D, \Gamma, \Delta}[\otimes] \quad \dfrac{\dfrac{\mathcal{C}_2}{\vdash B, C, \Gamma} \quad \dfrac{\mathcal{D}}{\vdash D, \Delta}}{\vdash B, C \otimes D, \Gamma, \Delta}[\otimes]}{\vdash A \,\&\, B, C \otimes D, \Gamma, \Delta}[\&]$$

En revanche, la permutation dans l'autre sens requiert non seulement que chaque sous-formule de $A \& B$ soit envoyée dans la même prémisse du $[\otimes]$ mais en outre que les deux occurences de l'autre prémisse du $[\otimes]$ soient conclusions de deux preuves égales (en inversant la flèche de la réduction ci-dessus).

Dans la suite de ce chapitre, nous allons en fait considérer une permutation plus facile à manipuler qui autorise la permutation non seulement si les deux sous-preuves $\mathcal{D}$ et $\mathcal{D}'$ sont égales, mais également si elles sont égales à permutation d'inférences près, auquel cas on retient toujours, par convention, la sous-preuve apparaissant dans la prémisse de gauche du $[\&]$:

$$\dfrac{\dfrac{\dfrac{\mathcal{C}_1}{\vdash A, C, \Gamma} \quad \dfrac{\mathcal{D}}{\vdash D, \Delta}}{\vdash A, C \otimes D, \Gamma, \Delta}[\otimes] \quad \dfrac{\dfrac{\mathcal{C}_2}{\vdash B, C, \Gamma} \quad \dfrac{\mathcal{D}'}{\vdash D, \Delta}}{\vdash B, C \otimes D, \Gamma, \Delta}[\otimes]}{\vdash A \& B, C \otimes D, \Gamma, \Delta}[\&] \quad \longrightarrow \quad \dfrac{\dfrac{\dfrac{\mathcal{C}_1}{\vdash A, C, \Gamma} \quad \dfrac{\mathcal{C}_2}{\vdash B, C, \Gamma}}{\vdash A \& B, C, \Gamma, \Delta}[\&] \quad \dfrac{\mathcal{D}}{\vdash D, \Delta}}{\vdash A \& B, C \otimes D, \Gamma, \Delta}[\otimes]$$

si $\mathcal{D} \sim \mathcal{D}'$.

Permutation iso-polaire. Il nous sera enfin utile de considérer une sous-équivalence de l'équivalence par permutation qui est l'équivalence d'isopolarité. On restreint les permutations aux permutations de type neg/neg et pos/pos :

Définition 10.9 (*Permutation isopolaire*)

> *La permutation* **isopolaire***, notée $\approx$, est la restriction de $\sim$ aux permutations entre inférences d'une même polarité, soit neg/neg, soit pos/pos.*

Cette relation d'isopolarité est plus contraignante que l'équivalence par permutation :
- elle n'utilise que des permutations réversibles, c'est-à-dire qu'on ne fait jamais de permutation qui risquerait d'être bloquante ;
- dans le système Σ_{MF}, deux preuves isopolaires sélectionnent systématiquement les mêmes ensembles de foyers lors de l'application d'une règle $[MF]$;
- la permutation isopolaire correspond à la séquentialité du calcul des séquents qui est purement due à la structure arborescente des séquents et non pas à la séquentialité qui peut avoir un sens logique comme une séquentialisation $\otimes$ / $\parr$;
- le calcul de l'équivalence $\approx$ est plus simple que celui de $\sim$ puisqu'il est localisé au sein des phases de polarité.

Nous avons maintenant les outils pour étudier la question de la maximalité des preuves multifocalisées.

10.4. Une notion de maximalité pour les preuves multifocalisées

10.4.1 Preuves maximalement multifocalisées

Nous montrons dans cette section que l'on peut définir une notion de maximalité parmi les preuves multifocalisées :

Définition 10.10 (*Preuve maximalement multifocalisée*)

Une preuve multifocalisée $\mathcal{D}$ *d'un séquent* $\vdash \Gamma$ *(resp.* $\vdash \Delta \Uparrow \Gamma$ *ou* $\vdash \Delta \Downarrow \Gamma$*) est* **maximale** *si toute sous-preuve* $\mathcal{E}$ *de* $\vdash \Gamma'$ *(resp.* $\vdash \Delta' \Uparrow \Gamma'$, $\vdash \Delta' \Downarrow \Gamma'$*) est telle que s'il existe une preuve multifocalisée* $\mathcal{D}'$ *de* $\vdash \Gamma$ *(resp.* $\vdash \Delta \Uparrow \Gamma$, $\vdash \Delta \Downarrow \Gamma$*) contenant une sous-preuve* $\mathcal{E}'$ *de* $\vdash \Gamma'$ *(resp.* $\vdash \Delta' \Uparrow \Gamma'$, $\vdash \Delta' \Downarrow \Gamma'$*), on a :*

$$\mathcal{E} \sim \mathcal{E}' \quad \Rightarrow \quad Root(\mathcal{E}') \subseteq Root(\mathcal{E}).$$

Une preuve est donc maximale si on ne peut pas trouver de preuve multifocalisée $\sim$équivalente qui ait une racine plus grande.

On est intéressé par les deux propriétés suivantes concernant les preuves multifocalisées :

- **Existence d'une preuve maximale :** Si $\mathcal{D}$ est une preuve multifocalisée, il existe une preuve maximalement multifocalisée $\mathcal{E} \sim \mathcal{D}$;
- **Canonicité des preuves maximales :** Soient $\mathcal{D}$ et $\mathcal{E}$ sont deux preuves maximalement multifocalisées. Si $\mathcal{E} \sim \mathcal{D}$ alors $\mathcal{E} \approx \mathcal{D}$.

Remarque 10.11

La canonicité est une conséquence simple de l'existence d'une preuve maximale puisque deux preuves maximales doivent avoir les mêmes racines.

10.4.2 Lemmes préliminaires

Le résultat suivant indique que l'équivalence par permutation $\sim$ ne fait jamais disparaître une inférence d'une preuve.

Lemme 10.12

Soient $\mathcal{D}_1$ *et* $\mathcal{D}_2$ *deux preuves multifocalisées de* $MALL^-$ *d'un séquent* $\vdash \Gamma$ *telles que* $\mathcal{D}_1 \sim \mathcal{D}_2$.

Si $F \in \Gamma$ *et* G *une sous-occurrence de* F*, s'il y a une inférence* R *dont* G *est formule principale dans* $\mathcal{D}_1$*, alors il y a une inférence* R' *dont* G *est formule principale dans* $\mathcal{D}_2$.

<u>**Démonstration :**</u> On vérifie simplement par une induction sur la taille des séquents que les permutations d'inférence de $MALL^-$ peuvent fusionner ou dupliquer des inférences, modifier leurs contextes, mais ne peuvent jamais effacer purement et simplement une inférence.

□

Lemme 10.13

Soient $\mathcal{D}_1$ et $\mathcal{D}_2$ deux preuves multifocalisées de $MALL^-$ d'un séquent $\vdash \Gamma$ telles que $\mathcal{D}_1 \sim \mathcal{D}_2$. Si R_1 et R_2 sont des inférences respectives de $\mathcal{D}_1$ et $\mathcal{D}_2$ qui se correspondent par $\sim$ de formule principale $A \otimes B$ de formes respectives :

$$\dfrac{\vdash F, \Delta_1 \quad \vdash G, \Delta_2}{\vdash F \otimes G, \Delta_1, \Delta_2}\, R_1 \qquad\qquad \dfrac{\vdash F, \Gamma_1 \quad \vdash G, \Gamma_2}{\vdash F \otimes G, \Gamma_1, \Gamma_2}\, R_2$$

On a alors, si $A \in \Gamma_1, \Gamma_2$ (resp. Δ_1, Δ_2) et si on note

$$\mathcal{A} = \{B \in \Delta_1, \Delta_2, t.q.\ B \text{ est une sous-occurrence de } A\}$$

(resp. $\mathcal{A} = \{B \in \Gamma_1, \Gamma_2, t.q.\ B \text{ est une sous-occurrence de } A\}$) alors :

$$\boxed{A \in \Gamma_i \text{ (resp. } A \in \Delta_i) \Rightarrow \mathcal{A} \subset \Delta_i \text{ (resp. } \mathcal{A} \subset \Gamma_i)}$$

Démonstration : On vérifie simplement que toutes les permutations d'inférences de $MALL^-$ impliquant une règle $\otimes$ préservent cette condition (le cas de la permutation $\otimes$ /[$\otimes$] nécessiterait une induction sur la taille du séquent pour assurer qu'on ne perd pas la propriété en ne conservant qu'une des deux preuves équivalentes parmi les prémisses des deux règles [$\otimes$]) :

$$\dfrac{\dfrac{\vdash A, C, D, \Gamma}{\boxed{\vdash A, C \parr D, \Gamma}}[\parr] \quad \boxed{\vdash B, \Delta}}{\boxed{\vdash A \otimes B, C \parr D, \Gamma, \Delta}}[\otimes] \quad\longrightarrow\quad \dfrac{\dfrac{\boxed{\vdash A, C, D, \Gamma} \quad \boxed{\vdash B, \Delta}}{\boxed{\vdash A \otimes B, C, D, \Gamma, \Delta}}[\otimes]}{\vdash A \otimes B, C \parr D, \Gamma, \Delta}[\parr]$$

$$\dfrac{\dfrac{\vdash A, C, \Gamma \quad \vdash A, D, \Gamma}{\boxed{\vdash A, C \,\&\, D, \Gamma}}[\&] \quad \boxed{\vdash B, \Delta}}{\boxed{\vdash A \otimes B, C \,\&\, D, \Gamma, \Delta}}[\otimes]$$

$$\longrightarrow\quad \dfrac{\dfrac{\boxed{\vdash A, C, \Gamma} \quad \boxed{\vdash B, \Delta}}{\boxed{\vdash A \otimes B, C, \Gamma, \Delta}}[\otimes] \quad \dfrac{\boxed{\vdash A, D, \Gamma} \quad \boxed{\vdash B, \Delta}}{\boxed{\vdash A \otimes B, D, \Gamma, \Delta}}[\otimes]}{\vdash A \otimes B, C \,\&\, D, \Gamma, \Delta}[\&]$$

$$\dfrac{\dfrac{\boxed{\vdash A, C, \Gamma} \quad \boxed{\vdash D, \Delta}}{\boxed{\vdash A, C \otimes D, \Gamma, \Delta}}[\otimes]}{\vdash A \oplus B, C \otimes D, \Gamma, \Delta}[\oplus_0] \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A, C, \Gamma}{\boxed{\vdash A \oplus B, C, \Gamma}}[\oplus_0] \quad \boxed{\vdash D, \Delta}}{\boxed{\vdash A \oplus B, C \otimes D, \Gamma, \Delta}}[\otimes]$$

$$\dfrac{\dfrac{\vdash A, C, \Gamma \quad \vdash D, \Delta}{\boxed{\vdash A, C \otimes D, \Gamma, \Delta}}[\otimes] \quad \boxed{\vdash B, \Sigma}}{\boxed{\vdash A \otimes B, C \otimes D, \Gamma, \Delta, \Sigma}}[\otimes] \quad\longleftrightarrow\quad \dfrac{\dfrac{\boxed{\vdash A, C, \Gamma} \quad \boxed{\vdash B, \Sigma}}{\boxed{\vdash A \otimes B, C, \Gamma, \Sigma}}[\otimes] \quad \vdash D, \Delta}{\vdash A \otimes B, C \otimes D, \Gamma, \Delta, \Sigma}[\otimes]$$

Les deux quantificateurs donnent lieu à la même forme de permutation vis-à-vis du $\otimes$, on ne montre que le cas de [$\forall$] :

$$\dfrac{\dfrac{\boxed{\vdash A, B, \Gamma} \quad \boxed{\vdash C, \Delta}}{\boxed{\vdash A, B \otimes C, \Gamma, \Delta}}[\otimes]}{\vdash \forall X.A, B \otimes C, \Gamma, \Delta}[\forall] \quad\longleftrightarrow\quad \dfrac{\dfrac{\vdash A, B, \Gamma}{\boxed{\vdash \forall X.A, B, \Gamma}}[\forall] \quad \boxed{\vdash C, \Delta}}{\boxed{\vdash \forall X.A, B \otimes C, \Gamma, \Delta}}[\otimes]$$

$\square$

10.4.3 Anti-réversibilité

Pour obtenir l'existence de preuves maximales, on va considérer une stratégie particulière de permutation d'inférence qui est l'inverse de l'opération de renversement des négatifs considérée au cours du processus de focalisation : on va tâcher de faire remonter les inférences négatives aussi haut que possible dans la preuve.

Définition 10.14

Un preuve multifocalisée $\mathcal{D}$ de $\vdash \Gamma$ est **anti-renversée** *si :*

- *$\mathcal{D}$ ne contient aucune inférence positive R^+ immédiatement au-dessus d'une inférence négative R^- dans une configuration telle qu'on puisse les faire permuter et*
- *aucune permutation pos/pos ou neg/neg ne permet de créer une telle configuration.*

Définition 10.15

Soit $\mathcal{D}$ une preuve multifocalisée de $\vdash \Gamma$. Une preuve $\mathcal{E}$ de $\vdash \Gamma$ est un **anti-renversement** *de $\mathcal{D}$ si :*

- **anti-renversement :** *$\mathcal{D} \sim \mathcal{E}$ et que de plus on peut transformer $\mathcal{D}$ en $\mathcal{E}$ par une séquence de permutations d'inférences $(\rho_i)_{i\in I}$ en n'effectuant que des permutations entre positifs (de type pos/pos) ou entre négatifs (de type neg/neg) ou bien en faisant permuter une règle d'inférence négative* **au-dessus** *d'une règle positive (permutation de type pos/neg) ;*
- **maximalité :** *$\mathcal{E}$ est une preuve* **anti-renversée**.

Le fait de pouvoir passer de $\mathcal{D}$ à $\mathcal{E}$ par une séquence de permutations telle que $(\rho_i)_{i\in I}$ est noté $\mathcal{D} \longrightarrow_{\text{anti-rév}} \mathcal{E}$.

10.4.4 Maximalité

Théorème 10.16

Soient $\mathcal{D}_1$ et $\mathcal{D}_2$ deux preuves multifocalisées d'un séquent $\vdash \Gamma$ telles que $\mathcal{D}_1 \sim \mathcal{D}_2$.

Il existe une preuve multifocalisée $\mathcal{D}_3$ telle que $\mathcal{D}_3 \sim \mathcal{D}_1 \sim \mathcal{D}_2$ et

$$Root(\mathcal{D}_3) = Root(\mathcal{D}_1) \cup Root(\mathcal{D}_2).$$

Démonstration : Soient $\mathcal{D}$ et $\mathcal{E}$ deux preuves multifocalisées de $\vdash \Gamma$ telles que $\mathcal{D} \sim \mathcal{E}$.

Nous montrons qu'il existe $\mathcal{F} \sim \mathcal{D}$ telle que les sous-occurrences de formules de $Root(\mathcal{D}) \cup Root(\mathcal{E})$ apparaissant dans la frontière du tronc positif de $\mathcal{F}$, $\mathcal{F}^+$, sont négatives ou atomiques. Cela suffit alors, par la méthode des graphes de focalisation adaptée pour sélectionner toutes les sources du graphe, à trouver une preuve multifocalisée $\mathcal{F}'$ telle que $Root(\mathcal{F}') \supseteq Root(\mathcal{D}) \cup Root(\mathcal{E})$.

Il suffit en fait de montrer que si $F \in Root(\mathcal{E}) \setminus Root(\mathcal{D})$, il existe $\mathcal{F} \sim \mathcal{D}$ telle que les sous-occurrences de formules de $Root(\mathcal{D}) \cup \{F\}$ apparaissant dans la frontière du tronc positif de $\mathcal{F}$, $\mathcal{F}^+$, sont négatives ou atomiques, un simple argument inductif permet ensuite de conclure.

Supposons donc qu'il existe une formule $F \in Root(\mathcal{E}) \setminus Root(\mathcal{D})$. On note $\mathcal{D}^+$ le tronc positif de $\mathcal{D}$ et $\mathcal{S}_i, 1 \leq i \leq k$ les séquents de la frontière de $\mathcal{D}^+$ qui sont séquents conclusions des preuves multifocalisées $\mathcal{D}_i, 1 \leq i \leq k$, que l'on pourra aussi désigner par $\mathcal{D}_{|\mathcal{S}_i}$. Soit $(\rho_i)_{1\leq i\leq n}$ une séquence de permutations d'inférences qui transforme

$\mathcal{D}$ en $\mathcal{E}$ et $(\theta_i)_{1\leq i\leq m}$ une séquence de permutations d'inférences qui transforme $\mathcal{E}$ en $\mathcal{D}$[4]. La formule F fait partie de Γ et n'est pas décomposée dans le tronc positif de $\mathcal{D}$: il existe donc un (unique) séquent de la frontière de $\mathcal{D}^+$ qui contient l'occurence de F considérée, nous supposerons dans la suite qu'il s'agit de $\mathcal{S}_1$. On applique une anti-réversibilité à $\mathcal{D}_1$: soit $\mathcal{D}_1'$, un anti-renversement de $\mathcal{D}_1$, c'est-à-dire une preuve de $\mathcal{S}_1$ telle qu'aucune règle d'inférence positive ne puisse permuter sous une règle d'inférence négative et telle que $\mathcal{D}_1 \longrightarrow_{\text{anti-rév}} \mathcal{D}_1'$. En remplaçant $\mathcal{D}_1$ par $\mathcal{D}_1'$ dans $\mathcal{D}$, on obtient une nouvelle preuve $\mathcal{D}'$ de $\vdash \Gamma$ (qui, a priori, n'est plus multifocalisée).

Nous montrons maintenant que $\mathcal{D}'$ est telle que les sous-occurrences de formules de $Root(\mathcal{D}) \cup \{F\}$ apparaissant dans la frontière du tronc positif de $\mathcal{D}'$, $\mathcal{D}'^+$, sont négatives ou atomiques. Pour cela, il nous suffit de montrer que le premier niveau de connecteurs positifs de F est décomposé dans le tronc positif de $\mathcal{D}_1'$ (ou plus précisément que toutes les sous-occurrences de F apparaissant dans un séquent de la frontière de ${\mathcal{D}_1'}^+$ sont négatives ou atomiques).

Supposons en effet que cela ne soit pas le cas. Cela signifie qu'il existe dans l'un des séquents de la frontière de $\mathcal{D}_1'$, noté $\mathcal{S}_i'$, une sous-occurrence positive de F, par exemple F'.

Par ailleurs, puisque $F \in Root(\mathcal{E})$, il y a dans le tronc positif de $\mathcal{E}$ un séquent contenant une occurrence de F' et une règle d'inférence appliquée à cette formule F' (grâce au lemme 10.12). Il y a donc forcément, dans $\mathcal{D}'$, une inférence R^+ appliquée à F' puisque $\mathcal{D}' \sim \mathcal{E}$ et que les permutations considérées ne font jamais disparaître de règle d'inférence (lemme 10.12 à nouveau). Il y a donc dans $\mathcal{D}'$ des occurrences de règles d'inférence appliquées à F' et celles-ci ne peuvent être que dans l'arbre de preuve enraciné en $\mathcal{S}_i'$ (il s'agit en effet du seul séquent de la frontière de $\mathcal{D}'$ qui contient une occurrence de F'). En outre, comme on a supposé que F' appartenait à la frontière de ${\mathcal{D}_1'}^+$, il y a nécessairement une inférence négative R^- sous l'inférence R^+ (sinon, F' serait décomposée dans le tronc positif).

Du fait que les règles positives permutent entre elles, on peut supposer que chaque occurrence d'une inférence sur F' est immédiatement au-dessus d'une règle négative (sinon, on fait permuter ces inférences vers le bas jusqu'à ce que ce soit le cas). On considère maintenant une règle négative R_- maximale parmi les règles négatives se trouvant sous une occurrence de R^+ (c'est-à-dire qu'aucune autre règle négative au-dessous d'une occurrence de R^+ n'est au-dessus de R_-).

Puisque l'anti-réversibilité ne fait que commuter vers le bas des positifs et remonter vers le haut des négatifs, la règle R_- est également au-dessous de R^+ dans $\mathcal{D}$. Par ailleurs, à la fin de la séquence de permutations $(\rho_i)_{1\leq i\leq n}$, la règle R^+ est au-dessous de R_- puisque $F \in Root(\mathcal{E})$. Il existe donc $\mathcal{D}^k \longrightarrow_{\rho_k} \mathcal{D}^{k+1}$ tel que ρ_k permute R^+ et R_-. De même, il existe un plus grand indice l tel que θ_l permute R^+ est au-dessus de R_- : $\mathcal{E}^l \longrightarrow_{\theta_l} \mathcal{E}^{l+1}$. Raisonnons maintenant par cas sur une telle règle R_- qui peut être parmi $[\parr]$, $[\&]$ ou $[\forall]$:

Cas $R_- = [\parr]$: On suppose que la formule à laquelle s'applique la règle R_- est de la forme $A \parr B$. Dans tous les cas sauf si $F' = C \otimes D$, on peut permuter R^+ avec R_-. Si $F' = C \otimes D$, le seul cas qui pourrait bloquer la permutation est un cas de la forme :

$$\dfrac{\vdash C, \Delta \quad \vdash D, \Delta'}{\vdash C \otimes D, \Delta, \Delta'}\ [\otimes]$$

avec $A \in \Delta$ et $B \in \Delta'$. Ce cas est impossible pour la raison suivante : dans la séquence menant de $\mathcal{D}$ à $\mathcal{E}$, on a, au niveau de ρ_k, une configuration de la forme

$$\dfrac{\dfrac{\vdash A, B, C, \Gamma \quad \vdash D, \Gamma'}{\vdash A, B, C \otimes D, \Gamma, \Gamma'}\ [\otimes]}{\vdash A \parr B, C \otimes D, \Gamma, \Gamma'}\ [\parr] \quad \longrightarrow_{\rho_k} \quad \dfrac{\dfrac{\vdash A, B, C, \Gamma}{\vdash A \parr B, C, \Gamma}\ [\parr] \quad \vdash D, \Gamma'}{\vdash A \parr B, C \otimes D, \Gamma, \Gamma'}\ [\otimes]$$

[4]Du fait de la permutation $\otimes$ / $\&$, $(\rho_i)_{1\leq i\leq n}$ et $(\theta_i)_{1\leq i\leq m}$ ne sont pas nécessairement inverse l'une de l'autre.

et le lemme 10.13 assure que $A, B \prec A \parr B$ sont tous les deux envoyés dans la même prémisse du $[\otimes]$ de $\mathcal{D}_1'^{+}$ permettant ainsi de permuter $[\parr]$ et $[\otimes]$ dans $\mathcal{D}_1'^{+}$.

Cas $R_- = [\&]$: Supposons que la formule à laquelle s'applique R_- soit $A \,\&\, B$. On sait qu'au dessus de chaque prémisse de la règle $[\&]$ il y a une règle appliquée à F', et qu'il s'agit de la même règle. On est donc dans une situation de la forme :

$$\dfrac{\dfrac{\mathcal{F}_1}{\vdash A, F', \Gamma}\,R^+ \qquad \dfrac{\mathcal{F}_2}{\vdash B, F', \Gamma}\,R^+}{\vdash A \,\&\, B, F', \Gamma}\,[\&]$$

Si R^+ est $[\oplus_i]$ ou $[\exists]$, la permutation ne pose pas de problème (en effet, si $R^+ = [\oplus_i]$ par exemple, on sait que si on avait à la fois $[\oplus_0]$ et $[\oplus_1]$, ces deux inférences auraient toutes deux une occurrence dans $\mathcal{E}$ ce qui est impossible puisque $F \in \mathit{Root}(E)$; le cas de $[\exists]$ se traite de la même manière).

Considérons le cas où $R^+ = [\otimes]$. D'après le lemme 10.13, on est dans une situation de la forme :

$$\dfrac{\dfrac{\dfrac{\mathcal{F}_1}{\vdash A, C, \Gamma} \quad \dfrac{\mathcal{F}_2}{\vdash D, \Gamma'}}{\vdash A, C \otimes D, \Gamma, \Gamma'}\,[\otimes] \qquad \dfrac{\dfrac{\mathcal{F}_1'}{\vdash B, C, \Gamma} \quad \dfrac{\mathcal{F}_2'}{\vdash D, \Gamma'}}{\vdash B, C \otimes D, \Gamma, \Gamma'}\,[\otimes]}{\vdash A \,\&\, B, C \otimes D, \Gamma, \Gamma'}\,[\&]$$

et la permutation est possible à condition que $\mathcal{F}_2 \sim \mathcal{F}_2'$ auquel cas on obtient :

$$\dfrac{\dfrac{\dfrac{\mathcal{F}_1}{\vdash A, C, \Gamma} \quad \dfrac{\mathcal{F}_1'}{\vdash B, C, \Gamma}}{\vdash A \,\&\, B, C, \Gamma}\,[\&] \qquad \dfrac{\mathcal{F}_2}{\vdash D, \Gamma'}}{\vdash A \,\&\, B, C \otimes D, \Gamma, \Gamma'}\,[\otimes]$$

Pour montrer que $\mathcal{F}_2 \sim \mathcal{F}_2'$, on considère la permutation de $\mathcal{E}^l$ à $\mathcal{D}'$: $\mathcal{E}^l \longrightarrow_{\theta_j, j \geq l} \mathcal{D} \longrightarrow^{\star}_{\text{anti-rév}} \mathcal{D}'$. En particulier, on a :

$$\mathcal{E}^l \ni \dfrac{\dfrac{\dfrac{\mathcal{G}_1}{\vdash A, C, \Sigma} \quad \dfrac{\mathcal{G}_1'}{\vdash B, C, \Sigma}}{\vdash A \,\&\, B, C, \Sigma}\,[\&] \qquad \dfrac{\mathcal{G}_2}{\vdash D, \Sigma'}}{\vdash A \,\&\, B, C \otimes D, \Sigma, \Sigma'}\,[\otimes]$$

$$\longrightarrow_{\theta_l} \dfrac{\dfrac{\dfrac{\mathcal{G}_1}{\vdash A, C, \Sigma} \quad \dfrac{\mathcal{G}_2}{\vdash D, \Sigma'}}{\vdash A, C \otimes D, \Sigma, \Sigma'}\,[\otimes] \qquad \dfrac{\dfrac{\mathcal{G}_1'}{\vdash B, C, \Sigma} \quad \dfrac{\mathcal{G}_2}{\vdash D, \Sigma'}}{\vdash B, C \otimes D, \Sigma, \Sigma'}\,[\otimes]}{\vdash A \,\&\, B, C \otimes D, \Sigma, \Sigma'}\,[\&] \in \mathcal{E}_{l+1}$$

On peut suivre l'évolution de la prémisse de droite des deux $[\otimes]$ de $\mathcal{E}^{l+1}$ le long de cette séquence de permutations, c'est à dire de dérivations $(\mathcal{H}_i \vdash D, \Sigma_i)_{0 \leq i \leq n}$ et $(\mathcal{H}_i' \vdash D, \Sigma_i')_{0 \leq i \leq n}$ avec $\mathcal{H}_0 = \mathcal{H}_0' = \mathcal{G}_2$, $\Sigma_0 = \Sigma_0' = \Sigma$, et à chaque permutation de $\mathcal{E}^{l+1}$ à $\mathcal{D}'$, on associe la prémisse de droite de chacun des $[\otimes]$ (en cas de permutation avec un $[\&]$, on suit la copie du $[\otimes]$ de la prémisse de gauche du $[\&]$) à $\mathcal{H}_i$ et $\mathcal{H}_i'$ et on note Σ_i (resp. Σ_i') le contexte de la formule D dans le séquent conclusion de $\mathcal{H}_i$ (resp. $\mathcal{H}_i'$).

On a alors : $\Sigma_i = \Sigma_j \Rightarrow \mathcal{H}_i \sim \mathcal{H}_j$ et par ailleurs, si on considère $\mathcal{H}_i \longrightarrow \mathcal{H}_{i+1}$, on a quatre possibilités :

- soit $\mathcal{H}_i = \mathcal{H}_{i+1}$ (la permutation a lieu dans une partie de la preuve qui n'implique pas $\mathcal{H}_i$) ;
- soit il y a une permutation de règles interne à $\mathcal{H}_i$;
- soit il y a une permutation de la dernière règle de $\mathcal{H}_i$ avec le $[\otimes]$ délimitant $\mathcal{H}_i$;

– soit il y a une permutation du $[\otimes]$ avec la règle qui est juste au-dessous.

Dans tous les cas, la prémisse de gauche de la règle $[\otimes]$ n'intervient pas dans la définition de $\mathcal{H}_{i+1}$, c'est-à-dire que les seules inférences de la preuve $\mathcal{E}^{l+1}$ qui pourront contribuer aux $\mathcal{H}_i$ sont les inférences qui se trouvaient dans $\mathcal{H}_0 = \mathcal{G}_2$ et celles qui se trouvaient sous le $[\&]$. Ces dernières, permutant avec $[\&]$ sont copiées à la fois dans la prémisse de gauche et celle de droite du $[\&]$. Ce dernier fait assure que l'on pourrait trouver une réduction à partir $\mathcal{E}^{l+1}$ dans laquelle $\mathcal{H}_i$ et $\mathcal{H}'_i$ sont échangées, ce qui nous amène à étendre l'énoncé précédent en : $\Sigma_i = \Sigma'_j \Rightarrow \mathcal{H}_i \sim \mathcal{H}'_j$ d'où l'on conclut que $\mathcal{F}_2 \sim \mathcal{F}'_2$ puisqu'elles ont mêmes conclusions et que $\mathcal{F}_2 = \mathcal{H}_n$ et $\mathcal{F}'_2 = \mathcal{H}'_n$ et que $\Sigma_n = \Gamma' = \Sigma'_n$.
On peut ainsi conclure que la permutation $[\&]/[\otimes]$ est possible.

Cas $R_- = [\forall]$: Supposons que la formule à laquelle s'applique R_- soit de la forme $\forall X.A$. Le seul cas problématique est celui où $R^+ = [\exists]$:

$$\dfrac{\dfrac{\vdash A, B[F/Y], \Gamma}{\vdash A, \exists Y.B, \Gamma}[\exists]}{\vdash \forall X.A, \exists Y.B, \Gamma}[\forall]$$

Dans ce cas, la permutation n'est possible que si $X \notin FV(F)$. On s'en assure en étudiant ρ_k. En effet, on est alors dans une configuration de la forme :

$$\dfrac{\dfrac{\vdash A, B[F/Y], \Gamma}{\vdash A, \exists Y.B, \Gamma}[\exists]}{\vdash \forall X.A, \exists Y.B, \Gamma}[\forall] \longrightarrow_{\rho_k} \dfrac{\dfrac{\vdash A, B[F/Y], \Gamma}{\vdash \forall X.A, B[F/Y], \Gamma}[\forall]\ \star}{\vdash \forall X.A, \exists Y.B, \Gamma}[\exists]$$

La condition $\star$ d'application de la règle $[\forall]$ assure que $X \notin FV(F)$ et qu'il est possible de permuter les deux règles des quantificateurs dans $\mathcal{D}'$.

Nous pouvons donc conclure de cette étude des règles d'inférence de $\mathcal{D}'$ que les seules sous-occurrences de F qui peuvent apparaître dans un séquent de la frontière de $\mathcal{D}_1'^{+}$ sont négative ou atomiques, ce qui conclut la preuve du théorème.

□

10.4.5 La question des unités

Alors que nous avons partiellement traité les unités dans [CMS08], nous les avons exclues du traitement de la maximalité dans le présent chapitre. Nous donnons ici quelques éléments de discussion à ce sujet.

Problèmes de permutation des unités négatives. Le problème vient avant tout des unités négatives, $[\top]$ et $[\bot]$.

– Le $[\top]$ pose problème car la permutation du $[\top]$ efface des portions de preuves :

$$\dfrac{\dfrac{}{\vdash \top, \Gamma, A}[\top] \quad \dfrac{\mathcal{D}}{\vdash \Delta, B}}{\vdash \top, \Gamma, \Delta, A \otimes B}[\otimes] \qquad \longrightarrow \qquad \dfrac{}{\vdash \top, \Gamma, \Delta, A \otimes B}[\top]$$

En particulier, si nous autorisons la permutation de la règle $[\top]$, on aurait

$$\dfrac{\dfrac{}{\vdash \top, \Gamma, A}[\top] \quad \dfrac{\mathcal{D}}{\vdash \Delta, B}}{\vdash \top, \Gamma, \Delta, A \otimes B}[\otimes] \qquad \sim \qquad \dfrac{\dfrac{}{\vdash \top, \Gamma, A}[\top] \quad \dfrac{\mathcal{E}}{\vdash \Delta, B}}{\vdash \top, \Gamma, \Delta, A \otimes B}[\otimes]$$

quelles que soient $\mathcal{D}$ et $\mathcal{E}$ des preuves de $\vdash \Delta, B$. Pour faire face à ce problème, nous n'avons considéré dans [CMS08] que les permutations de $[\top]$ dites ***iso-initiales***,

c'est-à-dire où la permutation de $[\top]$ avec une autre inférence ne crée pas ou n'efface d'instance de la règle initiale.
Sans cela, la notion de maximalité pose problème, comme le montre l'exemple suivant :

$$\mathcal{D} = \dfrac{\dfrac{\dfrac{\dfrac{}{\vdash a^{\perp}, a}\,ini}{\vdash a^{\perp}, \bot, a}\,[\bot] \qquad \dfrac{}{\vdash b^{\perp}, b}\,ini}{\vdash a^{\perp}, b^{\perp}, \bot, a \otimes b}\,\otimes \qquad \dfrac{\dfrac{}{\vdash \top, a \otimes b}\,[\top]}{\vdash \top \oplus \mathbf{0}, a \otimes b}\,[\oplus_0]}{\vdash a^{\perp}, b^{\perp}, \top \oplus \mathbf{0}, \bot \otimes \bot, a \otimes b, a \otimes b}\,[\otimes]$$

La preuve $\mathcal{D}$ est une preuve multifocalisée qui focalise sur $a^{\perp}, b^{\perp}, \top \oplus \mathbf{0}, \bot \otimes \bot$ ainsi que sur l'une des deux occurrences de $a \otimes b$. On constate évidemment que cette preuve est équivalente par permutation de la règle $[\top]$ à une preuve $\mathcal{D}'$ qui multifocalise sur l'autre occurrence de $a \otimes b$, or il n'existe pas de preuve de ce séquent qui multifocalise sur les deux occurrences en même temps. La restriction aux permutations iso-initiales nous permet d'éviter ce problème.

– Les permutations de l'inférence $[\bot]$ font perdre le lemme 10.13 pour le $\otimes$ puisqu'une inférence $[\bot]$ peut «se déplacer dans la preuve». Pour retrouver un résultat correspondant au lemme 10.13, il nous faudrait définir une notion de formule neutre dans une preuve $\mathcal{D}$, c'est-à-dire une formule n'ayant aucune sous-occurrence impliquée dans un axiome (ou dans le contexte d'une règle $[\top]$).
Plus fondamentalement, les permutations de la règle $[\bot]$ posent problème du point de vue du processus d'anti-réversibilité qui n'est plus suffisant pour maximiser les racines des preuves multifocalisées comme le montre l'exemple suivant :

$$\mathcal{D} = \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{D}_A}{\vdash A}}{\vdash A, \bot}\,[\bot]}{\vdash A, \bot \oplus C}\,[\oplus_0] \qquad \dfrac{\dfrac{\dfrac{\mathcal{D}_B}{\vdash B}}{\vdash B, \bot}\,[\bot]}{\vdash B, \bot \oplus D}\,[\oplus_0]}{\vdash A \otimes B, \bot \oplus C, \bot \oplus D}\,[\otimes]}{\vdash A \otimes B, (\bot \oplus C) \parr (\bot \oplus D)}\,[\parr]$$

En effet, l'anti-renversement ne peut pas amener la règle $[\parr]$ à permuter au-dessus de la règle $[\otimes]$ alors que $\mathcal{D}$ est équivalente à :

$$\mathcal{D}' = \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{D}_A}{\vdash A}}{\vdash A, \bot}\,[\bot]}{\vdash A, \bot, \bot}\,[\bot]}{\vdash A, \bot, \bot \oplus D}\,[\oplus_0]}{\vdash A, \bot \oplus C, \bot \oplus D}\,[\oplus_0]}{\vdash A, (\bot \oplus C) \parr (\bot \oplus D)}\,[\parr] \qquad \dfrac{\mathcal{D}_B}{\vdash B}}{\vdash A \otimes B, (\bot \oplus C) \parr (\bot \oplus D)}\,[\otimes]$$

Il faudrait donc faire précéder le processus d'anti-réversibilité d'un traitement préalable visant à assurer que diverses branches multiplicatives ne reçoivent pas d'occurrences de formules neutres comme dans l'exemple ci-dessus.

10.5. Multifocalisation et réseaux

Nous souhaitons dans cette section établir un résultat comparant preuves maximalement multifocalisées de MLL^- et réseaux de démonstration pour mettre ainsi en évidence la canonicité de notre notion de maximalité.

La manière usuelle d'aborder le problème de l'indentité des preuves en logique linéaire (et de donner une représentation canonique des preuves) consiste en effet à utiliser les réseaux de démonstration qui ont été initialement introduits par Girard [Gir87a] (voir section 1.5.2).

Puisque nous avons prouvé que les preuves maximalement multifocalisées fournissent également une notion de canonicité pour les preuves, il est naturel de comparer notre approche avec les réseaux de démonstration ; tel est le but de la présente section dans laquelle le fragment multiplicatif sans unité, MLL^-, pour lequel les réseaux de démonstration sont particulièrement simples et ont de très bonnes prorpiétés : nous donnerons une preuve directe que les preuves maximalement multifocalisées de MLL^- à iso-polarité près sont en correspondance bijective avec les réseaux de démonstration sans coupure de MLL^-. Les résultats précédents du chapitre nous assurent déjà qu'un tel résultat est vrai mais nous allons maintenant donner une preuve directe de ce fait en construisant explicitement la classe de preuves maximalement multifocalisées iso-polaires correspondant à un réseau de démonstration donné.

Nous sommes donc intéressés à fournir une preuve directe du théorème suivant :

Théorème 10.17

Deux preuves maximalement multifocalisées de MLL^- de $\vdash \cdot \Uparrow \Gamma$ sont iso-polaires si, et seulement si, elles ont le même réseau de démonstration.

Nous allons prouver le théorème en montrant qu'on peut associer à tout réseau de démonstration une unique preuve maximalement multifocalisée (à iso-polarité près). La réciproque, à savoir que deux preuves maximalement multifocalisées isopolaires correspondent au même réseau, est triviale.

Pour cela, nous commençons par adapter deux définitions de [AM99] dans lequel on trouve un algorithme de séquentialisation focalisé pour les réseaux de démonstration MLL^- :

Définition 10.18 (`split`(π), *d'après [AM99]*)

Soit π un réseau de démonstration MLL^-.

`split`(π) *est l'ensemble des conclusions F de π de la forme $A \otimes B$ telles qu'en enlevant le lien $\otimes$ conclusion de F on déconnecte π en deux réseaux de démonstration π_1 et π_2.*

Définition 10.19 (`foc`(π), *d'après [AM99]*)

Soit π un réseau de démonstration MLL^-.

`foc`(π) *est l'ensemble des conclusions F de π telles que*

- *soit F est un atome,*
- *soit $F \in$ `split`(π) et ses prémisses A et B sont respectivement conclusions de deux sous-réseaux π_1 et π_2 et sont telles que soit A (resp. B) est négative soit $A \in$ `foc`(π_1) (resp. $B \in$ `foc`(π_2)).*

Démonstration du Théorème 10.17 : Soit π un réseau de démonstration MLL^- de conclusions Γ et soient Γ^- les formules de Γ qui sont négatives et Γ^+ les formules de Γ qui sont positives ou atomiques[5]. Nous explicitons un algorithme de séquentialisation

[5] On utilisera cette notation tout au long de la preuve.

produisant une preuve maximalement multifocalisée de conclusion $\vdash \Gamma^+ \Uparrow \Gamma^-$ ce qui nous permettra d'obtenir une preuve maximalement multifocalisée de $\vdash \cdot \Uparrow \Gamma$ par applications répétées de $R\Uparrow$.

Nous raisonnons par induction sur la taille de π et par cas sur la structure de Γ :

<u>*Cas*</u> Γ *contient au moins une formule négative.*

Nous retirons toutes les cellules négatives (c'est-à-dire les cellules $⅋$) de π jusqu'à atteindre des cellules positives ou initiales. La structure de démonstration résultante est un réseau de preuve π' (elle vérifie de manière évidente le critère de correction) et ses conclusions Γ' sont positives ou atomiques.

Par hypothèse d'induction, nous pouvons séquentialiser π' en une preuve maximalement multi-focalisée $\mathcal{D}'$ de conclusion $\vdash \Gamma' \Uparrow \cdot$ (on a $\Gamma'^+ = \Gamma'; \Gamma'^- = \cdot$) et en séquentialisant dans n'importe quel ordre les règles négatives qui ont été enlevées dans l'étape précédente (les diverses possibilités donnent lieu à des preuves iso-polaires), nous obtenons ainsi une preuve $\mathcal{D}$ de la forme :

$$\mathcal{D} = \cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\mathcal{D}'}{\vdash \Gamma' \Uparrow \cdot}}{\vdots}\ R\Uparrow}{\vdash \cdot \Uparrow \Gamma'}\ R\Uparrow}{\vdots}\ [⅋]}{\vdash \cdot \Uparrow \Gamma}\ [⅋]$$

(les règles $R\Uparrow$ peuvent également permuter)

<u>*Cas*</u> Γ *ne contient que des formules positives ou atomiques.*

Si $\Gamma = a, a^\perp$, alors π est restreint à un lien axiome et on associe à π la preuve :

$$\mathcal{D} = \cfrac{}{\vdash a, a^\perp \Uparrow}\ ini$$

Sinon, puisque π est un réseau de démonstration, on a d'après [AM99] que $\texttt{foc}(\pi)$ est non vide et contient des formules non-atomiques. Considérons les formules de $\texttt{foc}(\pi)$: pour chaque $F \in \texttt{foc}(\pi)$, on considère les formules $(F_i)_{i \in I_F}$ obtenues en retirant héréditairement les connecteurs positifs de F jusqu'à atteindre une sous-formule positive ou atomique et on considère par ailleurs la structure de démonstration obtenue en retirant héréditairement les liens $\otimes$ de π pour les formules de $\texttt{foc}(\pi)$.

La structure de démonstration résultante n'est pas un réseau de démonstration puisqu'elle n'est pas connexe mais, en revanche, chacune de ses composantes connexes est un réseau (c'est immédiat d'après la définition de $\texttt{foc}(\pi)$). Soient $\pi_1, \ldots, \pi_n$ ces composantes connexes.

Pour $1 \leq i \leq n$, les conclusions Γ_i de π_i ont au moins une formule négative non-atomique ou bien π_i est réduit à un lien axiome. Dans le premier cas, on peut inductivement séquentialiser π_i en une preuve maximalement multifocalisée $\mathcal{D}_i$ de $\vdash \Gamma_i^+ \Uparrow \Gamma_i^-$, dans le second, on obtient une preuve du séquent $\vdash \Gamma_i \Uparrow \cdot$ restreinte à une règle initiale. Pour $1 \leq i \leq n$, on note Γ_i^{++} les formules positives de Γ_i^+ et Γ_i^{at} les formules atomiques de Γ_i^+ et on peut considérer $\mathcal{D}'_i$ qui a la forme :

$$\mathcal{D}'_i = \cfrac{\cfrac{\cfrac{\cfrac{\mathcal{D}_i}{\vdash \Gamma_i^+ \Uparrow \Gamma_i^-}}{\vdots}\ R\Uparrow}{\vdash \Gamma_i^{++} \Uparrow \Gamma_i^{at}, \Gamma_i^-}\ R\Uparrow}{\vdash \Gamma_i^{++} \Downarrow \Gamma_i^{at}, \Gamma_i^-}\ R\Downarrow$$

Pour conclure, il nous suffit maintenant de montrer que l'on obtient bien une preuve de $\vdash \Gamma \Uparrow \cdot$ à partir des $\mathcal{D}'_i$ et des cellules positives des formules de

$\texttt{foc}(\pi)$. Le caractère héréditairement splittant des formules non-atomiques de $\texttt{foc}(\pi)$ assure qu'il en est bien ainsi : en appliquant les règles $[\otimes]$ dans n'importe quel ordre (tant que cet ordre est compatible avec la structure de sous-formules bien sûr) on obtient une manière de séquentialiser π en une preuve $\mathcal{D}$ de la forme :

$$\mathcal{D} = \dfrac{\dfrac{\dfrac{\dfrac{\mathcal{D}'_1}{\vdash \Gamma_1^{++} \Downarrow \Gamma_1^{at}, \Gamma_1^-} \quad \cdots}{\vdots}[\otimes] \quad \cdots \quad \dfrac{\cdots \quad \dfrac{\mathcal{D}'_n}{\vdash \Gamma_n^{++} \Downarrow \Gamma_n^{at}, \Gamma_n^-}}{\vdots}[\otimes]}{\vdash \Gamma \setminus \texttt{foc}(\pi) \Downarrow \texttt{foc}(\pi)}[\otimes]}{\vdash \Gamma \Uparrow \cdot}[MF]$$

Pour terminer, il nous faut encore vérifier que la preuve obtenue par ce procédé est bien maximale, cela s'obtient aisément : soit F une formule qui pourrait potentiellement élargir l'ensemble de foyers et considérons une preuve $\mathcal{D}_F$ qui met en évidence cette possibilité, $\mathcal{D}_F$ focalise sur F.

En déséquentialisant $\mathcal{D}_F$, on obtient le réseau de démonstration π (puisque $\mathcal{D} \sim \mathcal{D}_F$) et puisque $\mathcal{D}_F$ est une séquentialisation de π qui focalise sur la formule F positive, alors F est héréditairement splittante dans π, c'est-à-dire que $F \in \texttt{foc}(\pi)$, et donc $\mathit{Root}(\mathcal{D}) = \texttt{foc}(\pi)$ est effectivement maximal.

Le procédé considéré dans cette preuve est non-déterministe (au sein d'une phase négative ou positive, nous séquentialisons *dans n'importe quel ordre*) et l'on vérifie aisément que les différentes preuves que l'on peut obtenir à partir de ce processus sont exactement les preuves maximalement multifocalisées de la classe d'iso-polarité correspondant au réseau de démonstration π.

□

Nous avons démontré dans cette section qu'il y a une bijection entre les réseaux de démonstration de MLL^- et les classes d'iso-polarité des preuves maximalement multifocalisées. Les réseaux de démonstration MLL^- sont certainement la présentation canonique des preuves la plus concise pour ce fragment.

Les preuves de séquents ont pourtant quelques avantages par rapport aux résultats actuellement connus pour les réseaux :

- Il est plus facile de vérifier la correction d'une preuve de séquent ;
- On ne sait pas bien faire de construction de réseaux de démonstration pour le moment. En particulier, il est mal-aisé de vérifier la correction d'un réseau en construction ;
- On ne sait pas traiter toute la logique linéaire de manière satisfaisante : seuls certains fragments de LL ont des présentations de réseaux vraiment satisfaisantes.

Troisième partie

Programmation Ludique

Chapitre 11

Introduction à la Ludique

Résumé:

Nous introduisons dans ce chapitre les notions de base de la Ludique [Gir01, Gir06].

Sommaire

«Qu'en est-il de l'interactionnisme symbolique ? Au lieu de se concentrer sur des notions abstraites comme le système social, il établit son domaine de recherche sur la concrétude des relations interindividuelles. Le monde social n'est pas préexistant à la manière d'une structure dont il faudrait s'accommoder, il est constamment créé et recréé par les interactions à travers des interprétations mutuelles suscitant un ajustement des acteurs les uns par rapport aux autres. L'interactionnisme traduit le souci d'identifier les processus à l'œuvre dans une société en train de se faire, il s'intéresse moins à l'institué qu'à l'instituant. Les normes et les règles sont l'objet d'une relecture constante, d'une négociation sociale, elles ne s'imposent pas de l'extérieur, les acteurs en sont les maîtres d'œuvre. C'est leur action mutuelle qui les rend effectives. Ainsi, par exemple, la déviance n'est pas une nature, répondant à la seule transgression d'une loi, elle est une construction sociale. Les lois sont souvent transgressées, mais parfois sans dommage pour les infracteurs. L'étiquette de déviance implique une mobilisation sociale et un processus de nomination.»

David le Breton
L'interactionnisme symbolique

Dans cette section, nous donnons les définitions de la plupart des objets et des notions considérés en ludique qui seront nécessaires pour faire de la recherche de preuve interactive.

11.1. Motivations et intuitions pour la Ludique

La ludique a été introduite par Girard [Gir01] comme une théorie interactive qui vise à dépasser la distinction traditionnelle entre syntaxe et sémantique en considérant que l'interaction devrait être une notion primitive et que la logique devrait être reconstruite après coup. Les desseins sont des objets intermédiaires entre syntaxe et sémantique qui peuvent interagir : on peut les voir à la fois comme des abstractions de preuves de séquents MALL et comme une présentation concrète des stratégies innocentes de la sémantique des jeux. Ainsi, on ne commence pas avec des objets syntaxiques auxquels on donne une interprétation sémantique, ni en choisissant un espace sémantique pour lequel on cherche un langage adéquat. Il s'agit d'objets interagissant les uns avec les autres dont les propriétés sont définies interactivement.

Nous décrivons dans les paragraphes qui suivent les motivations de base et les principales intuitions de la ludique.

Focalisation. La focalisation d'Andreoli [And92] (voir le chapitre 9) constitue le fondement d'une approche polarisée de la logique qui est une première étape vers une interprétation des preuves à base de jeux : la polarisation nous dit qui doit jouer. La focalisation permet de définir des connecteurs synthétiques et des règles d'inférence synthétiques en logique linéaire multiplicative et additive (MALL) et un calcul utilisant ces règles synthétiques. Les connecteurs de MALL peuvent être classifiés en deux ensembles de connecteurs : des connecteurs positifs ($\otimes, \oplus, \mathbf{1}, \mathbf{0}$) et des connecteurs négatifs ($\parr, \&, \perp, \top$). Quand on cherche une preuve, on alterne entre des phases positives et négatives. La prouvabilité ne peut pas être perdue pendant une phase négative alors que pendant une phase positive on peut faire un mauvais choix, ce qui entraînerait qu'on ne trouverait pas une preuve, même si le séquent avec lequel nous commençons était prouvable. Il y a alors clairement une phase active (la phase positive) et une phase passive (négative), les deux phases alternant.

Normalisation de preuves. Dans le processus d'élimination des coupures, une étape de conversion correspond à la sélection, par une règle positive, d'une continuation de la normalisation (comme par exemple la sélection de l'une des prémisses d'une règle $\&$ par une règle $\oplus$ pendant l'élimination des coupures). Mais il y a encore un problème pour qu'une interprétation interactive puisse tenir : il n'y a pas assez de preuves ! On ne peut pas trouver une preuve à la fois pour A et pour $A^\perp$. Pourtant, alors qu'on ne peut pas avoir une preuve à la fois pour une formule et sa négation, il est parfaitement acceptable de tenter de prouver A et $A^\perp$ à la fois. En particulier, si la recherche d'une preuve échoue, l'objet partiel que nous avons obtenu peut être utilisé dans une interaction avec des tentatives de preuve de la négation, excepté à la position où l'échec est rencontré (à ce point, la normalisation est encore indéfinie). Une tentative échouée de prouver A est un arbre dont certaines branches sont ouvertes. Ajoutons donc une nouvelle règle, le ***daimon***, pour noter le fait qu'une recherche de preuve a été interrompue : $\overline{\vdash \Gamma}$ ✠.

Quel peut être ce séquent $\vdash \Gamma$ où la recherche de preuve a été stoppée ? Ce serait stupide de s'arrêter si $\vdash \Gamma$ contient une formule négative puisqu'on sait que la décomposition de cette formule ne coûte rien (la prouvabilité ne peut pas être perdue à cause de cette règle d'inférence). On restreint donc l'application de la règle ✠ aux séquents qui ne sont constitués que de formules positives (les séquents positifs). On a donc des parapreuves pour tous séquents, même pour le séquent vide : $\vdash$.

La normalisation entre deux parapreuves est un processus à travers lequel chaque objet teste l'autre. La parapreuve qui utilise ✠ au cours de l'interaction est considérée comme le perdant de la partie et la partie se termine ici. On notera que ce processus de normalisation est donc une exploration des deux parapreuves : la coupure visite des portions des parapreuves au cours de la normalisation. Quand le daimon ✠ est atteint, les parapreuves sont dites ***orthogonales***. Une parapreuve qui gagne une interaction pourrait encore contenir un démon ailleurs : simplement, ce daimon ne ferait pas partie de cette interaction précise, mais elle serait détectée par une autre interaction.

Lieux. Alors qu'en programmation fonctionnelle il est important de savoir que les types d'une fonction et de son argument se correspondent, il n'est pas nécessaire, en recherche de preuve, de connaître toute la structure de la formule à prouver dès le début de la recherche. On a simplement besoin d'en connaître suffisamment sur cette structure pour choisir une règle d'inférence à appliquer. Cette idée se retrouve en ludique par l'utilisation d'adresses ou de lieux (ou ***loci***). Une formule est ainsi manipulée à travers son adresse ξ. Quand on applique une règle d'inférence R sur ξ on apprend *où* sont les sous-formules (ou ***sous-loci***) qui sont rendues disponibles par la règle R (et non pas *ce qu'elles sont*) : $\xi i, \xi j, \ldots$

Comportements et Incarnations. Une formule prouvable peut être interprétée comme l'ensemble de ses preuves ou parapreuves. En fait, les choses sont même plus radicales en ludique : les formules sont définies interactivement par une méthode de clôture par biorthogonal (habituelle en réalisabilité, sémantique des phases, etc.) ce qui définit des *comportements*. Étant donnée une parapreuve $\mathcal{D}$ dans un comportement $\mathbf{A}$ et une parapreuve $\mathcal{D}'$ dans $\mathbf{A}^\perp$, une partie de $\mathcal{D}$ peut être explorée interactivement en normalisant $\mathcal{D}$ avec $\mathcal{D}'$. Dans des circonstances particulières, il peut arriver que $\mathcal{D}$ soit entièrement visitée par $\mathcal{D}'$. Mais en général, il y a des parties de $\mathcal{D}$ qui ne peuvent pas être visitées, quel que soit $\mathcal{D}' \in \mathbf{A}^\perp$ avec lequel on le teste.

Il existe pourtant une classe de parapreuves particulièrement intéressantes dans une perspective interactive, la classe des parapreuves qui peuvent être entièrement visitées par normalisation avec des éléments de $\mathbf{A}^\perp$. Ces parapreuves sont dites ***materielles ou***

incarnées. Il s'agit des éléments les plus intéressants dans **A** puisqu'il peuvent être complètement caractérisés interactivement.

Connecteur et complétude interne. Les comportements sont donc des ensembles de desseins avec une propriété de clôture et ils représentent les formules, ou les types. Nous sommes donc intéressés à définir des connecteurs sur les comportements, c'est-à-dire des manières de construire de nouveaux comportements à partir de comportements plus simples.

La ludique possède une propriété remarquable à ce sujet, qui assure que pour de nombreux connecteurs, on a une construction directe de la partie essentielle d'un comportement (la partie matérielle) sans avoir besoin de clore notre construction par bi-orthogonal. C'est la propriété de complétude interne.

11.2. Actions et desseins

En ludique, les preuves sont remplacées par des desseins et les règles d'inférence par des actions tandis que les formules auxquelles une règle s'applique sont maintenant traitées à travers leur localisation.

Définition 11.1 (*Adresse*)

Une **adresse** *(ou* **lieu** *ou* **locus***) est une suite finie d'entiers (notée ξ).*

Définition 11.2 (*Action*)

Une **action** *est soit une paire constituée d'une adresse (le* **foyer***) et d'un ensemble fini d'entiers (la* **ramification***) et munie d'une polarité (on note $(\xi, I)^+$ ou $(\xi, I)^-$ et on parle d'***actions propres***), soit le* **daimon** *(✠) qui est une action positive.*

Quand on oublie la polarité d'une action propre κ, on parle d'une **action neutre** *et on écrit κ_ν.*

Définition 11.3 (*Justification*)

On dit qu'une action $(\xi, I)^\epsilon$ **crée** *les adresses $\xi \star i$ pour $i \in I$ et que l'action κ* **justifie** *κ' quand ce sont des actions propres de polarités opposées et que le foyer de κ' est créé par κ.*

Définition 11.4 (*Base*)

Une **base** *est un ensemble fini d'adresses polarisées (ξ^+ ou ξ^-) dont au plus une adresse est négative et telle qu'aucune adresse n'est préfixe d'une autre adresse de la base.*

On écrit $\xi \vdash \Lambda$ pour les **bases négatives** *et $\vdash \Lambda$ pour les* **bases positives**.

Une base qui est un singleton est dite **atomique** *et $\vdash$ est la* **base vide**.

Les desseins sont la contre-partie en ludique des preuves ou des stratégies. Plus précisément, les desseins correspondent à des preuves dans un calcul des séquents hyperséquentialisé (c'est-à-dire qui utilise les connecteurs synthétiques issus de la focalisation) d'où l'alternance de polarité des actions et la possibilité de branchements n-aires.

Définition 11.5 (*Desseins*)

Un **dessein** $\mathfrak{D}$ *sur une base* β *est un ensemble (éventuellement infini) clos par préfixe de suites finies d'actions (c'est-à-dire qu'il s'agit d'une forêt d'actions) tel que :*

- **Chroniques.** *Soit* $\chi = (\kappa_0, \dots, \kappa_n) \in \mathfrak{D}$.
  a) *Les actions de* χ *ont des polarités qui alternent et pour* $i < n$, κ_i *n'est pas* $\maltese$.
  b) *Chaque adresse a au plus une occurrence dans* χ.
  c) *Pour* $0 \leq i \leq n$ *alors*
     - *soit* $\kappa_i = \maltese$
     - *soit* $\kappa_i = (\xi, I)^\epsilon$ *et* $\xi^\epsilon \in \beta$
     - *soit* κ_i *est justifiée par* κ_j *avec* $j < i$.

     Si κ_{i+1} *est négative, elle est justifiée par* κ_i *;*
- **Positivité.** *Les feuilles de la forêt sont positives ;*
- **Branchement positif.** *L'arbre n'est branchant que sur les actions positives : si* $\chi_1, \chi_2 \in \mathfrak{D}$ *ne sont pas préfixes l'une de l'autre, alors elles diffèrent pour la première fois sur des actions négatives ;*
- **Partage additif.** *Si* κ_0 *et* κ_1 *sont deux actions différentes ayant le même foyer alors les suites qui conduisent à* κ_0 *et* κ_1 *diffèrent pour la première fois sur des actions (négatives) qui ont le même foyer :* $\kappa_0' = (\xi, I)^-$ *et* $\kappa_1' = (\xi, J)^-$ *;*
- **Totalité.** *Si la base est positive,* $\mathfrak{D} \neq \emptyset$.

Un dessein est **positif ou négatif** *selon sa base. Un dessein est dit* **atomique** *lorsque sa base est atomique.*

Remarque 11.6

Une autre approche des desseins est de les considérer comme générés co-inductivement par la grammaire suivante :

$$\boxed{\begin{array}{lll} \mathfrak{P} & ::= & \maltese^{\vdash\Gamma} \mid (\xi, I)^+ \cdot \{\mathfrak{N}_i^{\xi i \vdash \Gamma_i}, i \in I, i \neq j \Rightarrow \Gamma_i \cap \Gamma_j = \emptyset, \forall i \in I, \Gamma_i \subset \Gamma\}^{\vdash \xi, \Gamma} \\ \mathfrak{N} & ::= & \{(\xi, I)^- \cdot \mathfrak{P}_I^{\vdash \xi I, \Gamma_I}, I \in \mathcal{N} \subset \mathcal{P}_f(\omega), \forall I \in \mathcal{N}, \Gamma_I \subset \Gamma\}^{\xi \vdash \Gamma} \end{array}}$$

Par exemple, le dessein $\mathfrak{Fax}_{\xi \vdash \xi'}$ *est*

$$\mathfrak{Fax}_{\xi \vdash \xi'} = \{(\xi, I)^- \cdot (\xi', I)^+ \cdot \{\mathfrak{Fax}_{\xi' i \vdash \xi i}, i \in I\}, I \in \mathcal{P}_f(\omega)\}.$$

Définition 11.7 (*Tranche*)

Une **tranche** *est un dessein où chaque adresse apparaît au plus une fois. Une tranche d'un dessein* $\mathfrak{D}$ *est une tranche incluse dans* $\mathfrak{D}$. *En particulier, une tranche négative est un arbre.*

Quand on représente graphiquement les desseins et les tranches, on adopte ***La convention de Faggian*** : les actions positives sont encerclées et leur ramification est écrite à l'extérieur du cercle tandis que les actions négatives ne sont pas entourées.

On donne en figure 11.1 deux tranches et un dessein sur $\vdash \xi$. On remarquera que $\mathfrak{D}$ est la superposition de $\mathfrak{S}_1$ et de $\mathfrak{S}_2$. D'autres desseins sont montrés en figure 11.2.

11.3. Normalisation et Interaction

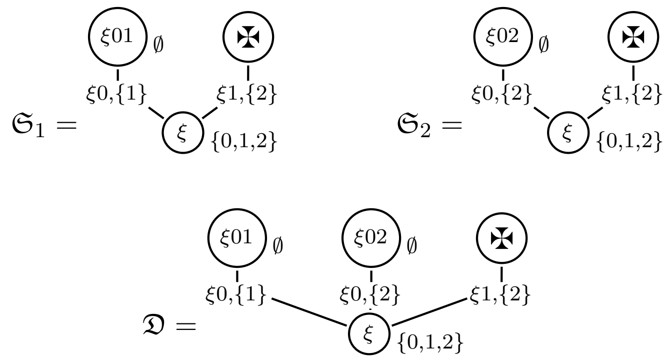


FIG. 11.1 – Deux tranches et un dessein.

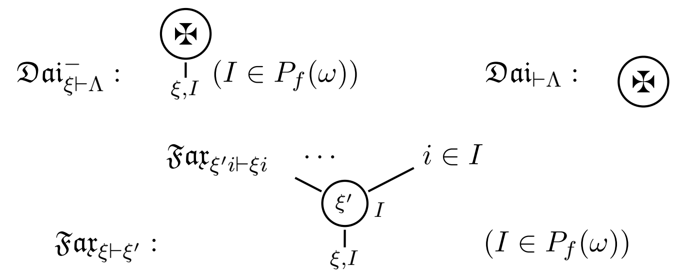


FIG. 11.2 – Des desseins importants : $\mathfrak{Dai}, \mathfrak{Dai}^-$ & $\mathfrak{Fax}$.

Les interactions de desseins sont construites par la normalisation des réseaux de coupure qui reflète l'élimination des coupures en calcul des séquents de la logique linéaire. En ludique, il n'y a pourtant pas de règle de coupure dans les desseins : une coupure est simplement la coïncidence d'un lieu avec des polarités opposées dans la base de deux desseins.

Définition 11.8 (*Réseau de coupure*)

Un **réseau de coupure** $\mathfrak{R} = (\mathfrak{D}_i)_{i\in I}$ *est un ensemble fini non vide de desseins sur des bases* $(\beta_i)_{i\in I}$ *tel que*

(i) les adresses des bases sont soit égales soit disjointes ;

(ii) une adresse ξ *apparaît dans au plus deux bases (dans ce cas, il apparaît une fois avec une polarité positive et une fois avec une polarité négative et il est appelé* **coupure** *dans* $\mathfrak{R}$*) ;*

(iii) les coupures définissent une relation binaire sur les desseins qui doit être connexe et acyclique.

Définition 11.9 (*Base d'un réseau de coupure*)

La **base** *de* $\mathfrak{R}$ *est l'ensemble des adresses polarisées des* $(\beta_i)_{i\in I}$ *qui* **ne sont pas** *des coupures.*

Définition 11.10 (*Réseau clos*)

Un réseau est dit **clos** *quand sa base est vide. Une action de $\mathcal{R}$ est* **visible** *s'il s'agit de ✠ ou si son foyer est une sous-adresse d'une adresse de la base du réseau, sinon elle est dite* **cachée**.

Définition 11.11 (*Dessein principal*)

Dans tout réseau de coupure, il y a un **dessein principal** *qui est le seul dessein positif du réseau si un tel dessein existe ou le seul dessein négatif dont la base est de la forme $\xi \vdash \Lambda$ avec ξ qui n'est pas une coupure.*

Alors que la condition de linéarité assure que dans une tranche, toutes les actions sont distinctes, un dessein peut contenir plusieurs copies de la même action. En conséquence, la description d'une action en donnant seulement son foyer et sa ramification est ambiguë et ne permet pas de la retrouver dans le dessein : nous avons besoin d'une information plus complète sur la *position* de l'action dans le dessein. La branche conduisant à l'action que nous considérons, en revanche est suffisante puisqu'elle montre le chemin qui doit être suivi dans le dessein pour atteindre l'action et cela sans ambiguïté. La définition suivante pour les vues fournit une manière de retrouver la chronique définissant une action pourvu que l'on connaisse un certain chemin dans les desseins d'un réseau de coupure conduisant à cette action.

Définition 11.12 (*Vues*)

Soit χ une suite d'actions neutres. Nous définissons les **vues positives et négatives** *pour χ par induction :*
– $\ulcorner\epsilon\urcorner^+ = \ulcorner\epsilon\urcorner^- = \epsilon$;
– $\ulcorner s \cdot (\xi, I)\urcorner^+ = \ulcorner s\urcorner^- \cdot (+, \xi, I)$;
– $\ulcorner s \cdot (\xi, I)\urcorner^- = \ulcorner t\urcorner^+ \cdot (-, \xi, I)$ *si $s = tu$ et u est le plus long suffixe de s tel qu'aucune action dans u ne crée ξ.*

On remarque que dans le dernier cas de la définition, t est soit vide, soit sa dernière action neutre est (σ, J) avec $\xi = \sigma j$ pour un certain $j \in J$. En outre, on peut trivialement étendre les vues positives aux suites se terminant avec ✠ (et cela même si ✠ n'est pas une action neutre).

Définition 11.13 (*Chemin de visite*)

Soit $\mathfrak{S}$ une tranche. Un chemin p dans $\mathfrak{S}$ (qui est une suite d'actions dans $\mathfrak{S}$) est un **chemin de visite** *s'il est tel que : (i) les polarités alternent ; (ii) il n'est fait que d'actions propres ; (iii) il est clos vers le bas (étant donné un préfixe p' de p dont la dernière action est κ, toutes les actions sous κ dans $\mathfrak{S}$ sont dans p').*

La polarité de p est la polarité de sa dernière action. Étant donné un chemin p, on écrit p_ν pour la suite d'actions neutres canoniquement associée à p.

On note qu'un chemin de visite ne peut pas nécessairement être réalisé par une interaction. La proposition suivante assure que les vues peuvent être utilisées pour parler de manière non ambiguë d'une action dans un dessein.

Proposition 11.14

Si $\mathfrak{S}$ est une tranche *sur une base atomique β et si p est un chemin de visite dans $\mathfrak{S}$ de polarité ϵ alors $\ulcorner p_\nu\urcorner^\epsilon$ est la branche de $\mathfrak{S}$ qui conduit jusqu'à la dernière action de p. Si la dernière action est κ, on notera la branche $Ch(\kappa)$.*

Définition 11.15 (*Loci Abstract Machine*)

La **Loci Abstract Machine** *(LAM [Fag02]), est une machine abstraite qui calcule l'interaction d'un réseau de coupure $\mathfrak{R}$. L'interaction est décrite comme le parcours de jetons dans le réseau de coupure. Soit $\mathfrak{R}$ un réseau de coupure de base β. Un* **jeton** *est une paire (s, κ) d'une suite d'actions neutres s et d'une action κ, qui représente la position du jeton dans $\mathfrak{R}$: s enregistre le chemin que le jeton a suivi depuis l'état initial jusqu'à κ. Soit $T_{\mathfrak{R}}$ l'ensemble de toutes les positions atteintes par les jetons durant la normalisation.*

- **Initialisation.** *Si κ est à la racine du dessein principal du réseau $\mathfrak{R}$ alors $(\epsilon, \kappa) \in T_{\mathfrak{R}}$;*
- **Transitions.** *Soit $(s, \kappa) \in T_{\mathfrak{R}}$. Il y a trois cas :*
  – **Visible.** *Si κ est une action visible de polarité ϵ, alors pour chaque κ' tel que $\ulcorner s\kappa \urcorner^{\epsilon}\kappa' \in \mathfrak{R}$, $(s\kappa_{\nu}, \kappa') \in T_{\mathfrak{R}}$ (on note que $\ulcorner s\kappa \urcorner^{\epsilon}$ est la chronique conduisant à κ) ;*
  – **Up.** *Si κ est une action négative cachée, alors soit κ' le successeur de la dernière action de $\ulcorner s\kappa \urcorner^{-}$, on a $(s\kappa, \kappa') \in T_{\mathfrak{R}}$;*
  – **Jump.** *Si κ est une action positive cachée, alors soit κ' la même action que κ mais de polarité opposée. Si $\ulcorner s\kappa \urcorner^{-} \in \mathfrak{R}$ alors $(s, \kappa') \in T_{\mathfrak{R}}$. Si $\ulcorner s\kappa \urcorner^{-} \notin \mathfrak{R}$ alors la normalisation échoue.*

Définition 11.16 (*Chemin de normalisation*)

Soit $\mathfrak{R}$ un réseau de coupure et soit $T_{\mathfrak{R}}$ l'ensemble des positions atteintes par les jetons durant la normalisation. Un **chemin de normalisation** *est une suite d'actions qui sont visitées pendant la normalisation de $\mathfrak{R}$: $Path(\mathfrak{R})$ est défini comme l'ensemble $\{s \cdot \kappa_{\nu} / (s, \kappa) \in T_{\mathfrak{R}}$ tel que s est maximale$\}$.*

On définit aussi

Définition 11.17 (*hide*(*p*))

$hide(p)$ comme la suite obtenue en enlevant toutes les actions cachées de p et **Hide($\mathfrak{R}$)** *comme l'ensemble $\{hide(p), p \in Path(\mathfrak{R})\}$.*

Définition 11.18 (s^{ϵ})

Si s est une séquence d'actions neutres, on définit s^{ϵ} comme :

(i) $\epsilon^{+} = \epsilon^{-} = \epsilon$;

(ii) $(s \cdot \kappa)^{+} = s^{-} \cdot \kappa^{+}$;

(iii) $(s \cdot \kappa)^{-} = s^{+} \cdot \kappa^{-}$.

Définition 11.19 (*Forme normale*)

La **forme normale** *d'un réseau de coupure $\mathfrak{R}$ est le dessein défini comme :*

$$[\![\mathfrak{R}]\!] = \{\chi/\chi \text{ est préfixe de } p^{+} \text{ avec } p \in Hide(\mathfrak{R})\}.$$

Définition 11.20 (*Dispute*)

Si $\mathfrak{R}$ est un réseau de coupure clos, on appelle **dispute** *le chemin de normalisation de $\mathfrak{R}$. Si le réseau est $\{\mathfrak{D}, \mathfrak{E}\}$, on écrit $[\mathfrak{D} \rightleftharpoons \mathfrak{E}]$ pour la dispute en question.*

11.4. Théorèmes analytiques

La Ludique est organisée autour de quatre théorèmes fondamentaux à propos de l'interaction de desseins à partir desquels on peut effectuer les constructions de connecteurs :

- le ***théorème de séparation*** est une version ludique du théorème de séparation (théorème de Böhm [Bö68]) du λ-calcul que nous avons étudié dans la partie de cette thèse consacrée au $\Lambda\mu$-calcul ;
- le ***théorème de stabilité*** affirme que la normalisation commute aux intersections compatibles ;
- le ***théorème d'associativité*** est une sorte de propriété de Church-Rosser concernant la normalisation des desseins (voir 10.1) ;
- le ***théorème de croissance*** indique que la normalisation est croissante pour l'ordre extensionnel.

Orthogonalité. L'orthogonalité décrit les normalisations qui ont été des succès :

Définition 11.21 (*Orthogonalité*)

Deux desseins atomiques $\mathfrak{D}, \mathfrak{E}$ sont **orthogonaux** *s'ils forment un réseau de coupure et si $[\![\mathfrak{D}, \mathfrak{E}]\!] = \maltese$. On écrit alors $\mathfrak{D} \perp \mathfrak{E}$.*

En général, si $\mathfrak{D}$ est un dessein de base $\xi_1^{\epsilon_1}, \ldots, \xi_n^{\epsilon_n}$ et si $(\mathfrak{E}_{\xi_i})_{1 \leq i \leq n}$ sont des desseins de bases atomiques $\xi_i^{-\epsilon_i}$, $(\mathfrak{D}, \mathfrak{E}_{\xi_1}, \ldots, \mathfrak{E}_{\xi_n})$ forme un réseau de coupure clos. Si $[\![\mathfrak{D}, \mathfrak{E}_{\xi_1}, \ldots, \mathfrak{E}_{\xi_n}]\!] = \maltese$ on écrit $\mathfrak{D} \perp (\mathfrak{E}_{\xi_i})_{1 \leq i \leq n}$.

Définition 11.22 (*Orthogonal d'un dessein*)

*L'***orthogonal** *d'un dessein $\mathfrak{D}$ de base atomique est :*

$$\mathfrak{D}^\perp = \{\mathfrak{E} \,/\, \mathfrak{D} \perp \mathfrak{E}\}.$$

Définition 11.23 (*Pré-ordre de préséance*)

On définit le **pré-ordre de préséance** *sur les desseins de même base comme suit :*

$$\mathfrak{D} \preccurlyeq \mathfrak{E} \iff \mathfrak{D}^\perp \subseteq \mathfrak{E}^\perp.$$

Théorème de séparation. Le théorème de séparation indique que le pré-ordre de préséance est en fait un ordre :

Théorème 11.24 (*Séparation*)

Soient $\mathfrak{D}, \mathfrak{E}$ deux desseins atomiques sur la même base. On a :

$$\mathfrak{D} = \mathfrak{E} \iff \forall \mathfrak{F}, [\![\mathfrak{D}, \mathfrak{F}]\!] = [\![\mathfrak{E}, \mathfrak{F}]\!].$$

et le pré-ordre de préséance est un ordre partiel sur les desseins.

Remarque 11.25

On a $\mathfrak{D} \preccurlyeq \mathfrak{E}$ si et seulement si toute chronique χ qui est dans $\mathfrak{D}$ sans être dans $\mathfrak{E}$ peut s'écrire sous la forme $\chi_1 \cdot \chi_2$ avec $\chi_1 \cdot \maltese \in \mathfrak{E}$.

Théorème de stabilité. On considère naturellement un autre ordre sur les desseins, l'inclusion de chroniques, que l'on appelle l'ordre stable :

Définition 11.26 (*Ordre stable*)

Soient $\mathfrak{D}$ et $\mathfrak{E}$ des desseins sur une même base. On définit **l'ordre stable** *comme l'ordre d'inclusion pour les desseins considérés comme ensembles de chronique :*

$$\mathfrak{D} \sqsubseteq \mathfrak{E} \iff \mathfrak{D} \subseteq \mathfrak{E}.$$

Théorème 11.27 (*Stabilité*)

Soit $(\mathfrak{R}_i)_{i\in I}$ une famille de réseaux de coupure tous inclus dans un réseau $\mathfrak{R}$ (pour tout $i \in I$, $\mathfrak{R}_i \subseteq \mathfrak{R}$). On a :

$$[\![\cap_{i\in I} \mathfrak{R}_i]\!] = \cap_{i\in I} [\![\mathfrak{R}_i]\!].$$

Remarque 11.28

L'intersection considérée dans l'énoncé ci-dessus est l'intersection des réseaux vus comme ensembles de chroniques.

Remarque 11.29

L'hypothèse d'intersection compatible est nécessaire pour assurer que l'on a bien égalité (sans cette hypothèse, des réseaux d'intersection nulle pourraient avoir la même forme normale par exemple).

Théorème d'associativité. Le théorème d'associativité est une sorte de propriété de Church-Rosser pour la ludique. On se rappellera la discussion de la section 10.1.

Théorème 11.30 (*Associativité*)

Soit $\mathfrak{R}$ un réseau et $\{\mathfrak{R}_1, \dots, \mathfrak{R}_n\}$ une partition de $\mathfrak{R}$ en un ensemble de réseaux (ce qu'on appelle un réseau de réseaux). On a :

$$[\![\mathfrak{R}]\!] = [\![[\![\mathfrak{R}_1]\!], \dots, [\![\mathfrak{R}_n]\!]]\!].$$

En combinant l'associativité et la séparation, on obtient le principe de clôture :

Théorème 11.31 (*Principe de clôture*)

Soit $\mathfrak{R}$ un réseau sur une base quelconque β. La forme normale de $\mathfrak{R}$, $[\![\mathfrak{R}]\!]$, est l'unique dessein $\mathfrak{D}$ tel que pour toute famille de desseins atomiques $(\mathfrak{E}_{\xi^\epsilon})$ (de base ξ^ϵ si $\xi^{-\epsilon} \in \beta$), on a $\mathfrak{D} \perp (\mathfrak{E}_{\xi^\epsilon})$ si, et seulement si, $[\![\mathfrak{R} \cup (\mathfrak{E}_{\xi^\epsilon}]\!])_{\xi^{-\epsilon}\in\beta} = \maltese$

Théorème de croissance. Le théorème de croissance affirme que la normalisation est croissante par rapport à l'ordre extensionnel :

Théorème 11.32 (*Croissance*)

Si $\mathfrak{D}_1 \preccurlyeq \mathfrak{E}_1, \dots, \mathfrak{D}_n \preccurlyeq \mathfrak{E}_n$, alors

$$[\![\mathfrak{D}_1, \dots, \mathfrak{D}_n]\!] \preccurlyeq [\![\mathfrak{E}_1, \dots, \mathfrak{E}_n]\!].$$

11.5. Comportements

> «L'interactionnisme s'intéresse à ce qui se joue entre les acteurs dans la détermination mutuelle de leur comportement.»
>
> David le Breton
>
> *L'interactionnisme symbolique*

Définition 11.33 (*Éthique*)

Un ensemble de desseins de même base atomique, noté $\mathbf{E}$*, est appelé une* **éthique** *et son orthogonal est* $\mathbf{E}^{\perp} = \{\mathfrak{D} \,/\, \forall \mathfrak{E} \in \mathbf{E}, \mathfrak{D} \perp \mathfrak{E}\}$.

Propriété 11.34

Étant données deux éthiques $\mathbf{E}$ *et* $\mathbf{E}'$*, on a :*

- *si* $\mathbf{E} \subseteq \mathbf{E}'$ *alors* $\mathbf{E}'^{\perp} \subseteq \mathbf{E}^{\perp}$ *;*
- $\mathbf{E} \subseteq \mathbf{E}^{\perp\perp}$ *;*
- $\mathbf{E}^{\perp} = \mathbf{E}^{\perp\perp\perp}$.

Définition 11.35 (*Comportement*)

Un **comportement** $\mathbf{G}$ *est une éthique qui est égale à son bi-orthogonal :* $\mathbf{G} = \mathbf{G}^{\perp\perp}$.

Il est évident que ***l'orthogonal d'une éthique*** est un comportement.

Définition 11.36 (*Comportement principal*)

Soit $\mathfrak{D}$ *un dessein. Le* **comportement principal** *de* $\mathfrak{D}$ *est* $\{\mathfrak{D}\}^{\perp\perp}$*. C'est le plus petit comportement (pour l'inclusion) contenant* $\mathfrak{D}$.

Proposition 11.37

Si $\mathfrak{E} \in \mathbf{G}$*, alors il existe un plus petit dessein* $\mathfrak{D} \subset \mathfrak{E}$ *tel que* $\mathfrak{D} \in \mathbf{G}$.

Définition 11.38 (*Incarnation d'un dessein dans un comportement*)

Le plus petit dessein $\mathfrak{D}$ *dont l'existence est assurée par la proposition précédente est l'***incarnation** *de* $\mathfrak{E}$ *dans* $\mathbf{G}$ *notée* $|\mathfrak{E}|_{\mathbf{G}}$.

Définition 11.39 (*Matérialité*)

Un dessein est dit **matériel** *dans un comportement quand il est égal à son incarnation dans son comportement.*

On note $|\mathbf{G}|$ *l'ensemble des desseins matériels du comportement* $\mathbf{G}$.

11.5.1 Connexité, disjonction, indépendance et extranéité

Définition 11.40 (*Répertoire d'un comportement*)

Soit $\mathbf{G}$ *un comportement atomique. Le* **répertoire** *de* $\mathbf{G}$*, noté* $\P\mathbf{G}$*, est une partie de* $\mathcal{P}_{fin}(\mathbb{N})$ *définie comme suit :*

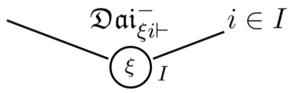
- *Si* $\mathbf{G}$ *est positif, alors* $\P\mathbf{G}$ *est l'ensemble des* $I \subset_{fin} \mathbb{N}$ *tels* $(\xi, I)^{+}$ *est la première action d'un dessein de* $\mathbf{G}$ *;*
- *Si* $\mathbf{G}$ *est négatif, alors* $\P\mathbf{G}$ *est l'ensemble des* $I \subset_{fin} \mathbb{N}$ *tels que* $(\xi, I)^{-}$ *est une première action d'un dessein incarné* $\mathfrak{D}$ *de* $\mathbf{G}$.

Le **répertoire d'une éthique** $\mathbf{E}$ *sera le répertoire du comportement associé* $\mathbf{E}^{\perp\perp}$.

Remarque 11.41

On peut également caractériser $\P\mathbf{G}$ *comme suit :*

- $\mathbf{G}$ *positif :* $I \in \P\mathbf{G}$ *si, et seulement si, le dessein suivant appartient à* $\mathbf{G}$ *:*



- $\mathbf{G}$ *négatif :* $I \in \P\mathbf{G}$ *si, et seulement si,* $\underset{\xi,I}{\maltese}$ *est l'une des tranches de l'incarnation du daimon négatif* $\mathfrak{Dai}^-_{\xi\vdash}$ *dans* $\mathbf{G}$.

On a par ailleurs la propriété suivante $\P\mathbf{G} = \P\mathbf{G}^\perp$.

Définition 11.42 (*Comportement connexe*)

Un comportement est **connexe** *si son répertoire est un singleton* $\{I\}$.

Définition 11.43 (*Comportements disjoints*)

Deux comportements $\mathbf{G}$ *et* $\mathbf{H}$ *de base identique sont dits* **disjoints** *si* $\P\mathbf{G}$ *et* $\P\mathbf{H}$ *sont des ensembles disjoints.*

On notera que cela signifie en particulier que, si $\mathbf{G}$ et $\mathbf{H}$ sont positifs, un dessein de $\mathbf{G}$ ne peut avoir la même première action qu'un dessein de $\mathbf{H}$.

Les notions de connexité et de disjonction s'étendent naturellement aux éthiques.

Définition 11.44 (*Réservoir d'un comportement*)

Le **réservoir d'un comportement**, *noté* $\S\mathbf{G}$, *est une partie de* $\mathbb{N}$ *(finie ou infinie) définie par :* $\S\mathbf{G} = \bigcup \P\mathbf{G}$.

Définition 11.45 (*Comportements indépendants, étrangers*)

- *Deux comportements* $\mathbf{G}$ *et* $\mathbf{H}$ *de même base sont* **indépendants** *lorsque*

$$\forall I, I' \in \S\mathbf{G}, \forall J, J' \in \S\mathbf{H}, \quad I \cup J = I' \cup J' \Rightarrow I = I' \text{ et } J = J'.$$

- *Deux comportements* $\mathbf{G}$ *et* $\mathbf{H}$ *de même base sont* **étrangers** *lorsque* $\S\mathbf{G} \cap \S\mathbf{H} = \emptyset$.

On peut bien évidemment parler de l'***indépendance de deux réservoirs*** R et R' ; en particulier, étant donné un élément J de $\{I \cup I', I \in R, I' \in R'\}$ pour R et R' des réservoirs indépendants, on peut reconstruire la partie venant de R et la partie venant de R'.

Deux comportements étrangers sont bien sûr indépendants : si $I \cup J = I' \cup J' = K$ avec $I, I' \in \S\mathbf{G}$ et $J, J' \in \S\mathbf{H}$, alors $K \cap \S\mathbf{G} = I = I'$ et $K \cap \S\mathbf{H} = J = J'$.

Deux comportements étrangers n'auront aucune adresse en commun à part le foyer de leur première action.

11.6. Connecteurs

Les comportements de la ludique représentent des formules ou des types, on veut donc définir des connecteurs permettant de construire des comportements à partir de comportements plus simples.

On va considérer :

- des pré-connecteurs qui sont des éthiques construites à partir de comportements, généralement définies à partir des desseins contenus dans les comportements (dans la mesure où on définit les pré-connecteurs à partir des desseins d'un comportement, on peut souvent les définir pour une éthique, sans demander qu'il s'agisse d'un comportement) ;
- et des connecteurs, qui seront le comportement engendré par le pré-connecteur via une clôture par double orthogonal.

11.6.1 Fléchage, $\updownarrow$

Le fléchage, ou décalage, correspond à un simple changement de polarité.

Définition 11.46 (***Fléchage d'une chronique, d'un dessein***)

Une chronique χ' de base β, ξ^{ϵ} est le **fléché d'une chronique** χ *de base $\beta, \xi i^{-\epsilon}$ si $\chi' = (\xi, \{i\})^{\epsilon} \cdot \chi$. On notera $\updownarrow \chi$ le fléché de χ.*

Un dessein $\mathfrak{D}'$ de base β, ξ^{ϵ} est le **fléché d'un dessein** $\mathfrak{D}$ *de base $\beta, \xi i^{-\epsilon}$ si $\mathfrak{D}'$ est obtenu en prenant le fléché de chaque chronique de $\mathfrak{D}$:*

$$\mathfrak{D}' = \{\updownarrow \chi, \chi \in \mathfrak{D}\} \cup \{(\xi, \{i\})^{\epsilon}\}$$

On pourra spécialiser la notation du fléché en fonction de la polarité : si χ et $\mathfrak{D}$ sont une chronique et un dessein positifs (resp. négatifs) on notera $\uparrow \chi$ et $\uparrow \mathfrak{D}$ (resp. $\downarrow \chi$ et $\downarrow \mathfrak{D}$) leur fléchés.

On étend naturellement la définition précédente à une éthique :

Définition 11.47 (***Pré-connecteur de décalage***)

Le **pré-fléché d'une éthique E** *de base $\xi i \vdash \Gamma$ (resp. $\vdash \Gamma, \xi i$), noté $\searrow \mathbf{E}$ (resp. $\nearrow \mathbf{E}$) est une éthique de base $\vdash \Gamma, \xi$ (resp. $\xi \vdash \Gamma$) définie comme suit :*

- *Si* **E** *est une éthique négative, le fléché de* **E** *est l'éthique*

$$\searrow \mathbf{E} = \{\downarrow \mathfrak{D}, \mathfrak{D} \in \mathbf{E}\} \cup \{\mathfrak{Dai}\};$$

- *Si* **E** *est une éthique positive, le fléché de* **E** *est l'éthique*

$$\nearrow \mathbf{E} = \{\uparrow \mathfrak{D}, \mathfrak{D} \in \mathbf{E}\}.$$

Définition 11.48 (***Connecteur de décalage***)

Un comportement **G** *est le* **fléché d'une éthique E**, *noté $\updownarrow \mathbf{E}$ s'il est de la forme :*

$$\mathbf{G} = \{\updownarrow \mathfrak{D}, \mathfrak{D} \in \mathbf{E}\}^{\perp\perp}$$

11.6.2 Connecteurs $\bigcup$ et $\bigcap$

Définition 11.49 (***Pré-connecteur*** $\bigcap$)

Soit $(\mathbf{G}_i)_{i\in I}$ une famille d'éthiques de même base β (d'une polarité quelconque). On définit un pré-connecteur $\bigcap(\mathbf{G}_i)_{i\in I}$ par

$$\bigcap(\mathbf{G}_i)_{i\in I} = \{\mathfrak{D}, \mathfrak{D} \in \mathbf{G}_i \forall i \in I\}.$$

Définition 11.50 (***Connecteur*** $\bigcap^{\perp\perp}$)

Soit $(\mathbf{G}_i)_{i\in I}$ une famille de comportements de même base β (d'une polarité quelconque). On définit le connecteur $\bigcap^{\perp\perp}$ par

$$\bigcap^{\perp\perp}(\mathbf{G}_i)_{i\in I} = \{\mathfrak{D}, \forall i \in I, \mathfrak{D} \in \mathbf{G}_i\}^{\perp\perp}.$$

Définition 11.51 (***Pré-connecteur*** $\bigcup$)

Soit $(\mathbf{G}_i)_{i\in I}$ une famille d'éthiques de même base β (d'une polarité quelconque). On définit un pré-connecteur $\bigcup(\mathbf{G}_i)_{i\in I}$ par

$$\bigcup(\mathbf{G}_i)_{i\in I} = \{\mathfrak{D}, \exists i \in I, \mathfrak{D} \in \mathbf{G}_i\}.$$

Définition 11.52 (***Connecteur*** $\bigcup^{\perp\perp}$)

Soit $(\mathbf{G}_i)_{i\in I}$ une famille de comportements de même base β (d'une polarité quelconque). On définit le connecteur $\bigcup^{\perp\perp}$ par

$$\bigcup^{\perp\perp}(\mathbf{G}_i)_{i\in I} = \{\mathfrak{D}, \exists i \in I, \mathfrak{D} \in \mathbf{G}_i\}^{\perp\perp}.$$

On a la propriété :

Théorème 11.53

Soit $\mathfrak{D}$ un dessein de $\bigcap(\mathbf{G}_i)_{i\in I}$. On a $|\mathfrak{D}|_{\bigcap(\mathbf{G}_i)_{i\in I}} = \bigcup_{i\in I} |\mathfrak{D}|_{\mathbf{G}_i}$.

Les connecteurs d'union et d'intersection sont des connecteurs totaux, qui sont définis quels que soit la famille de desseins qu'on leur donne, pourvu qu'ils soient de même base.

On va définir des connecteurs partiels, c'est-à-dire qu'ils ne sont définis qu'à condition que les comportements qu'on leur donne vérifient certaines propriétés. On va en fait demander que les comportements connectés soient disjoints et qu'ils soient polarisés de la bonne manière.

11.6.3 Connecteurs additifs

Connecteur avec, $\&$. On définit une conjonction additive, version négative et disjointe de l'intersection.

Définition 11.54 (***Connecteur*** $\&$)

Le connecteur $\&$ est le connecteur $\bigcap$ appliqué à des comportements négatifs disjoints.

On peut bien évidemment définir une version n-aire du connecteur $\&$, voire s'appliquant à une famille.

Connecteur plus, $\oplus$. On définit une disjonction additive, version positive et disjointe de l'union.

Définition 11.55 (***Connecteur*** $\oplus$)

Le connecteur $\oplus$ est le connecteur $\bigcup^{\perp\perp}$ appliqué à des comportements positifs disjoints.

On peut bien évidemment définir une version n-aire du connecteur $\oplus$, voire s'appliquant à une famille.

11.6.4 Connecteurs multiplicatifs

Définition 11.56 (***Tenseur de deux desseins***)

Soient $\mathfrak{D}$ et $\mathfrak{D}'$ deux desseins positifs sur la même base atomique $\vdash \xi$. On construit le dessein $\mathfrak{D} \odot \mathfrak{D}'$ comme suit :
- *Si $\mathfrak{D}$ ou $\mathfrak{D}'$ est $\mathfrak{Dai}$, alors $\mathfrak{D} \odot \mathfrak{D}' = \mathfrak{Dai}$;*
- *Sinon, soient I et I' les ramifications telles que $(\xi, I)^+$ et $(\xi, I)^+$ sont les premières actions de $\mathfrak{D}$ et $\mathfrak{D}'$ respectivement.*
  - *Si $I \cap I' \neq \emptyset$, alors $\mathfrak{D} \odot \mathfrak{D}' = \mathfrak{Dai}$;*
  - *Sinon, $I \cap I' = \emptyset$ et on définit $\mathfrak{D} \odot \mathfrak{D}'$ comme suit : Chaque chronique $(\xi, I)^+ \cdot \chi$ de $\mathfrak{D}$ est transformée en $(\xi, I \cup I')^+ \cdot \chi$, formant $\mathfrak{D}_{I \cup I'}$, chaque chronique $(\xi, I')^+ \cdot \chi'$ de $\mathfrak{D}'$ est transformée en $(\xi, I \cup I')^+ \cdot \chi'$, formant $\mathfrak{D}'_{I \cup I'}$, on pose alors $\mathfrak{D} \odot \mathfrak{D}' = \mathfrak{D}_{I \cup I'} \cup \mathfrak{D}'_{I \cup I'}$, qui est un dessein.*

Définition 11.57 (***Pré-connecteur tenseur***)

Si $\mathbf{E}$ et $\mathbf{F}$ sont deux éthiques positives sur une même base atomique, $\mathbf{E} \odot \mathbf{F}$ est l'éthique définie par :

$$\mathbf{E} \odot \mathbf{F} = \{\mathfrak{E} \odot \mathfrak{F}, \mathfrak{E} \in \mathbf{E}, \mathfrak{F} \in \mathbf{F}\}.$$

Définition 11.58 (***Tenseur de comportements***)

Si $\mathbf{G}$ et $\mathbf{H}$ sont deux comportements positifs sur une même base atomique, $\mathbf{G} \otimes \mathbf{H}$ est défini par :

$$\mathbf{G} \otimes \mathbf{H} = \{\mathfrak{D} \odot \mathfrak{E}, \mathfrak{D} \in \mathbf{G}, \mathfrak{E} \in \mathbf{H}\}^{\perp\perp}.$$

Proposition 11.59 (***voir [Gir01]***)

Soit $\mathfrak{F}$ un dessein négatif sur $\xi \vdash$, soit $\mathfrak{D}$ un dessein positif sur $\vdash \xi$. Il existe un unique dessein $(\mathfrak{F})\mathfrak{D}$ sur $\xi \vdash$ tel que pour tout $\mathfrak{E}$, dessein positif sur $\vdash \xi$, on ait

$$[\![\mathfrak{F}, \mathfrak{D} \odot \mathfrak{E}]\!] = [\![(\mathfrak{F})\mathfrak{D}, \mathfrak{E}]\!].$$

Définition 11.60 (***Adjonction***)

L'adjoint $(\mathfrak{F})\mathfrak{D}$ est l'unique dessein satisfaisant les conditions de la proposition ci-dessus.

Remarque 11.61

Par la commutativité du tenseur, on a également $[\![\mathfrak{F}, \mathfrak{D} \odot \mathfrak{E}]\!] = [\![(\mathfrak{F})\mathfrak{E}, \mathfrak{D}]\!]$.

Le connecteur $⅋$ est défini par orthogonalité :

Définition 11.62 (***Connecteur*** $⅋$)

Soient $\mathbf{G}, \mathbf{H}$ des comportements négatifs sur $\xi \vdash$, on a alors

$$\begin{aligned} \mathbf{G} ⅋ \mathbf{H} &= (\mathbf{G}^\perp \otimes \mathbf{H}^\perp)^\perp \\ &= \{\mathfrak{D} \odot \mathfrak{E}, \quad \mathfrak{D} \in \mathbf{G}^\perp, \mathfrak{E} \in \mathbf{H}^\perp\}^\perp. \end{aligned}$$

11.7. Propriétés de complétude interne

On présente dans cette section les résultats de complétude interne en insistant sur une notion essentiellement équivalente à la complétude interne mais qui est originale et met l'accent sur la notion de matérialité et de composition, la complétude matérielle.

Les propriétés de complétudes matérielles utilisent les propriétés de complétude interne dont on trouvera les preuves dans [Gir01].

11.7.1 Complétude interne

L'idée de la complétude interne de la ludique est que la définition des connecteurs n'ait pas besoin de clôture par bi-orthogonal : que la construction n'ait pas besoin d'être complétée.

Pour des raisons techniques, on ne peut pas obtenir, dans le cas général, un comportement en composant deux comportements. En revanche, on constate qu'on peut souvent construire les parties matérielles des comportements sans avoir à recourir au bi-orthogonal.

On définit alors la complétude d'une éthique non pas en demandant qu'elle soit égale à sont bi-orthogonal, mais comme suit :

Définition 11.63 (*Éthique complète*)

Soit $\mathbf{E}$ *une éthique.* $\mathbf{E}$ *est dite* **complète** *lorsque :*

$$\left|\mathbf{E}^{\perp\perp}\right| \subset \mathbf{E}$$

On dira qu'un connecteur a la propriété de complétude interne lorsque l'éthique obtenue est complète. On énonce la propriété de complétude interne dans la cas binaire, mais elle s'étend de manière évidente au cas unaire ou au cas n-aire :

Définition 11.64 (*Complétude interne*)

Soit $\bowtie$ *un connecteur binaire et soient* $\mathbf{G}$ *et* $\mathbf{H}$ *des comportements.* $\bowtie$ *a la propriété de* **complétude interne** *si* $\mathbf{G} \bowtie \mathbf{H}$ *est une éthique complète pour* $(\mathbf{G} \bowtie \mathbf{H})^{\perp\perp}$.

On va s'intéresser pour notre part à une variante de la complétude interne, essentiellement équivalente, mais mettant l'accent sur la matérialité et la composition de connecteurs. Nous l'appelons ***complétude matériellement interne***, ou ***complétude matérielle*** pour pouvoir la distinguer de la notion standard de complétude interne :

Définition 11.65 (*Complétude matérielle*)

Soit $\bowtie$ *un connecteur binaire et soient* $\mathbf{G}$ *et* $\mathbf{H}$ *des comportements.* $\bowtie$ *a la propriété de* **complétude matérielle** *si* $|\mathbf{G}| \bowtie |\mathbf{H}|$ *est une éthique complète pour* $\left|(\mathbf{G} \bowtie \mathbf{H})^{\perp\perp}\right|$, *c'est-à-dire si :*

$$\left|(\mathbf{G} \bowtie \mathbf{H})^{\perp\perp}\right| \subset |\mathbf{G}| \bowtie |\mathbf{H}|$$

11.7.2 Fléchage, $\updownarrow$

Proposition 11.66 (*Complétude interne du décalage*)

- *Si* $\mathbf{G}$ *est un comportement négatif, alors* $\downarrow \mathbf{G} = \searrow \mathbf{G}$ *;*
- *Si* $\mathbf{G}$ *est un comportement positif, alors* $\nearrow \mathbf{G}$ *est une éthique complète pour* $\uparrow \mathbf{G}$.

Dans le cas négatif, on a vraiment une égalité tandis que dans le cas positif, on n'a qu'une éthique complète parce qu'il manque des desseins non-incarnés qui ont des tranches supplémentaires, jamais testés dans $\uparrow \mathbf{G}$ (des desseins négatifs qui ont parmi leur premières actions des actions de la forme $(\xi, I)^-, I \neq \{i\}$).

Proposition 11.67 (*Complétude matérielle du décalage*)

- *Si $\mathbf{G}$ est un comportement négatif, alors $|\downarrow \mathbf{G}| = \searrow |\mathbf{G}|$;*
- *Si $\mathbf{G}$ est un comportement positif, alors $|\uparrow \mathbf{G}| = \nearrow |\mathbf{G}|$.*

Démonstration : On se limite au cas atomique.

Supposons que $\mathbf{G}$ soit un comportement négatif. Soit $\mathfrak{D} \in |\downarrow \mathbf{G}|$ un dessein matériel. De deux choses l'une : soit $\mathfrak{D} = \mathfrak{Dai}$ et dans ce cas, on a $\mathfrak{D} \in \searrow |\mathbf{G}|$ par définition de $\searrow |\mathbf{G}|$, soit $\mathfrak{D} \neq \mathfrak{Dai}$. Dans le second cas, $\mathfrak{D}$ est un dessein de base $\vdash \xi$ de la forme :

$$\begin{array}{c}\mathfrak{D}' \\ | \\ \xi \; {\scriptstyle \{i\}}\end{array} \qquad \text{où } \mathfrak{D}' \in \mathbf{G}.$$

Il nous suffit alors de prouver que $\mathfrak{D}'$ est matériel dans $\mathbf{G}$ pour avoir le résultat. Soit $\mathfrak{E}' \subset \mathfrak{D}'$ avec $\mathfrak{E}' \in \mathbf{G}$. On a alors $\downarrow \mathfrak{E}' \in \downarrow \mathbf{G}$ par définition du fléchage et $\downarrow \mathfrak{E}' \subset \mathfrak{D}$. La matérialité de $\mathfrak{D}$ entraîne l'égalité qui elle-même a pour conséquence que $\mathfrak{E}' = \mathfrak{D}'$. On en déduit la minimalité de $\mathfrak{D}'$ dans $\mathbf{G}$, c'est-à-dire que $\mathfrak{D}'$ est matériel et donc que $\mathfrak{D} \in \searrow |\mathbf{G}|$.

Supposons maintenant que $\mathbf{G}$ soit un comportement positif, et soit $\mathfrak{D} \in |\uparrow \mathbf{G}|$ un dessein matériel de $\mathbf{G}$ de base $\xi \vdash$.

$\mathfrak{D}$ est de la forme $\begin{array}{c}\mathfrak{D}' \\ | \\ {\scriptstyle \xi, \{i\}}\end{array}$, c'est-à-dire de la forme $\uparrow \mathfrak{D}'$ avec $\mathfrak{D}' \in \mathbf{G}$. On montre comme précédemment que $\mathfrak{D}'$ est matériel en montrant qu'un dessein $\mathfrak{E}'$ de $\mathbf{G}$ inclus dans $\mathfrak{D}'$ permet d'obtenir un dessein $\mathfrak{E}$ de $\uparrow \mathbf{G}$ inclus dans $\mathfrak{D}$ et donc égal à $\mathfrak{D}$ entraînant l'égalité de $\mathfrak{E}'$ et $\mathfrak{D}'$ d'où l'on peut conclure à la matérialité de $\mathfrak{D}'$ et donc à l'inclusion de $|\uparrow \mathbf{G}|$ dans $\nearrow |\mathbf{G}|$.

Les inclusions $|\downarrow \mathbf{G}| \supset \searrow |\mathbf{G}|$ et $|\uparrow \mathbf{G}| \supset \nearrow |\mathbf{G}|$ sont faciles à obtenir. En effet, soit $\mathfrak{D}$ un dessein de la forme $\updownarrow \mathfrak{D}'$ avec $\mathfrak{D}'$ matériel dans $\mathbf{G}$, on a alors $\mathfrak{D}$ dans $\updownarrow \mathbf{G}$ et la minimalité de $\mathfrak{D}$ vient de la complétude interne du décalage. On sait en effet que si $\mathfrak{E} \subset \mathfrak{D}$ avec $\mathfrak{E} \in \updownarrow \mathbf{G}$, alors $\mathfrak{E}$ est de la forme $\updownarrow \mathfrak{E}'$ pour un certain dessein $\mathfrak{E}'$ de $\mathbf{G}$ et on a $\mathfrak{E}' \subset \mathfrak{D}'$, d'où l'égalité $\mathfrak{E} = \mathfrak{D}$. Si enfin on est dans le cas où $\mathbf{G}$ est négatif et que $\mathfrak{D}$ est $\mathfrak{Dai}$, alors il est évidemment incarné dans $\downarrow \mathbf{G}$ et ceci conclut la preuve des inclusions $|\downarrow \mathbf{G}| \supset \searrow |\mathbf{G}|$ et $|\uparrow \mathbf{G}| \supset \nearrow |\mathbf{G}|$ d'où nous concluons à l'égalité.

□

11.7.3 Connecteurs $\bigcup$ et $\bigcap$

Proposition 11.68 (*Complétude interne de $\bigcap$*)

Le pré-connecteur $\bigcap$ est en fait un connecteur : si $(\mathbf{G}_i)_{i \in I}$ une famille de comportements de même base β, $\bigcap(\mathbf{G}_i)_{i \in I}$ est un comportement.

Démonstration : On a en effet :

$$\begin{array}{rcl}\bigcap(\mathbf{G}_i)_{i \in I} & = & \bigcap_{i \in I} \mathbf{G}_i^{\perp\perp} \\ & = & \{\mathfrak{D}, \mathfrak{D} \in \mathbf{G}_i^{\perp\perp}, \forall i \in I\} \\ & = & \{\mathfrak{D}, \mathfrak{D} \perp \mathfrak{E}, \forall \mathfrak{E} \in \mathbf{G}_i^{\perp}, \forall i \in I\} \\ & = & \{\mathfrak{D}, \mathfrak{D} \perp \mathfrak{E}, \forall \mathfrak{E} \in \bigcup_{i \in I} \mathbf{G}_i^{\perp}\} \\ & = & (\bigcup_{i \in I} \mathbf{G}_i^{\perp})^{\perp}\end{array}$$

□

Par contre, $\bigcap$ ne vérifie pas la propriété de complétude matérielle. En effet, il n'y a aucune raison pour que $\bigcap(|\mathbf{G}_i|)_{i\in I}$ contienne les desseins matériels de $\bigcap(\mathbf{G}_i)_{i\in I}$ comme l'indique le théorème 11.53.

On n'a pas de propriété de complétude interne pour l'union, et a fortiori pas non plus de résultat de complétude matérielle.

Remarque 11.69

L'exemple du connecteur $\bigcap$ montre que complétude interne et complétude matérielle ne coïncident pas nécessairement.

11.7.4 Connecteurs additifs

Mystère de l'incarnation. Le connecteur & vérifie une propriété de complétude interne que Girard appelle le mystère de l'incarnation :

Proposition 11.70 (*Mystère de l'incarnation*)

Soient $\mathbf{G}$ et $\mathbf{H}$ deux comportements disjoints sur une même base négative β. On a :

$$|\mathbf{G} \,\&\, \mathbf{H}| = |\mathbf{G}| \times |\mathbf{H}|$$

Dans la proposition précédente, le produit cartésien considéré est un peu particulier puisqu'il est défini comme suit : si $\mathbf{E}, \mathbf{F}$ sont des éthiques, $\mathbf{E} \times \mathbf{F} = \{\mathfrak{E} \cup \mathfrak{F}, \mathfrak{E} \in \mathbf{E}, \mathfrak{F} \in \mathbf{F}\}$ où $\mathfrak{E} \cup \mathfrak{F}$ est bien sûr l'union de desseins pris comme ensembles de chroniques. La condition de disjonction et la matérialité assurent que les ensembles de chroniques obtenus de cette manière sont bien des desseins.

On notera que le mystère de l'incarnation énonce également une propriété de complétude interne matérielle. C'est d'ailleurs de là que vient la formulation de la notion de complétude matérielle.

Propriété de la disjonction. On a la complétude interne pour le connecteur $\oplus$:

Proposition 11.71 (*Propriété de la disjonction*)

Soient $\mathbf{G}$ et $\mathbf{H}$ deux comportements disjoints sur une même base positive β. On a :

$$\mathbf{G} \oplus \mathbf{H} = \mathbf{G} \bigcup \mathbf{H}$$

Complétude matérielle pour $\oplus$. On a également une propriété de complétude matérielle pour $\oplus$:

Proposition 11.72

$$|\oplus_{i\in I} \mathbf{G}_i| \subset \oplus_{i\in I} |\mathbf{G}_i|$$

Démonstration : En effet, soit $\mathfrak{D}$ matériel dans $\oplus_{i\in I} \mathbf{G}_i$, les $\mathbf{G}_i$ étant deux à deux disjoints. On sait par la propriété de disjonction que $\mathfrak{D} \in \mathbf{G}_k$ pour un certain $k \in I$. Soit $\mathfrak{E} \subset \mathfrak{D}$ avec $\mathfrak{E} \in \mathbf{G}_k$. On a alors $\mathfrak{E} \in \oplus_{i\in I} \mathbf{G}_i$, mais alors la matérialité de $\mathfrak{D}$ dans $\oplus_{i\in I} \mathbf{G}_i$ assure que $\mathfrak{E} = \mathfrak{D}$ d'où le résultat.

On a même une réciproque : soit $\mathfrak{D} \in \oplus_{i\in I} |\mathbf{G}_i|$. Il existe donc k tel que $\mathfrak{D}$ est matériel dans $\mathbf{G}_k$ et du fait de l'hypothèse de comportements disjoints, ce k est unique sauf si $\mathfrak{D} = \mathfrak{Dai}$. Si $\mathfrak{D} = \mathfrak{Dai}$, alors il est évidemment matériel dans $\oplus_{i\in I} \mathbf{G}_i$, sinon on considère l'unique indice k mentionné plus haut, et on l'utilise pour montrer que $\mathfrak{D}$ est matériel dans $\oplus_{i\in I} \mathbf{G}_i$. Soit $\mathfrak{E} \subset \mathfrak{D}$ avec $\mathfrak{E} \in \oplus_{i\in I} \mathbf{G}_i$. Il existe i tel que $\mathfrak{E} \in \mathbf{G}_i$, mais $\mathfrak{E} \subset \mathfrak{D}$ entraîne que $\mathfrak{D} \in \mathbf{G}_i$ et donc que $i = k$, on peut donc utiliser l'hypothèse de matérialité de $\mathfrak{D}$ dans $\mathbf{G}_k$ pour conclure que $\mathfrak{E} = \mathfrak{D}$ et donc que $\mathfrak{D}$ est matériel dans $\oplus_{i\in I} \mathbf{G}_i$ ce qui achève la preuve.

□

Décomposition additive. On peut décomposer des comportements en comportements plus simples, connectés de manière additive :

Proposition 11.73 (*Décomposition additive*)

- *Tout comportement positif $\mathbf{G}$ peut se décomposer sous la forme $\oplus_{I\in\P\mathbf{G}} \mathbf{G}_I$ où les comportements $\mathbf{G}_I$ sont des comportements connexes ;*
- *Tout comportement négatif $\mathbf{G}$ peut se décomposer sous la forme $\&_{I\in\P\mathbf{G}} \mathbf{G}_I$ où les comportements $\mathbf{G}_I$ sont des comportements connexes.*

11.7.5 Connecteurs multiplicatifs

On n'a pas de propriété de complétude interne avec des comportements généraux. On va devoir ajouter une condition sur les comportements pour retrouver une propriété de complétude interne, il s'agit de la condition d'indépendance :

Proposition 11.74 (*Complétude interne du $\otimes$*)

Si $\mathbf{G}$ et $\mathbf{H}$ sont des comportements positifs indépendants, alors $\mathbf{G} \odot \mathbf{H}$ est une éthique complète pour $\mathbf{G} \otimes \mathbf{H}$.

On notera que la condition d'***extranéité*** assure *a fortiori* la complétude interne.

La proposition suivante fournie un résultat de complétude interne pour le connecteur $\parr$:

Proposition 11.75

Soient $\mathbf{G}, \mathbf{H}$ des comportements négatifs sur $\xi \vdash$, on a alors :

$$\begin{aligned} \mathfrak{F} \in \mathbf{G} \parr \mathbf{H} \quad &\Leftrightarrow \quad \forall \mathfrak{D} \in \mathbf{G}^{\perp}, \quad (\mathfrak{F})\mathfrak{D} \in \mathbf{H} \\ &\Leftrightarrow \quad \forall \mathfrak{E} \in \mathbf{H}^{\perp}, \quad (\mathfrak{F})\mathfrak{D} \in \mathbf{G} \end{aligned}$$

Complétude matérielle pour les multiplicatifs. On montre que le connecteur $\otimes$ satisfait lui aussi une propriété de complétude matérielle.

Proposition 11.76 (*Complétude matérielle pour $\otimes$*)

$$|\mathbf{G} \otimes \mathbf{H}| \subset |\mathbf{G}| \odot |\mathbf{H}|$$

Démonstration : Soit $\mathfrak{D}$ matériel dans $\mathbf{G} \otimes \mathbf{H}$. On a $\mathfrak{D} \in \mathbf{G} \odot \mathbf{H}$ par complétude interne de $\otimes$ et indépendance de $\mathbf{G}$ et $\mathbf{H}$.

De deux choses l'une. Soit $\mathfrak{D} = \mathfrak{Dai}$ et alors $\mathfrak{D} = \mathfrak{Dai} \odot \mathfrak{Dai}$ avec $\mathfrak{Dai}$ matériel dans $\mathbf{G}$ et $\mathbf{H}$, d'où $\mathfrak{D} \in |\mathbf{G}| \odot |\mathbf{H}|$, soit il existe $\mathfrak{E} \in \mathbf{G}$ et $\mathfrak{F} \in \mathbf{H}$ tels que $\mathfrak{D} = \mathfrak{E} \odot \mathfrak{F}$ avec $(\xi, I)^+$ première action de $\mathfrak{E}$, $(\xi, J)^+$ première action de $\mathfrak{F}$ et $(\xi, I \cup J)^+$ première action de $\mathfrak{D}$ ($I \cap J = \emptyset$).

Soient alors $\mathfrak{E}' \in \mathbf{G}$ et $\mathfrak{F}' \in \mathbf{H}$ tels que $\mathfrak{E}' \subset \mathfrak{E}$ et $\mathfrak{F}' \subset \mathfrak{F}$. $\mathfrak{D}' = \mathfrak{E}' \odot \mathfrak{F}'$ est un dessein de $\mathbf{G} \otimes \mathbf{H}$ qui commence par une action $(\xi, I \cup J)^+$ du fait de l'indépendance de $\mathbf{G}$ et $\mathbf{H}$ et donc $\mathfrak{D}' \subset \mathfrak{D}$. La matérialité de $\mathfrak{D}$ nous assure alors que $\mathfrak{D}' = \mathfrak{D}$ d'où l'on déduit, par l'indépendance de $\mathbf{G}$ et $\mathbf{H}$ que $\mathfrak{E}' = \mathfrak{E}$ et $\mathfrak{F}' = \mathfrak{F}$ et donc que $\mathfrak{E}$ et $\mathfrak{F}$ sont matériels dans $\mathbf{G}$ et $\mathbf{H}$ respectivement d'où $\mathfrak{D} \in |\mathbf{G}| \odot |\mathbf{H}|$.

□

Complétude matérielle du $\parr$. On donne un énoncé de complétude matérielle pour le $\parr$ dérivé de la proposition 11.75

Proposition 11.77

Soient $\mathbf{G}, \mathbf{H}$ *des comportements négatifs sur* $\xi \vdash$*, on a alors :*

$$\begin{aligned} \mathfrak{F} \in \mathbf{G} \parr \mathbf{H} \quad &\Leftrightarrow \quad \forall \mathfrak{D} \in \left|\mathbf{G}^{\perp}\right|, \quad (\mathfrak{F})\mathfrak{D} \in \mathbf{H} \\ &\Leftrightarrow \quad \forall \mathfrak{E} \in \left|\mathbf{H}^{\perp}\right|, \quad (\mathfrak{F})\mathfrak{D} \in \mathbf{G} \end{aligned}$$

La propriété peut se réécrire :

$$\mathbf{G} \parr \mathbf{H} = \{\mathfrak{F}, \quad \forall \mathfrak{D} \in \left|\mathbf{G}^{\perp}\right|, \exists \mathfrak{E} \in |\mathbf{H}|, (\mathfrak{F})\mathfrak{D} \supset \mathfrak{E}\}$$

Chapitre 12

Vers une Programmation Ludique : Recherche de Preuve Interactive

Résumé:

Nous proposons dans ce chapitre un cadre pour la «recherche de preuve interactive» c'est-à-dire une recherche de preuve non plus guidée par séquent et des instructions de recherche de preuve, mais guidée par la nécessité d'interagir avec un ensemble donné de tests. Nous commençons par motiver notre approche puis nous donnons quelques exemples informels dans un calcul adapté de MALL avant de passer au cadre ludique et de proposer une machine abstraite pour une construction interactive de desseins, la SLAM.

Nous expliquons comment le backtracking peut être traité dans ce cadre et nous présentons quelques exemples de recherche de preuve interactive.

Références : Les résultats de ce chapitre ont donné lieu à publication dans l'article *Towards Ludics Programming : Interactive Proof Search* [Sau08b].

Sommaire

12.1. Motivation de la recherche de preuve interactive

Théorie de la démonstration et calcul. De récentes avancées en théorie de la démonstration ont conduit à des développements majeurs en théorie des langages de programmation. La modélisation du calcul par des preuves a eu un impact profond sur les études fondamentales des langages de programmation de même que sur les questions pratiques liées à leur implémentation, en fournissant de puissants outils d'analyse formelle des propriétés de programmes. Les langages de programmation déclaratifs ont été reliés de plusieurs manières à la théorie mathématique de preuves : d'un côté, le paradigme du «calcul comme réduction de preuve» a fourni un fondement aux langages de programmation fonctionnelle grâce à la correspondance bien connue de Curry-Howard [How80]. D'un autre côté, le paradigme de «calcul comme recherche de preuve» a fourni un fondement à la programmation logique : le calcul d'un programme y est vu comme la recherche d'une preuve dans un certain système déductif.

Le calcul comme recherche de preuve. Les preuves uniformes et les langages de programmation logique abstraits [MNPS91] ainsi que la propriété de focalisation [And92] (voir chapitre 9) en logique linéaire [Gir87a] ont permis de considérer la recherche de preuve comme un modèle de calcul pour des fragments beaucoup plus riches que les systèmes originaux de clauses de Horn du premier ordre pour lesquels on utilise la méthode de résolution et qui fondaient la programmation logique (Formules héréditaires de Harrop, logiques d'ordre supérieur, logique linéaire) et de bénéficier des propriétés géométriques et de la structure riche du calcul des séquents ce qui a enrichi la dynamique de la recherche et donc les formes de calcul que l'on pouvait modéliser logiquement. Ainsi, dans le modèle des preuves uniformes, précurseur de la focalisation, le calcul sera modélisé comme la recherche d'une preuve d'un séquent $\mathcal{P} \vdash G$ où $\mathcal{P}$ représente le programme logique et où G représente le but à calculer. Le calcul procède alors comme la recherche d'une preuve de $\mathcal{P} \vdash G$, recherche dirigée par le but G, le programme logique $\mathcal{P}$ n'étant utilisé que grâce au backchaining (chaînage arrière) quand le but est une formule atomique.

Cela a profondément influencé la conception des langages de programmation logique en permettant de capturer de nombreuses primitives calculatoires supplémentaires de manière logique (programmation d'ordre supérieur, modules, gestion des ressources, primitives de concurrence...). Néanmoins, des constructions essentielles des langages logiques n'ont pas pu être traitées de manière satisfaisante en conservant une approche logique en particulier en ce qui concerne les mécanismes de contrôle de l'exécution [Nai86, Nai95] (backtracking, backtracking intelligent, prédicat de coupure (`cut`)...). En conséquence, des parties importantes de ces langages n'ont pas de sémantique logique bien établie. Cela entraîne qu'il n'est pas facile d'analyser et de raisonner à propos des programmes qui utilisent ces constructions même s'ils sont extrêmement nombreux en programmation en Prolog.

L'une des directions de recherche qui remonte presque aux début de la recherche de preuve comme paradigme de calcul consiste à chercher à traiter logiquement ces primitives extra-logiques de manière à se rapprocher de la correspondance idéale : «Algorithme = Logique» [Mil99]. On peut à ce sujet dresser une comparaison qui nous semble utile et éclarairante avec le cas de la programmation fonctionnelle : le fait d'étendre la correspondance de Curry-Howard à la logique classique a permis de capturer logiquement plusieurs opérateurs de contrôle qui étaient utilisés en pratique (comme le `call/cc` notamment) grâce aux règles de typage de ces opérateurs [Gri90b] ou grâce à des extensions du λ-calcul comme le $\lambda\mu$-calcul de Parigot [Par92] (voir la première partie de cette thèse).

Pourquoi est-il si difficile de traiter logiquement le contrôle ? Pourtant, jusqu'à présent, on n'est pas parvenu à un tel résultat du côté de la programmation logique, ce qui peut être vu comme le résultat d'un décalage entre la recherche de preuve en calcul des séquents et la programmation logique : alors qu'en calcul des séquents on manipule des preuves, le processus de recherche et de construction d'une preuve n'a pas affaire à des preuves jusqu'à ce que le calcul soit achevé. Au lieu de cela, les objets de la recherche de preuve sont des preuves partielles (ou des preuves ouvertes) qui peuvent se révéler ne pas conduire du tout à une preuve mais à un échec. De telles recherches de preuve échouées ne font pas partie de la théorie de la démonstration du calcul des séquents.

En outre, alors qu'on considère qu'un séquent représente l'état d'un calcul, l'élément essentiel pour la recherche de preuve ne réside pas dans les séquents eux-mêmes, mais dans les règles d'inférence qui sont appliquées à ces séquents. Ainsi, la dynamique elle-même du calcul ne correspond pas bien à la théorie dans laquelle nous travaillons.

Enfin, un autre élément qui peut contribuer à rendre difficile la modélisation du contrôle dans les approches habituelles avec le calcul des séquents est le caractère inhomogène de cet approche dans laquelle une partie du calcul est codée dans la logique utilisée, une autre partie réside dans la grammaire de formule utilisable pour les clauses de programme et enfin où les stratégies de recherche et leur variantes opérées par les instructions de contrôle sortent de ce cadre.

Jeux et programmation logique. S'il semble assez naturel d'utiliser des formalismes de jeux pour la programmation logique, cette approche est loin d'être largement développée. On peut même dire qu'elle a été étonnamment peu investiguée depuis les début des recherches en programmation logique :

- Van Emden a été le premier à remarquer les connections qui existaient entre les calculs en programmation logique et les jeux à deux joueurs utilisant l'algorithme $\alpha\beta$ [vE86].
- Cette approche a été reprise plus récemment et approfondie par Loddo et al. [CLN98, LC00, Lod02] et appliquée à la programmation logique avec contraintes.
- Pym et Ritter [PR04, PR05] ont pour leur part proposé une sémantique des jeux pour les preuves uniformes et le backtracking en reliant prouvabilité intuitionniste et prouvabilité classique (et en utilisant les preuves classiques pour stocker plusieurs tentatives de recherches de preuves intuitionnistes)
- tandis que nous avons proposé avec Miller [MS06] une approche neutre des preuves pour les preuves et les réfutations inspirée par le mécanisme de recherche de Prolog, ce qui a été étendu plus récemment par Delande et Miller [DM08].
- Tout récemment, Galanaki et al. [GRW08] ont proposé une généralisation des jeux de van Emden pour les programmes logiques avec négation bien-fondée.

Alors que la dynamique de la recherche de preuve concerne des preuves partielles, la théorie de la démonstration du calcul des séquents est une théorie de preuves complètes. Il est donc difficile de parler des échecs, du backtrack ou du fait de changer de stratégie de recherche (comme ce qui est fait avec le prédicat de coupure `cut` de Prolog) dans ce cadre. Nous proposons de considérer une autre approche qui considérera la recherche de preuve interactivement.

12.1.1 Chercher des preuves interactivement

Dans le paradigme du calcul comme recherche de preuve, un séquent $\mathcal{P} \vdash G$ correspond à l'état courant du calcul mais il s'agit également d'une manière de contraindre le calcul à effectuer. De la même manière, les restrictions sur les règles d'inférence autorisées (comme en logique linéaire) ou sur les stratégies de démonstration imposent également des contraintes à la recherche de preuve. Mais tout ceci est implicite et n'est pas fait explicitement. En particulier, il est assez difficile d'analyser ces contraintes en théorie de la démonstration elle-même quand bien même ce serait très utile puisque d'importantes primitives de programmation dépendent de telles contraintes.

L'approche interactive de la recherche de preuve que nous étudions vise précisément à rendre explicites ces contraintes sur la recherche de preuve : au lieu de construire une preuve dépendant d'un séquent donné, on considérera la construction d'une preuve qui doit passer certains tests, qui sera opposée à des tentatives de réfutation. Ces tests auront la forme de (para)preuves et seront donc construits dans le même système que celui dans lequel on cherche les preuves.

Le modèle de la recherche de preuve interactive. Nous souhaitons développer un cadre calculatoire correspondant à ce qui suit :

- Nous souhaitons chercher une preuve $\mathfrak{D}$ de $\vdash A$.
- La formule A est en fait donnée comme un ensemble de tests (les tests qui doivent être passés avec succès par la preuve que nous sommes en train de construire) : $(\mathfrak{E}_i)_{i \in I}$
- La construction de preuve procède alors par consensus avec les tests : $\mathfrak{D}$ peut être développée avec la règle R seulement si l'objet obtenu interagit bien avec tous les tests.
- Après un certain nombre d'étapes de calcul, il se peut que nous ayons un objet qui ne puisse plus être étendu. Soit la construction est terminée parce que $\mathfrak{D}$ ne peut pas réussir certains des tests, soit parce que tous les tests sont satisfaits et que plus aucune contrainte ne s'applique à $\mathfrak{D}$ et qu'il n'y a pas de raison de vouloir l'étendre plus. Dans le premier cas, on a un échec et dans le second un succès.
- Un élément intéressant avec cette approche interactive réside dans le fait que le cadre est symétrique : $\mathfrak{D}$ est testé par $\mathfrak{E}_i$ mais il peut aussi servir de test pour les $\mathfrak{E}_i$s. En particulier, les échecs peuvent être des objets intéressants et utiles. Si le résultat d'un calcul est un échec $\mathfrak{D}^{\maltese}$, on pourrait souhaiter tenter une nouvelle recherche. En effet, il se peut qu'à un certain point nous ayons choisi la mauvaise manière d'étendre $\mathfrak{D}$ et que cela cause l'échec alors qu'une autre manière de construire $\mathfrak{D}$ aurait conduit à un succès. Il y a une manière standard de bactracker, qui consiste à effacer certaines parties de $\mathfrak{D}^{\maltese}$ et à essayer une autre construction. Mais puisque nous sommes dans un cadre interactif, il y a une autre option qui consiste à utiliser $\mathfrak{D}^{\maltese}$ de manière à produire plus de tests : les $\mathfrak{E}_j^{\mathfrak{D}^{\maltese}}$ imposeront de nouvelles contraintes à la recherche d'une preuve qui devrait donc être différente de l'échec $\mathfrak{D}^{\maltese}$.

On utilisera l'abbréviation *IPS* pour désigner la recherche de preuve interactive, pour l'anglais *interactive proof search*.

12.1.2 Rechercher des preuves interactivement, dans MALL

Ajouter de nouvelles preuves : $MALL\maltese$. Si nous voulons rechercher des preuves par interaction dans MALL, nous devons élargir l'ensemble des objets de preuves que nous aurons à disposition puisque nous n'avons jamais à la fois une preuve de A et de $A^{\perp}$ dans MALL. Nous allons donc considérer des preuves dans $MALL\maltese$, c'est-à-dire des preuves

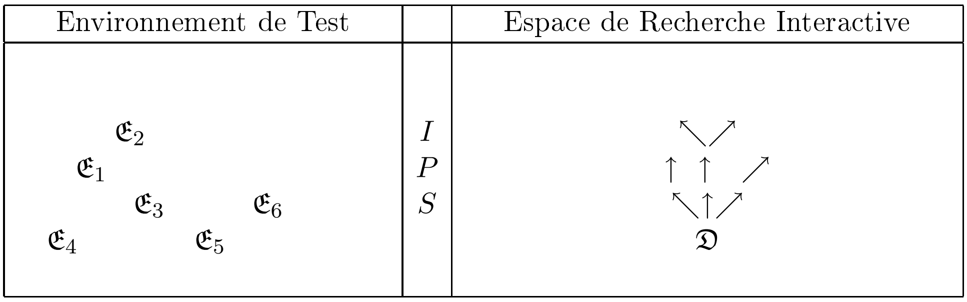


FIG. 12.1 –

---

construites dans MALL sans atomes auquel on a ajouté la règle $\maltese$, qui peut s'appliquer à tout séquent $\vdash \Gamma$ qui ne contient aucune formule négative :

$$\frac{}{\vdash \Gamma}\maltese$$

En outre, de manière à distinguer plus facilement les occurrences de formules, nous allons attribuer des indices aux formules intermédiaires, anticipant ainsi sur l'introduction des lieux que nous utiliserons lorsque nous travaillerons directement en ludique.

Un exemple de recherche interactive dans MALL$\maltese$. Soient $\mathcal{D}_0$ et $\mathcal{D}_1$ les deux parapreuves (ou tests) du séquent $\vdash \mathbf{1}_0 \mathbin{\&} (\bot_{10} \oplus_1 \bot_{11})$ données ci-dessous :

$$\mathcal{D}_i = \cfrac{\cfrac{}{\vdash \mathbf{1}_0}[\mathbf{1}] \qquad \cfrac{\cfrac{\cfrac{}{\vdash}\maltese}{\vdash \bot_{1i}}[\bot]}{\vdash \bot_{10} \oplus_1 \bot_{11}}[\oplus_i]}{\vdash \mathbf{1}_0 \mathbin{\&} (\bot_{10} \oplus_1 \bot_{11})}[\&] \qquad \text{avec } i \in \{0,1\}$$

Supposons maintenant que l'on cherche à construire une parapreuve $\mathcal{D}$ qui passerait les deux tests $\mathcal{D}_0$ et $\mathcal{D}_1$, c'est-à-dire que $\mathcal{D}$ devrait être une preuve de $MALL\maltese$ du séquent $\vdash \bot_0 \oplus (\mathbf{1}_{10} \mathbin{\&_1} \mathbf{1}_{11})$ telle que, en construisant les parapreuves $\frac{\mathcal{D}_i \quad \mathcal{D}}{\vdash}\,cut$ ($i \in \{0,1\}$) en coupant $\mathcal{D}$ avec les $\mathcal{D}_0$ et $\mathcal{D}_1$ respectivement, l'élimination de coupure normalise :

$$\boxed{\cfrac{\cfrac{\mathcal{D}_i}{\vdash \mathbf{1}_0 \mathbin{\&} (\bot_{10} \oplus_1 \bot_{11})} \qquad \cfrac{\mathcal{D}}{\vdash \bot_0 \oplus (\mathbf{1}_{10} \mathbin{\&_1} \mathbf{1}_{11})}}{\vdash}\,cut \quad \rightsquigarrow_{cut-elim} \quad \cfrac{}{\vdash}\maltese}$$

La réduction de coupures de $\mathcal{E}_0$ et $\mathcal{E}_1$ imposera des contraintes sur $\mathcal{D}$ qui peuvent être utilisées comme guide pour chercher une parapreuve de $\vdash \bot_0 \oplus (\mathbf{1}_{10} \mathbin{\&_1} \mathbf{1}_{11})$.

On peut ainsi partir d'une preuve $\mathcal{D}$ non spécifiée, la couper avec $\mathcal{D}_0$ et $\mathcal{D}_1$ et étendre la preuve $\mathcal{D}$ en fonction de possibilités d'interaction offertes par $\mathcal{D}_0$ et $\mathcal{D}_1$. Dans le cas particulier que nous considérons, nous pouvons par exemple obtenir par interaction les deux MALL$\maltese$-preuves :

$$\mathcal{D} = \cfrac{\cfrac{\cfrac{}{\vdash \mathbf{1}_{10}}[\mathbf{1}]^{\star_0} \qquad \cfrac{}{\vdash \mathbf{1}_{11}}[\mathbf{1}]^{\star_1}}{\vdash \mathbf{1}_{10} \mathbin{\&_1} \mathbf{1}_{11}}[\&]}{\vdash \bot_0 \oplus (\mathbf{1}_{10} \mathbin{\&_1} \mathbf{1}_{11})}[\oplus_1]$$

ou avec

$$\mathcal{D}' = \dfrac{\dfrac{\dfrac{}{\vdash}\,\maltese}{\vdash \bot_0}\,[\bot]}{\vdash \bot_0 \oplus (\mathbf{1}_{10} \mathbin{\&_1} \mathbf{1}_{11})}\,[\oplus_0]$$

Les deux MALL$\maltese$-preuves sont justifiées comme suit :

- Dans le cas de $\mathcal{D}$, la preuve en construction, en utilisant la règle $[\oplus_1]$, a poursuivi l'interaction dans les branches prémisses de droite des règles $[\&]$ de $\mathcal{D}_0$ et $\mathcal{D}_1$, puis, pour pouvoir interagir à la fois avec la règle $[\oplus_0]$ de $\mathcal{D}_0$ et la règle $[\oplus_1]$ de $\mathcal{D}_1$, la recherche interactive doit ajouter une règle $[\&]$ au-dessus de $[\oplus_1]$. À ce point, la prémisse de gauche de $[\&]$ dans $\mathcal{D}$ interagit avec $\mathcal{D}_0$ tandis que la prémisse de droite interagit avec $\mathcal{D}_1$: la branche qui se termine en $\star_i$ a été construite en interagissant avec $\mathcal{D}_i$, chaque règle $[\mathbf{1}]$ correspondant à l'interaction avec l'une des deux règles $[\bot]$. Comme en ludique, la normalisation se termine lorsqu'une règle $\maltese$ est rencontrée. Dans notre modèle, cela s'interprète comme le fait que les tests $\mathcal{D}_0$ et $\mathcal{D}_1$ mettent fin à leur test, en abandonnant la partie, et donc en n'imposant plus de contrainte à $\mathcal{D}$.
- La construction de $\mathcal{D}'$ a débuté en utilisant la règle $[\oplus_0]$, c'est donc avec les prémisses de gauche de la règle $[\&]$ de $\mathcal{D}_0$ et $\mathcal{D}_1$ que la parapreuve en construction devra interagir. Placée face à une règle $[\mathbf{1}]$, la preuve en construction $\mathcal{D}'$ utilise une règle $[\bot]$, mais ce faisant, elle se retrouve avec un séquent vide pour lequel seule la règle $\maltese$ est disponible. $\mathcal{D}'$ utilise donc $\maltese$ et cette construction interactive aboutit donc à un échec. Ainsi, dans la construction de $\mathcal{D}'$ les tests $\mathcal{D}_i$ ont pu détecter l'«erreur» que contenait $\mathcal{D}'$ (le fait qu'elle utilisait le daimon) sans pour autant révéler à $\mathcal{D}'$ qu'elles mêmes utilisaient un daimon.

Pas de recherche sans interaction. Une autre possibilité aurait été qu'à la place des règles $[\mathbf{1}]$, $\mathcal{D}_0$ et $\mathcal{D}_1$ aient utilisé une règle $\maltese$:

$$\mathcal{D}_i^{\maltese} = \dfrac{\dfrac{}{\vdash \mathbf{1}_0}\,\maltese \qquad \dfrac{\dfrac{\dfrac{}{\vdash}\,\maltese}{\vdash \bot_{1i}}\,[\bot]}{\vdash \bot_{10} \oplus_1 \bot_{11}}\,[\oplus_i]}{\vdash \mathbf{1}_0 \mathbin{\&} (\bot_{10} \oplus_1 \bot_{11})}\,[\&] \qquad \text{avec } i \in \{0,1\}$$

Dans ce cas, elles auraient abandonné l'interaction avant de découvrir que $\mathcal{D}'$ ne pouvait pas progresser plus loin, et l'interaction se serait donc terminée sur une preuve $\mathcal{D}''$ de la forme :

$$\mathcal{D}'' = \dfrac{\dfrac{}{\vdash \bot_0}\,\left[\bot_{|\emptyset}\right]}{\vdash \bot_0 \oplus (\mathbf{1}_{10} \mathbin{\&_1} \mathbf{1}_{11})}\,[\oplus_0]$$

(la notation $\left[\bot_{|\emptyset}\right]$ sera expliquée plus tard).

On remarque qu'au cours de la construction de $\mathcal{D}$ et de $\mathcal{D}'$, on n'a jamais utilisé la structure du séquent pour choisir les règles à développer ; d'ailleurs la recherche menant à $\mathcal{D}''$ s'interrompt alors qu'il y a encore des séquents non clos parce que plus aucune contrainte ne s'applique à la construction en cours.

L'utilisation de la règle $[\&]$ dans les tests $\mathcal{D}_i$ laisse le choix, lors de la construction interactive, entre l'interaction avec la prémisse de gauche ou de droite, c'est-à-dire le choix entre les deux règles $[\oplus_0]$ et $[\oplus_1]$.

On aurait également pu se trouver dans le cas suivant :

$$\mathcal{E}_0 = \dfrac{\dfrac{}{\vdash \mathbf{1}_0}\,[\mathbf{1}] \qquad \dfrac{\dfrac{\dfrac{}{\vdash}\,\maltese}{\vdash \bot_{1i}}\,[\bot]}{\vdash \bot_{10} \oplus_1 \bot_{11}}\,[\oplus_0]}{\vdash \mathbf{1}_0 \,\&\, (\bot_{10} \oplus_1 \bot_{11})}\,[\&] \qquad\qquad \mathcal{E}_1 = \dfrac{\dfrac{}{\vdash \mathbf{1}_0}\,[\mathbf{1}] \qquad \dfrac{}{\vdash \bot_{10} \oplus_1 \bot_{11}}\,\maltese}{\vdash \mathbf{1}_0 \,\&\, (\bot_{10} \oplus_1 \bot_{11})}\,[\&]$$

Dans ce cas, si la construction interactive développe l'interaction avec les prémisses de droite du $[\&]$, le test $\mathcal{E}_0$ contraint de la même manière qu'auparavant la construction interactive tandis que $\mathcal{E}_1$ abandonne le test de manière prématurée. La parapreuve construite interactivement est alors de la forme :

$$\mathcal{E} = \dfrac{\dfrac{\dfrac{}{\vdash \mathbf{1}_{10}}\,[\mathbf{1}]^{\star 0}}{\vdash \mathbf{1}_{10} \,\&_1\, \mathbf{1}_{11}}\,[\&_{|0}]}{\vdash \bot_0 \oplus (\mathbf{1}_{10} \,\&_1\, \mathbf{1}_{11})}\,[\oplus_1]$$

où seule la prémisse de gauche du $[\&]$ a été construite, l'autre n'ayant pas de raison de se développer puisque ne se voyant pas appliquer de contrainte.

On constate ainsi que la preuve $\mathcal{E}$ construite interactivement utilise une règle d'inférence qui n'était pas dans le système MALL$\maltese$, la règle $\left[\&_{|0}\right]$, de même qu'un peu plus haut nous avons rencontré l'inférence $\left[\bot_{|\emptyset}\right]$.

Puisque nous cherchons à construire un système symétrique où la construction des preuves est guidée par des objets de même nature que les preuves elles-mêmes, nous devrions autoriser les tests à utiliser ce genre d'inférences.

Au delà de MALL$\maltese$. On pourrait donc ajouter les parapreuves $\mathcal{D}_2$ et $\mathcal{D}_3$ suivantes qui n'utilisent que des règles d'inférences partielles pour $[\&]$ à l'ensemble des tests :

$$\mathcal{D}_2 = \dfrac{\dfrac{}{\vdash \mathbf{1}_0}\,\maltese}{\vdash \mathbf{1}_0 \,\&\, (\bot_{10} \oplus_1 \bot_{11})}\,\left[\&_{|0}\right] \qquad\qquad \mathcal{D}_3 = \dfrac{\dfrac{}{\vdash \bot_{10} \oplus \bot_{11}}\,\maltese}{\vdash \mathbf{1}_0 \,\&\, (\bot_{10} \oplus_1 \bot_{11})}\,\left[\&_{|1}\right]$$

On constate que l'ajout de ces tests fait apparaître de nouveaux comportements de recherche interactive :

- si $\mathcal{D}_2$ fait partie de l'environnement de normalisation alors $\mathcal{D}$ ***est forcée*** d'utiliser $[\oplus_0]$ comme première règle car la règle $[\oplus_1]$ ne permettrait pas à la réduction de coupure de progresser ;
- alors que si on fait une recherche de preuve interactive avec $\mathcal{D}_3$, cela interdirait la recherche qui avait conduit à un échec en forçant la sélection de la règle $[\oplus_1]$.

$\mathcal{D}_2$ et $\mathcal{D}_3$ peuvent donc être utilisées pour interdire certaines constructions interactives. Imaginons par exemple que nous ayons construit la parapreuve $\mathcal{D}'$ au cours d'une première recherche, recherche interactive qui nous a donc conduit à un échec via l'utilisation de la règle $\maltese$. Si nous ajoutons à l'ensemble de tests, consitué de $\mathcal{D}_0$ et $\mathcal{D}_1$, le nouveau test $\mathcal{D}_3$, alors nous sommes certains qu'une nouvelle recherche évitera l'exploration de la branche de gauche du $[\&]$ puisque le test $\mathcal{D}_3$ est incapable d'interagir selon cette branche. Ceci est la base de l'idée du traitement interactif du backtracking.

Vers la ludique. Cette brève étude nous a montré qu'il y a de nombreuses possibilités de guider (ou contraindre) interactivement la recherche de preuve.

Pourtant, il est nécessaire d'affaiblir certains principes logiques (qui, d'habitude, nous assurent de la cohérence du raisonnement logique et sont précisément les raisons pour lesquelles il est souhaitable d'utiliser la logique). Par exemple, il est nécessaire d'ajouter le

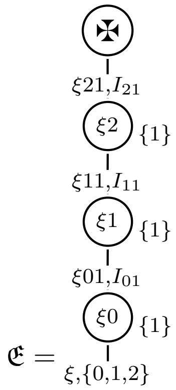


FIG. 12.2 – Le dessein $\mathfrak{E}$.

daimon ✠ qui permet de prouver n'importe quel séquent, mais il est également important d'admettre des règles d'inférence «partielles» (comme dans $\mathcal{D}_2$ et $\mathcal{D}_3$), règle dont le statut n'est pas clair ; il faut également reconsidérer d'autres principes de la logique linéaire (comme la limitation de l'affaiblissement par exemple).

Ces éléments constituent certaines des raisons pour lesquelles nous nous tournons maintenant vers la ludique pour développer plus formellement cette théorie de la recherche de preuve interactive puisque la ludique fournit une bonne théorie de l'interaction ainsi que la possibilité de considérer simplement l'ensemble des éléments déjà évoqués. En outre, le niveau d'abstraction de la ludique est intéressant pour développer le type de modèle qui nous intéresse et les règles partielles qui n'ont pas de statut clair dans MALL✠ ne poseront aucun problème en ludique.

12.2. Idée de l'algorithme de recherche de preuve interactive

Avant d'introduire la machine abstraite pour la recherche de preuve interactive en ludique, nous expliquons comment fonctionne la recherche de preuve interactive sur un exemple simple de manière à montrer les principales structures mises en œuvre dans la recherche. On considère une recherche interactive guidée par un environnement de test très simple constitué d'un seul dessein présenté en figure 12.2, la procédure décrite ci-dessous est illustrée par la figure 12.3.

On se donne donc l'environnement constitué du dessein $\{\mathfrak{E}\}$ que l'on utilise pour construire un dessein $\mathfrak{D}$.

0. Pour commencer, $\mathfrak{D}_0$ est vide et nous n'avons visité que le chemin vide :

$$Path_0 = \epsilon$$

1. $\mathfrak{E}$ est un dessein négatif, il s'agit donc d'une forêt : le dessein pourrait commencer par plusieurs actions négatives de foyer ξ, et il nous faudrait choisir de suivre l'une d'entre elles au cours du processus de normalisation. Ses actions initiales sont dans $\mathrm{Init}_{\mathfrak{E}} = \{(\xi, \{0,1,2\})^-\}$. On choisit donc une action κ_1^- dans $\mathrm{Init}_{\mathfrak{E}}$ et on ajoute $\kappa_{1\nu}$ au chemin de normalisation et κ_1^+ devient la première action de $\mathfrak{D}$:

$$Path_1 = \langle(\xi, \{0,1,2\})\rangle = \langle\kappa_{1\nu}\rangle$$

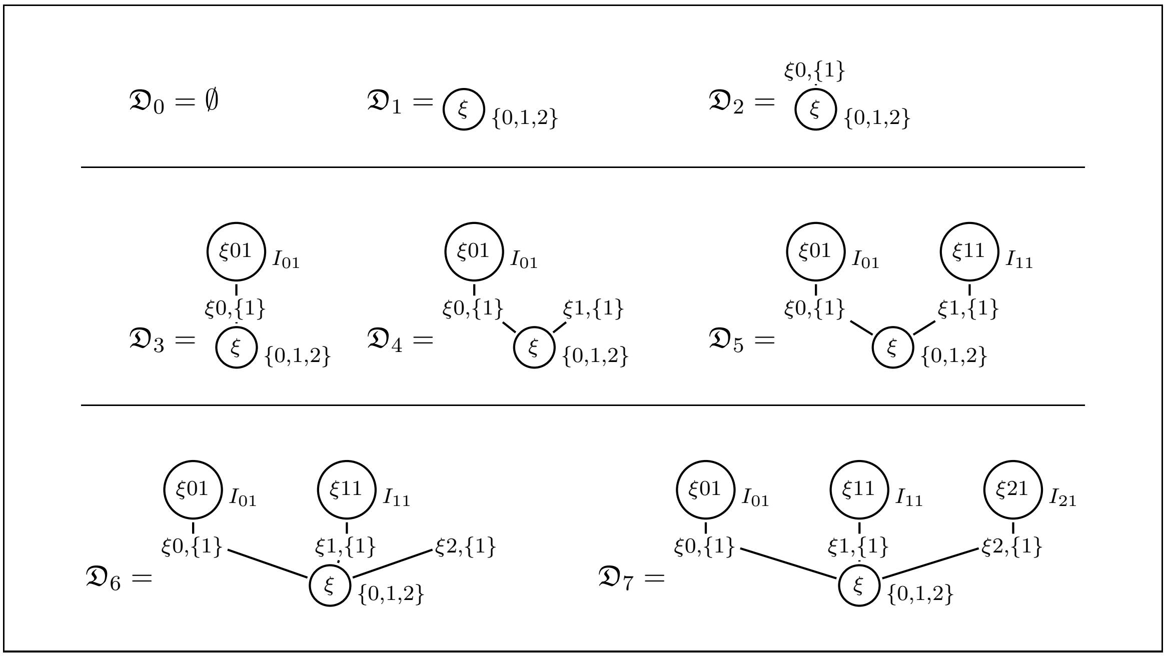


FIG. 12.3 – Recherche Interactive pour $\mathfrak{D}$.

2. Le dessein $\mathfrak{D}$ en construction pourrait avoir plusieurs actions négatives au dessus de κ_1^+ mais à ce point, la normalisation ne suivrait qu'une seule action qui correspondrait à l'action positive après κ_1^- dans $\mathfrak{E}$. Cette action est $\kappa_2^+ = (\xi 0, \{1\})^+$ et donc :

$$Path_2 = \langle \kappa_{1\nu}, \kappa_{2\nu} \rangle$$

3. Dans $\mathfrak{E}$, κ_2^+ est suivie par les actions dans $\{(\xi 01, I_{01})^-\}$, on choisit κ_3^- dans cet ensemble et on étend $Path_2$ avec $\kappa_{3\nu}$ et $\mathfrak{D}$ avec κ_3^+ :

$$Path_3 = \langle \kappa_{1\nu}, \kappa_{2\nu}, \kappa_{3\nu} \rangle$$

4. Dans $\mathfrak{E}$, κ_3^- n'est suivi que de l'action $\kappa_4^+ = (\xi 1, \{1\})^+$ et donc $Path_3$ sera étendue par $\kappa_{4\nu}$. $\mathfrak{D}_4$ doit contenir κ_4^- de manière à interagir correctement avec l'environnement, mais cette action négative doit être placée juste avant l'action positive qui la justifie. La branche conduisant à κ_4^- dans $\mathfrak{D}$ est donnée par : $\ulcorner \kappa_{1\nu}, \kappa_{2\nu}, \kappa_{3\nu}, \kappa_{4\nu} \urcorner^- = \kappa_1^+, \kappa_4^-$ et donc :

$$Path_4 = \langle \kappa_{1\nu}, \kappa_{2\nu}, \kappa_{3\nu}, \kappa_{4\nu} \rangle$$

5. Dans $\mathfrak{E}$, $Succ(\kappa_4^+) = \{(\xi 11, I_{11})^-\}$. On ajoute alors l'action κ_5^+ à $\mathfrak{D}_5$ et on étend alors la dispute comme suit :

$$Path_5 = \langle \kappa_{1\nu}, \ldots, \kappa_{5\nu} \rangle$$

6. ...

8. Dans $\mathfrak{E}$, κ_7^- est suivie par une action unique, $\kappa_8^+ = \maltese$. La normalisation se termine avec $\mathfrak{E}$ en utilisant $\maltese$ et la dispute finale est :

$$[\mathfrak{D} \rightleftharpoons \mathfrak{E}] = \langle \kappa_{1\nu}, \ldots, \kappa_{7\nu}, \maltese \rangle$$

$\rightarrow$ Après le processus d'IPS, on a construit un dessein $\mathfrak{D}$ sur la base $\vdash \xi$ tel que $[\![\mathfrak{D}, \mathfrak{E}]\!] = \maltese$, le daimon étant causé par $\mathfrak{E}$.

Cet exemple avait pour but d'illustrer les mécanismes de base que nous rencontrerons en faisant de la recherche de preuve interactive. L'exemple que nous avons choisi était bien sûr volontairement simplificateur et certains phénomènes qui ne se sont pas présentés dans l'exemple précédent devront être traité par la suite :

- Le dessein $\mathfrak{E}$ n'avait pas de branchement. Cela implique plusieurs choses :
  1. l'ensemble *Init* des actions disponibles contenait toujours exactement une action. Dans le cas général, on pourrait avoir à choisir entre plusieurs actions ;
  2. s'il y avait branchement, ce branchement pourrait être de deux types : additif ou multiplicatif. Plus précisément, au niveau des chroniques on peut avoir un branchement pour deux raisons : soit parce que l'action positive est suivie de plusieurs desseins négatifs (c'est le branchement multiplicatif du $\otimes$ de LL), soit parce que le dessein négatif qui suit l'action positive est une forêt, qu'il commence par plusieurs actions négatives (c'est le branchement additif du $\&$ de LL). On peut bien sûr avoir une combinaison des deux...

     Ces deux cas de branchement devront être traités de manières distinctes, notamment du point de vue des continuations de l'interaction ;
  3. dans l'exemple, les actions du dessein en construction sont systématiquement justifiées par l'action qui les précède immédiatement. Ceci est également dû au fait que *Init* est un singleton. Dans le cas général, on n'aura pas forcément ce phénomène ;
  4. Il se pourrait également que *Init* soit vide...
- L'environnement de tests n'est constitué que d'un seul dessein $\mathfrak{E}$. Nous devrons traiter le cas général où plusieurs tests contraignent l'interaction et où il y a des synchronisations entre les tests ;
- Suivre le déplacement de l'interaction dans $\mathfrak{E}$ était facile dans cet exemple, mais il faudra faire particulièrement attention à ce point quand on traitera la machine abstraite, en particulier au sujet de la synchronisation des tests.
- Lorsque plusieurs actions sont disponibles, pour poursuivre l'interaction, il faut choisir parmi ces actions. Il y a ici une question de non-déterminisme qu'il faudra traiter, mais par ailleurs, il pourrait arriver que parmi toutes les actions disponibles dans *Init* (du fait du branchement positif, mais peut-être également du fait de branchements antérieurs qui ont laissé des actions disponibles), certaines actions ne soient en fait pas utilisables à ce point parce que la chronique que l'on doit étendre ne justifie pas ces actions. Il faudra donc effectuer un test de justification en sélectionnant les possibilités d'extension d'une chronique.

Nous introduisons maintenant les définitions formelles qui sont nécessaires pour donner une définition précise du processus de recherche de preuve interactive.

12.3. La SLAM, Searching LAM

On va considérer dans cette section des machines abstraites pour la recherche de preuve interactive que nous appelerons SLAM-1 ou SLAM-n selon le cas, la Searching Loci Abstract Machine.

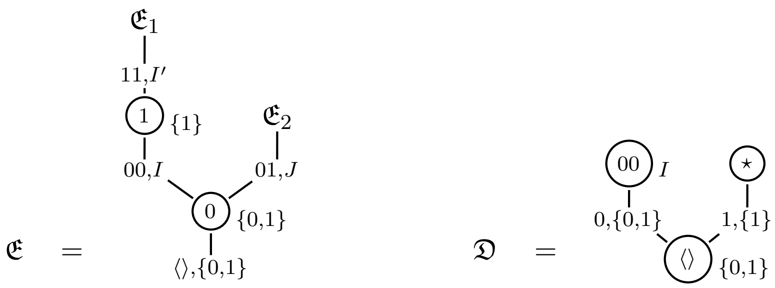


FIG. 12.4 –

12.3.1 La SLAM-1

Nous commençons par introduire une machine abstraite pour la recherche interactive de desseins dans le cas restreint de la section précédente, c'est-à-dire quand l'environnement de tests initial n'est constitué que d'un seul dessein $\mathfrak{E}$.

Définition 12.1 (*États de la SLAM-1*)

Les **états** *de la SLAM-1 sont des triplets $\langle p \bullet \mathbf{E} \mid \mathfrak{D}\rangle$ d'une suite d'actions neutres p, d'un ensemble de desseins $\mathbf{E}$ (l'environnement de tests courant) contenant au plus un dessein positif et un ensemble de chroniques $\mathfrak{D}$ (le dessein en construction, pour lequel p est un chemin de visite).*

Un **état initial** *est de la forme $\langle \epsilon \bullet \{\mathfrak{E}\} \mid \emptyset\rangle$.*

Un **état final** *de la forme $\langle p \bullet \emptyset \mid \mathfrak{D}\rangle$.*

Un état de la SLAM-1 représente un point d'une interaction de $\mathfrak{E}$ avec le dessein $\mathfrak{D}$ en construction : la dispute p a déjà été visitée par l'interaction et les desseins de $\mathbf{E}$ sont les résidus de $\mathfrak{E}$ à ce point de l'interaction. Si $\mathfrak{R}$ est le réseau de coupure atteint à partir de $(\mathfrak{D}, \mathfrak{E})$ après la dispute p, alors les desseins de $\mathbf{E}$ correspondent à la portion du réseau $\mathfrak{R}$ qui est constituée de sous-desseins de $\mathfrak{E}$.

Une transition de la machine va donc consister à :
- poursuivre la dispute par une action κ ;
- mettre à jour l'environnement $\mathbf{E}$ après le coup κ ;
- ajouter l'extension de $\mathfrak{D}$ qui aura été calculée interactivement.

Une question qui se pose immédiatement est de savoir où placer l'action que l'on va ajouter. Comme on l'a déjà expliqué, c'est la vue qui nous indiquera où placer cette action :
- dans le cas de l'ajout d'une action négative, il n'y a pas de problème puisque la vue déterminera quelle action précédemment visitée justifie l'action que l'on souhaite ajouter et placer l'action au bon endroit ;
- dans le cas de l'ajout d'une action positive, en revanche, les choses doivent être traitées avec un peu plus d'attention. En effet, $\ulcorner p \cdot \kappa \urcorner^{+} = \ulcorner p \urcorner^{-} \cdot \kappa^{+}$ et on n'a en soi aucune assurance que l'action κ est bien justifiée dans $\ulcorner p \urcorner^{-}$.

  Il est d'ailleurs facile de trouver des exemples où ce n'est pas le cas : en considérant les desseins de la figure 12.4, le dessein $\mathfrak{E}$ est un dessein de test grâce auquel on construit interactivement le dessein $\mathfrak{D}$. On doit étendre $\mathfrak{D}$ au niveau de la branche notée $\star$ et pour cela, on a deux actions disponibles : soit l'action $(11, I')^{-}$, soit l'action $(01, J)^{-}$ (qui était déjà disponible plus tôt). Pourtant, parmi ces deux actions, une seule, $(11, I')^{-}$, serait justifiée si on la plaçait au dessus de $(1, \{1\})^{-}$.

La remarque précédente nous conduit aux deux définitions suivantes :

Définition 12.2 (*Init*(E))

Soit $\mathbf{E}$ *une famille de desseins négatifs :* $\mathbf{E} = (\mathfrak{D}_i)_{i\in I}$ *où chaque* $\mathfrak{D}_i$ *est de la forme :*

$$\mathfrak{D}_i = \{\kappa_j^{i\,-} \cdot \mathfrak{D}_j^i, j \in J_i\}$$

$Init(\mathbf{E})$ *est l'ensemble d'actions négatives défini comme :*

$$Init(\mathbf{E}) = \{\kappa_j^{i\,-} \;\; tq \;\; i \in I, j \in J_i\}$$

Définition 12.3 ($\Im(p, Init, \mathfrak{D})$)

Soit p *une suite d'actions neutres et* $Init$ *un ensemble d'actions négatives et* $\mathfrak{D}$ *un ensemble de chroniques. On pose :*

$$\Im(p, Init, \mathfrak{D}) = \{\kappa \in Init \,/\, \kappa \text{ est justifiée par une action de } \ulcorner p \urcorner^{-}.\}$$

Remarque 12.4

Dans la définition précédente, la composante $\mathfrak{D}$ *n'est pas utilisée. Elle nous sera utile pour la généralisation de la machine à des environnement généraux pour laquelle nous adapterons la définition de* $\Im(p, Init, \mathfrak{D})$.

Nous donnons donc ci-dessous la définition de la machine :

Définition 12.5 (*La SLAM-1 non-déterministe*)

Étant donné un dessein atomique $\mathfrak{E}$*, la machine est définie comme suit :*

- *État initial :* $\langle \epsilon \;\bullet\; \{\mathfrak{E}\} \;\mid\; \emptyset\rangle$
- *Transitions :* $\langle p \;\bullet\; \mathbf{E} \;\mid\; \mathfrak{D}\rangle \longrightarrow \langle p' \;\bullet\; \mathbf{E}' \;\mid\; \mathfrak{D}'\rangle$
  - *Si* $\mathbf{E}$ *contient un dessein positif* $\mathfrak{D}^+ = \kappa^+ \cdot \{\mathfrak{D}'_j, j \in J\}$ *alors si* κ^+ *est une action propre :*
    - $p' = p \cdot \kappa$
    - $\mathbf{E}' = \mathbf{E} \setminus \{\mathfrak{D}^+\} \cup \{\mathfrak{D}'_j, j \in J\}$
    - $\mathfrak{D}' = \mathfrak{D} \cup \{\ulcorner p \cdot \kappa \urcorner^{-}\}$

    sinon, $\kappa^+ = \maltese$ *et un état final,* $\langle p \cdot \maltese \;\bullet\; \emptyset \;\mid\; \mathfrak{D}\rangle$*, est atteint.*
  - *Si* $\mathbf{E}$ *contient seulement des desseins négatifs* $(\mathfrak{D}_i)_{i\in I}$*, alors soit* $Init = \Im(p, Init(\mathbf{E}), \mathfrak{D})$*. De deux choses l'une, soit* $Init \neq \emptyset$ *et on choisit un* $i \in I$ *et un* $j \in J_i$ *tels que* $\kappa_j^{i\,-} \in Init$ *et on pose :*
    - $p' = p \cdot \kappa_j^i$
    - $\mathbf{E}' = \mathbf{E} \setminus \{\mathfrak{D}_i\} \cup \{\mathfrak{D}_j^i\}$
    - $\mathfrak{D}' = \mathfrak{D} \cup \{\ulcorner p \cdot \kappa_j^i \urcorner^{+}\}$

    Sinon, $Init = \emptyset$ *et alors on ajoute la chronique* $\ulcorner p \cdot \maltese \urcorner^{+}$ *au dessein en construction, et on a atteint un état final :* $\langle p \;\bullet\; \emptyset \;\mid\; \mathfrak{D} \cup \{\ulcorner p \cdot \maltese \urcorner^{+}\}\rangle$.

Définition 12.6 (*Résultat de la SLAM-1*)

Le **résultat** *d'une exécution de la machine consiste en la troisième composante d'un état final.*

La machine que l'on vient de décrire est non-déterministe : nous avons vu dans l'exemple de MALL$\maltese$ ainsi qu'à la section précédente qu'il peut y avoir à faire des choix pendant l'IPS quand plusieurs actions négatives sont disponibles. En effet, la poursuite de la recherche interactive passe alors par le choix d'une de ces actions négatives disponibles de manière à ajouter l'action positive correspondante dans le dessein en construction.

Dans l'optique de définir une machine qui effectuerait une recherche déterministe (par exemple une stratégie de recherche en profondeur d'abord avec sélection à gauche), nous introduisons des ***fonctions de sélection*** qui paramétreront la machine abstraite donnée précédemment :

Définition 12.7 (*Fonction de sélection*)

Une fonction de sélection prend comme argument un état $\mathcal{S}$ de la machine et un ensemble d'actions négatives $Init$ et retourne un sous-ensemble de $Init$, qui n'est pas vide, à moins que $Init$ soit lui-même vide.

Une fonction de sélection Select est dite **déterministe** *quand $Select(\mathcal{S}, Init)$ est un singleton sauf si $Init = \emptyset$.*

Remarque 12.8

Quand on prend comme fonction de sélection la seconde projection, on retrouvera la machine non-déterministe précédente.

Définition 12.9 (*La SLAM-1*)

Soit Select une fonction de sélection et $\mathfrak{E}$ un dessein atomique.

La SLAM-1 est définie comme suit :

État Initial : $\langle \epsilon \bullet \{\mathfrak{E}\} \mid \emptyset \rangle$

Transitions : $\langle p \bullet \mathbf{E} \mid \mathfrak{D} \rangle \longrightarrow \langle p' \bullet \mathbf{E}' \mid \mathfrak{D}' \rangle$

- **Cas positif.** *Si* $\mathbf{E}$ *contient un dessein positif $\mathfrak{D}^+ = \kappa^+ \cdot \{\mathfrak{D}'_j, j \in J\}$ et des desseins négatifs $(\mathfrak{D}_i)_{i \in I}$.*
  *Si $\kappa^+ = \maltese$, alors on atteint l'*__état final__ $\langle p \cdot \maltese \bullet \emptyset \mid \mathfrak{D} \rangle$.
  Sinon, κ^+ est une action propre et on pose :
  (i) $p' = p \cdot \kappa$,
  (ii) $\mathbf{E}' = \mathbf{E} \setminus \{\mathfrak{D}^+\} \cup \{\mathfrak{D}'_j, j \in J\} = \{\mathfrak{D}_i, i \in I\} \cup \{\mathfrak{D}'_j, j \in J\}$ *et*
  (iii) $\mathfrak{D}' = \mathfrak{D} \cup \{\ulcorner p \cdot \kappa \urcorner^-\}$.
- **Cas négatif.** *Si* $\mathbf{E}$ *ne contient que des desseins négatifs $(\mathfrak{D}_i = \{\kappa_j^{i\,-} \cdot \mathfrak{D}_j^i, j \in J_i\})_{i \in I}$. Soit $Init = \Im(p, Init(\mathbf{E}), \mathfrak{D})$.*
  Si $Select(\langle p \bullet \mathbf{E} \mid \mathfrak{D} \rangle, Init) \neq \emptyset$, on choisit une action $\kappa \in Select(\langle p \bullet \mathbf{E} \mid \mathfrak{D} \rangle, Init)$ – éventuellement la seule action κ disponible si Select est déterministe – et on considère $\mathfrak{D}_i$ $(i \in I)$ **le** *dessein négatif dont κ est une action initiale et $\mathfrak{D}_j^i$ $(j \in J_i)$ le dessein positif immédiatement positif au-dessus de κ dans $\mathfrak{D}_i$ (ie. $\kappa = \kappa_j^i$).*
  On pose alors :
  (i) $p' = p \cdot \kappa$,
  (ii) $\mathbf{E}' = \mathbf{E} \setminus \{\mathfrak{D}_i\} \cup \{\mathfrak{D}_j^i\}$ *et*
  (iii) $\mathfrak{D}' = \mathfrak{D} \cup \{\ulcorner p \cdot \kappa \urcorner^+\}$.
  *Si $Init = \emptyset$, on ajoute la chronique $\ulcorner p \cdot \maltese \urcorner^+$ au dessein en construction et on a atteint l'*__état final__ $\langle p \bullet \emptyset \mid \mathfrak{D} \cup \{\ulcorner p \cdot \maltese \urcorner^+\} \rangle$.

Un **résultat** *de la machine consiste en la troisième composante d'un état final.*

Proposition 12.10

Dans le cas où $\mathbf{E}$ ne contient que des desseins négatifs, une fois qu'on a sélectionné une action κ, cela détermine effectivement un unique dessein négatif.

Démonstration : On vérifie que tout au long des transitions de la machine, les différents desseins qui constituent les $\mathbf{E}_i$ contiennent bien des ensembles d'actions disjointes.

On part d'un état initial où $\mathbf{E}$ est réduit à un unique dessein, la condition est ainsi vérifiée.

Lorsqu'on fait une transition, $\langle p \bullet \mathbf{E} \mid \mathfrak{D}\rangle \longrightarrow \langle p' \bullet \mathbf{E}' \mid \mathfrak{D}'\rangle$, on a par hypothèse que les desseins de $\mathbf{E}$ vérifient la condition et on en conclut que dans les deux cas de transition, la condition est préservée :

- Si $\mathbf{E}' = \mathbf{E} \setminus \{\mathfrak{D}_i\} \cup \{\mathfrak{D}^i_j\}$ (cas de la transition négative), alors on a que toute action de $\mathfrak{D}^i_j$ est présente dans $\mathfrak{D}_i$ et la condition est évidemment préservée ;
- Si $\mathbf{E}' = \{\mathfrak{D}_i, i \in I\} \cup \{\mathfrak{D}'_j, j \in J\}$ (cas de la transition positive) alors d'une part les actions des $\mathfrak{D}'_j, j \in J$ étaient toutes présentes dans $\mathfrak{D}$ et sont donc distinctes des actions des $\mathfrak{D}_i, i \in I$ et d'autre part les $\mathfrak{D}'_j, j \in J$ ont des actions deux à deux disjointes puisqu'ils sont le résultat d'un branchement multiplicatif et que leurs premières actions ont donc des foyers différents (deux adresses différentes créées par l'action κ^+ de $\mathfrak{D}^+$), ils ne peuvent donc y avoir d'actions identiques dans deux $\mathfrak{D}'_j$ et la condition est vérifiée.

□

12.3.2 Propriétés de la SLAM-1

Nous étudions quelques propriétés des exécutions de la SLAM-1. Nous considérons dans la suite une recherche de preuve interactive à partir de l'environnement de tests $\mathfrak{E}$.

Les ensembles de chroniques construits par interaction pendant une évaluation de la machine satisfont les conditions de cohérence pour les desseins présentées en définition 11.5, page 249 :

Proposition 12.11

Les résultats des exécutions de la SLAM-1 sont des desseins.

Démonstration : On vérifie qu'à tout moment d'une recherche de preuve interactive, à partir de la première transition, la troisième composante de l'état vérifie les conditions des desseins à l'exception de la condition de positivité.

- **Chroniques.** Soit κ^ϵ une action de $\mathfrak{D}$ obtenue après la dispute $p \cdot \kappa$. La chronique de κ dans $\mathfrak{D}$ est la vue de polarité ϵ, $\chi = \ulcorner p \cdot \kappa \urcorner^\epsilon$.
  La condition d'alternance de polarité est évidemment vérifiée par la définition des vues qui sont des séquences d'actions de polarité alternante. La seule occurrence de ✠ dans $\mathfrak{D}$ est lorsqu'on atteint l'état final, c'est bien sûr à l'extrémité d'une chronique. La condition de justification est assurée par la définition de l'ensemble $\Im(p, Init(\mathbf{E}), \mathfrak{D})$ et la structure des vues et le fait que le test $\mathfrak{E}$ est un dessein (en effet, si $\chi = \ulcorner p \cdot \kappa \urcorner^\epsilon$, la chronique correspondante à κ dans $\mathfrak{E}$ est $\ulcorner p \cdot \kappa \urcorner^{-\epsilon}$). Deux actions de χ ont forcément des adresses différentes puisque toutes les actions de p ont des adresses différentes (il s'agit d'une dispute de $\mathfrak{E}$).
- **Positivité.** La condition de positivité est vérifiée pour une transition sur deux, dès que la dernière action ajoutée est positive. Dans tous les cas, la condition pour obtenir un état final assure que la positivité est vérifiée puisque, quand l'état final est atteint dans le cas positif, la condition est vérifiée, et quand l'état final est atteint dans le cas négatif, on ajoute une action positive ✠ qui assure que la condition est respectée.
- **Branchement positif.** La condition de branchement positif est assurée par la structure des vues qui nous servent à ajouter des actions dans le dessein

en construction. En effet, l'ajout d'une action positive s'effectue en prenant une vue positive $\ulcorner p \cdot \kappa \urcorner^+$ tandis que l'ajout d'une action négative s'effectue en prenant une vue négative $\ulcorner p \cdot \kappa \urcorner^-$. L'action positive est ainsi ajoutée juste au-dessus de l'action négative qui l'a précédée tandis que l'action négative est placée au-dessus de la précédente action, dans p, qui la justifie. Il ne peut donc pas y avoir de branchement après une action négative (il y a une seule action positive) en revanche rien n'interdit que plusieurs actions négatives distinctes soient justifiées par la même action positive κ^+, auquel cas il y a un branchement au-dessus de κ^+.
- **Partage additif.** On a en fait plus que la condition de partage additif puisque toutes les actions de $\mathfrak{D}$ sont dans la dispute p et qu'elles ont donc toutes des foyers distincts (voir la proposition suivante).
- **Totalité.** La seule possibilité pour que le résultat soit un dessein vide est qu'aucune transition n'ait lieu (l'état initial est dans un cas où on atteint un état final) et que nous soyons dans le cas positif (l'unique dessein positif – qui est aussi l'unique dessein de $\mathbf{E}$ puisqu'il s'agit d'un état initial – est réduit à ✠), mais alors cela impose que la base du dessein en construction soit négative et la condition de totalité ne s'applique pas.

□

Le résultat d'une recherche interactive est en fait une ***tranche***. En effet, puisque le dessein $\mathfrak{D}$ est intégralement construit par interaction avec un seul dessein $\mathfrak{E}$ (c'est-à-dire que toutes les actions de $\mathfrak{D}$ sont, en version non polarisée, présentes dans la dispute p correspondant à l'interaction). Les éventuels branchements de $\mathfrak{D}$ (comme dans l'exemple de la figure 12.3) sont donc multiplicatifs et non pas additifs.

Proposition 12.12

Si $\mathfrak{D}$ est un résutlat de la SLAM-1, alors $\mathfrak{D} \in \{\mathfrak{E}\}^\perp$. En outre, $\mathfrak{D}$ est matériel dans $\{\mathfrak{E}\}^\perp$.

Démonstration : p qui est une dispute enregistrant l'interaction de $\mathfrak{D}$ avec $\mathfrak{E}$. Puisque $\langle p \bullet \emptyset \mid \mathfrak{D}\rangle$ est un état final, soit il y avait un ✠ comme unique dessein positif d'un environnement de tests $\mathbf{E}$, soit l'ensemble $Init$ des actions négatives initiales des desseins de $\mathbf{E}$ était vide et on a ajouté un ✠ dans $\mathfrak{D}$. Dans chacun de ces deux cas, la dispute p atteint le daimon d'où l'orthogonalité.

La matérialité de $\mathfrak{D}$ vient du fait que toutes les actions de $\mathfrak{D}$ sont dans p.

□

Définition 12.13 ($\mathfrak{D}_i^{✠}$)

Si $(\langle p_i \bullet \mathbf{E}_i \mid \mathfrak{D}_i\rangle)_{0\leq i\leq n}$ est une exécution de la SLAM-1, alors pour $0 < i < n$ on peut construire un dessein $\mathfrak{D}_i^{✠}$ en ajoutant un daimon si la dernière action visitée est négative ou remplacer la dernière règle visitée par un daimon s'il est positif.

Les $\mathfrak{D}_i^{✠}$ sont de plus en plus précis :

Proposition 12.14

Pour $0 < i < n$, on a :
- *$\mathfrak{D}_i^{✠}$ est une tranche,*
- *elle est matérielle dans $\mathbf{E}^\perp$ et*
- *pour $0 < i \leq j < n$, on a : $\{\mathfrak{D}_i^{✠}\}^{\perp\perp} \subseteq \{\mathfrak{D}_j^{✠}\}^{\perp\perp} \subseteq \mathbf{E}^\perp$.*

Démonstration : Cette proposition s'obtient de manière similaire aux propositions précédentes, si ce n'est que le ✠ est toujours déclenché par $\mathfrak{D}_i^{✠}$.

La dernière assertion, $\{\mathfrak{D}_i^{✠}\}^{\perp\perp} \subseteq \{\mathfrak{D}_j^{✠}\}^{\perp\perp}$ est équivalente à $\{\mathfrak{D}_j^{✠}\}^{\perp} \preccurlyeq \{\mathfrak{D}_i^{✠}\}^{\perp}$ ce dont la remarque 11.25 page 253 nous assure.

□

La procédure de recherche de preuve interactive décrite par la SLAM-1 ne produit donc que des tranches. Ce cas est fondamental et permet de simplifier l'approche, mais il s'agit tout de même d'un cas relativement restreint qui, par exemple, ne permet pas de construire des preuves avec un branchement additif et ne permet pas de traiter le backtracking interactivement.

On pourrait ainsi souhaiter travailler avec des environnements de tests plus généraux comme ceux considérés en section 12.1.2 lorsqu'on utilisait $\mathcal{D}_0$ et $\mathcal{D}_1$ pour construire les deux prémisses d'une règle «avec» ou qu'on utilisait $\mathcal{D}_2$ ou $\mathcal{D}_3$ pour éviter de visiter certaines branches. Nous allons donc maintenant lever la restriction sur la cardinalité de l'environnement initial et présenter une machine plus complexe que nous définissons dans la suite.

12.3.3 La SLAM-n

Notre double motivation en généralisant la SLAM-1 est de :

- pouvoir construire interactivement des desseins qui ne soient pas que des tranches ;
- pouvoir ajouter des éléments à l'environnement, ce qui nécessite de ne pas être limité à un environnement de cardinalité 1.

Cela nous conduit à chercher à définir une machine plus complexe que la SLAM-1. En particulier, les continuations possibles d'une interaction vont maintenant dépendre de plusieurs desseins et non plus d'un seul – nous aurons donc en conséquence besoin d'un mécanisme pour synchroniser les tests qui contribuent à la construction d'une même branche – et il peut y avoir des parties d'un environnement de tests qui contribuent à la construction d'une certaine branche d'un dessein tandis qu'une autre partie contribue à la construction d'une autre branche – il est donc nécessaire de localiser les interactions.

Nous retirons maintenant la restriction et autorisons les environnements de tests à être aussi grands qu'on le souhaite, tout en restant des ensembles finis de desseins (la généralisation au cas infinitaire ne poserait aucun problème en principe mais cela pose des problèmes vis-à-vis de l'effectivité de notre procédure).

Outre les complications de définition de la version étendue de la machine, il nous faudra également faire attention aux conditions de compatibilité entre les chroniques ajoutées lors de l'interaction pour maintenir les propriétés de la SLAM-1, ce sur quoi nous reviendrons une fois la machine définie.

Définition 12.15 (*États de la SLAM-n*)

Les états de la SLAM-n ont la forme $\langle (p_i \bullet (\mathbf{E}^{ij})_{j \in J_i})_{i \in I} \mid \mathfrak{D} \rangle$ *où*

- *les* $(p_i)_{i \in I}$ *sont des suites d'actions neutres, deux à deux incomparables ;*
- *les* $(\mathbf{E}^{ij})_{i \in I, j \in J_i}$ *sont des ensembles de desseins tels que pour chaque* $i \in I$ *soit chaque* $\mathbf{E}^{ij}$ *(avec* $j \in J_i$*) contient un dessein positif, soit tous les* $\mathbf{E}^{ij}$ *(avec* $j \in J_i$*) ne sont constitués que de desseins négatifs (ils représentent l'environnement de tests),*
- *et* $\mathfrak{D}$ *est un ensemble de chroniques (*$\mathfrak{D}$ *représente le dessein en construction, les* $(p_i)_{i \in I}$ *étant des chemins de visite de ce dessein).*

On adapte de manière naturelle les fonctions de sélection pour fonctionner avec ces nouveaux états.

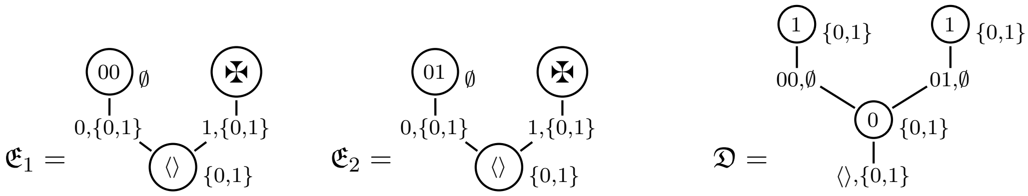


FIG. 12.5 – Desseins non-cohérents $\mathfrak{E}_1$ et $\mathfrak{E}_2$.

Définition 12.16 (*La SLAM-n*)

Soit Select une fonction de sélection et soient $(\mathfrak{E}_j)_{j\in J}$ des desseins sur une même base atomique $\xi \vdash$.

La SLAM-n est alors définie comme suit :

État Initial : $\langle(\epsilon \bullet (\{\mathfrak{E}_j\})_{j\in J}) \mid \emptyset\rangle$

Transitions : $\langle(p_i \bullet (\mathbf{E}^{ij})_{j\in J_i})_{i\in I} \mid \mathfrak{D}\rangle \longrightarrow \langle(p'_i \bullet (\mathbf{E}'^{ij})_{j\in J'_i})_{i\in I'} \mid \mathfrak{D}'\rangle$

Les transitions sont montrées dans la table 12.1.

Remarque 12.17

Il est simple de vérifier que la condition sur la structure des états est préservée par les transitions de la SLAM-n.

Remarque 12.18

Une exécution de la SLAM-n sur un état initial où le cardinal de J vaut 1 se déroule comme une exécution de la SLAM-1.

Alors qu'une exécution de la SLAM-1 ne produisait qu'une tranche, la SLAM-n est plus complexe : la SLAM-n contient un cas où la machine est bloquée.

En outre, quand les desseins de l'environnement de tests sont choisis totalement arbitrairement, et même si l'exécution de la SLAM-n n'atteint jamais un cas de blocage, l'ensemble de chroniques produit par la SLAM-n peut ne pas être un dessein comme le montre la figure 12.5 : le dessein $\mathfrak{D}$, qui est le résultat d'une recherche de preuve interactive avec $\mathfrak{E}_1$ et $\mathfrak{E}_2$ pour environnement de tests, viole la condition ***partage additif*** des desseins (encore appelée condition de ***propagation*** pour les chroniques [Gir01]).

Nous pouvons ainsi penser à considérer l'ajout d'une ***condition de cohérence*** sur l'ensemble des tests qui limiterait les tests que l'on peut considérer de manière à assurer que les résultats de la machine sont bien des desseins :

Définition 12.19 (*Tests Cohérents*)

Deux desseins atomiques $\mathfrak{E}_1$ et $\mathfrak{E}_2$ sur la même base sont dits **cohérents** *si, pour toutes chroniques $\chi_1 \in \mathfrak{E}_1$ et $\chi_2 \in \mathfrak{E}_2$, si χ_1 et χ_2 diffèrent pour la première fois sur des actions positives propres $(\zeta, I)^+$ et $(\xi, J)^+$, alors $\zeta = \xi$.*

Un environnement de tests est **cohérent** *lorsqu'il est constitué de desseins deux à deux cohérents.*

Transitions :

$$\langle (p_i \bullet (\mathbf{E}^{ij})_{j\in J_i})_{i\in I} \mid \mathfrak{D}\rangle \longrightarrow \langle (p'_i \bullet (\mathbf{E}'^{ij})_{j\in J'_i})_{i\in I'} \mid \mathfrak{D}'\rangle$$

On choisit un indice i_0 dans I tel que la dernière action action de p_{i_0} n'est pas ✠.

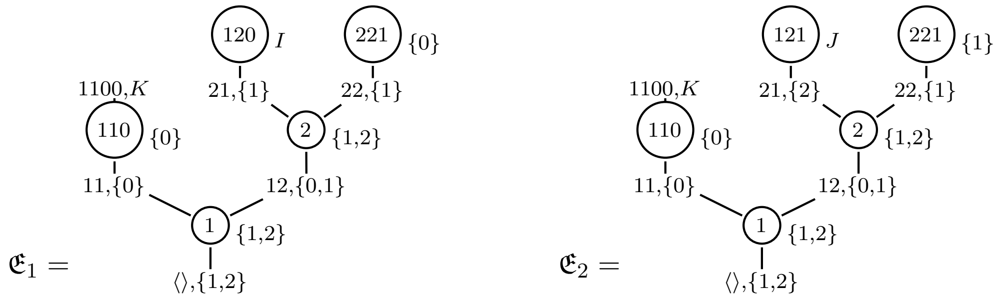
1. ***Cas positif.*** Si chaque $\mathbf{E}^{i_0j}$ contient un dessein positif $\mathfrak{D}^+_{i_0j} = \kappa^+_{i_0j} \cdot \{\mathfrak{D}'_k, k \in K_{i_0j}\}$ alors soit $J'_{i_0} = \{j \in J_{i_0}, \kappa^+_{i_0j}$ est une action propre$\}$.
   On partitionne J'_{i_0} en sous-ensembles non-vides maximaux $(J^l_{i_0})_{l\in L}$ tels que si $l \in L$, si $m, n \in J^l_{i_0}$, on a $\kappa^+_{i_0m} = \kappa^+_{i_0n}$ (c'est-à-dire que deux desseins du même ensemble $J^l_{i_0}$ ont même première action et par ailleurs si $k \neq l, m \in J^k_{i_0}, n \in J^l_{i_0}$ alors $\kappa^+_{i_0m} \neq \kappa^+_{i_0n}$).
   Soit κ'_{i_0l} l'action canoniquement associée à $J^l_{i_0}$.
   (i) Si $\exists l \in L, \ulcorner p_{i_0} \cdot \kappa'_{i_0l} \urcorner^- \in \mathfrak{D}$, la SLAM-n est ***bloquée***,
   (ii) sinon, on effectue une transition vers un nouvel état décrit comme suit :
   – $I' = I \setminus \{i_0\} \cup L$, $J'_i = J_i$ if $i \in I \setminus \{i_0\}$ et$J'_l = J_{i_0l}, l \in L$
   – $p'_i = p_i$ si $i \in I \setminus \{i_0\}$, $p'_l = p_{i_0} \cdot \kappa'_{i_0l}, l \in L$
   – $\mathbf{E}'^{ij} = \mathbf{E}^{ij}$ si $i \in I \setminus \{i_0\}, j \in J_i$
   – $\mathbf{E}'^{lj} = \mathbf{E}^{i_0j} \setminus \{\mathfrak{D}^+_{i_0j}\} \cup \{\mathfrak{D}'_k, k \in K_{i_0j}\}$ for $l \in L, j \in J_{i_0l}$
   – $\mathfrak{D}' = \mathfrak{D} \cup \{\ulcorner p_{i_0} \cdot \kappa'_{i_0l} \urcorner^-, l \in L\}$
2. ***Cas négatif.*** Si pour tout $j \in J_{i_0}$, $\mathbf{E}^{i_0j}$ contient seulement des desseins négatifs :

$$\mathbf{E}^{i_0j} = (\mathfrak{D}_{jl} = \{\kappa^{jl-}_k \cdot \mathfrak{D}^{jl}_k, k \in K_{jl}\})_{l\in L_j}.$$

   Soient $Init_j = Init(\mathbf{E}^{i_0j})$, $Init = \Im(p_{i_0}, \cap_{j\in J_{i_0}} Init_j, \mathfrak{D})$.
   (i) Si $Init \neq \emptyset$, soit κ une action de $Select(\langle (p_i \bullet (\mathbf{E}^{ij})_{j\in J_i})_{i\in I} \mid \mathfrak{D}\rangle, Init)$, et pour tout $j \in J_{i_0}$, on considère le dessein négatif $\mathfrak{D}_{jl}$ dont κ est une action initiale dans $\mathbf{E}^{i_0j}$ et $k_0 \in K_{jl}$ tel que $\mathfrak{D}^{jl}_{k_0}$ est le dessein positif immédiatement au-dessus de κ dans $\mathfrak{D}_{jl}$. Alors :
   – $I' = I$ et $J'_i = J_i, \forall i \in I'$
   – $p'_i = p_i$ pour $i \neq i_0$ et $p'_{i_0} = p_{i_0} \cdot \kappa$
   – $\mathbf{E}'^{ij} = \mathbf{E}^{ij}$ pour $i \in I', j \in J'_i, i \neq i_0$
   – $\mathbf{E}'^{i_0j} = \mathbf{E}^{i_0j} \setminus \{\mathfrak{D}_{jl}\} \cup \{\mathfrak{D}^{jl}_{k_0}\}$, $\forall j \in J_{i_0}$
   – $\mathfrak{D}' = \mathfrak{D} \cup \{\ulcorner p_{i_0} \cdot \kappa \urcorner^+\}$
   (ii) Si $Init = \emptyset$, alors nous ajoutons la chronique $\ulcorner p_{i_0} \cdot$ ✠$\urcorner^+$ au dessein en construction atteignant l'état :

$$\langle (p_i \bullet (\mathbf{E}^{ij}_E)_{j\in J_i})_{i\in I\setminus\{i_0\}} \mid \mathfrak{D} \cup \{\ulcorner p_{i_0} \cdot \text{✠} \urcorner^+\}\rangle.$$

TAB. 12.1 – Transitions de la SLAM-n.



L'interaction avec des deux desseins donne lieu, par exemple, aux disputes suivantes :

– dispute avec $\mathfrak{E}_1$: $\langle(\langle\rangle, \{1,2\}) \cdot (1, \{1,2\}) \cdot (12, \{0,1\}) \cdot (2, \{1,2\}) \cdot (22, \{1\}) \cdot (221, \{0\}) \cdot (21, \{1\}) \cdot (120, I) \cdot (11, \{0\}) \cdot (110, \{0\}) \cdot (1100, K)\rangle$
– dispute avec $\mathfrak{E}_2$: $\langle(\langle\rangle, \{1,2\}) \cdot (1, \{1,2\}) \cdot (12, \{0,1\}) \cdot (2, \{1,2\}) \cdot (22, \{1\}) \cdot (221, \{1\}) \cdot (21, \{2\}) \cdot (120, J) \cdot (11, \{0\}) \cdot (110, \{0\}) \cdot (1100, K)$

Ce qui conduit à l'arbre :

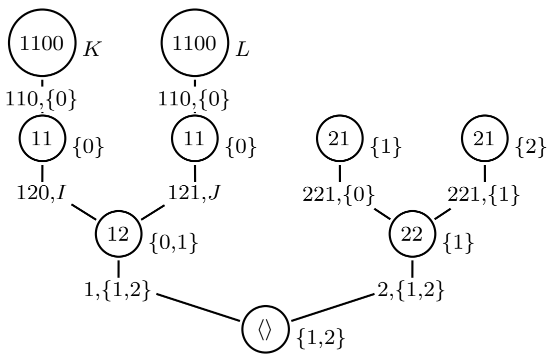


FIG. 12.6 – Tests cohérents pour lesquels la SLAM-n ne produit pas un dessein

Remarque 12.20

La condition de cohérence peut être comprise comme suit : alors que la SLAM-1 construit des tranches, le but de la SLAM-n est de construire des desseins plus généraux qui sont le résultat de la superposition de tranches. Quand plusieurs desseins de l'environnement participent à des parties distinctes du dessein, ils doivent contribuer à des tranches distinctes.

Les desseins $\mathfrak{E}_1$ et $\mathfrak{E}_2$ dans la figure 12.5 sont en effet non-cohérents à cause des actions $(00, \emptyset)^+$ et $(01, \emptyset)^+$; l'arbre $\mathfrak{D}$ obtenu par recherche interactive n'est d'ailleurs pas un dessein car on a des actions sur l'adresse 1 dans deux branches dont la première différence est une paire d'actions négatives de foyers distincts, ce qui devrait impliquer que toutes les actions qui sont au-dessus de cette différence ont des foyers distincts.

Pourtant, même avec les tests cohérents, on n'a pas toujours un dessein comme le montre l'exemple de la figure 12.6. Les deux tests $\mathfrak{E}_1$ et $\mathfrak{E}_2$ vérifient bien la condition de cohérence mais pourtant l'objet $\mathfrak{D}$ construit par interaction n'est pas un dessein. La condition de cohérence mentionnée ci-dessus ne peut en effet pas prévenir le non-partitionnement multiplicatif dès lors que l'interaction a effectué deux sauts comme dans l'exemple considéré.

On en vient donc à la solution la plus naturelle qui est non pas de restreindre les tests, mais de faire que la machine prenne dynamiquement en compte l'obligation d'assurer la condition de propagation. Cela passe par une redéfinition de $\Im(\cdot,\cdot,\cdot)$:

Définition 12.21 ($\Im(p, Init, \mathfrak{D})$)

Soit p une suite d'actions neutres, $Init$ un ensemble d'actions négatives et $\mathfrak{D}$ un ensemble de chroniques. On pose :

$$\Im(p, Init, \mathfrak{D}) = \{(\sigma, I)^- \,/ (\sigma, I)^- \textit{est justifiée dans } \ulcorner p \urcorner^- \textit{ et s'il existe } \chi \cdot (\sigma, L)^+ \in \mathfrak{D},$$

$$\textit{la première différence entre } \chi \textit{ et } \ulcorner p \urcorner^- \textit{ a lieu sur des actions négatives de même foyer}\}$$

On adapte la SLAM-n pour qu'elle utilise maintenant cette définition de $\Im(p, Init, \mathfrak{D})$ dans la sélection des actions initiales.

On peut alors prouver que les résultats des exécutions de la SLAM-n sont des desseins :

Proposition 12.22

Si $\langle(\epsilon \bullet \{(\mathfrak{E}_i)_{i\in I}\}) \mid \emptyset\rangle$ est un état initial, alors une exécution de la SLAM-n qui n'est jamais bloquée (c'est-à-dire que le cas 1.(i) n'est jamais rencontré) construit interactivement un ensemble de chroniques qui est un dessein.

Démonstration : On s'intéresse à la condition de partage additif. Supposons que deux chroniques χ_1 et χ_2 d'un résultat d'une exécution dans l'environnement $\mathbf{E}$ soient de la forme : $\chi_1 = \chi\cdot(\xi, I)^-\cdot\chi'_1$ et $\chi_2 = \chi\cdot(\zeta, J)^-\cdot\chi'_2$ et supposons que χ'_1 et χ'_2 contiennent des actions de foyer identique κ_1 et κ_2. On considère de telles actions minimales dans le sens où aucune paire d'action de χ'_1, χ'_2 précédant κ_1 et κ_2 n'ont le même foyer. On en déduit que κ_1 et κ_2 sont positives. On considère alors la transition qui ajoute l'action κ_1 dans le dessein en construction interactive ($e_i \longrightarrow e_{i+1}$) et la transition qui ajoute l'action κ_2 dans le dessein en construction interactive ($e_j \longrightarrow e_{j+1}$). On a forcément $i \neq j$ et on peut supposer $i < j$. On a donc pour que $e_j \longrightarrow e_{j+1}$ soit réalisable, $\kappa_2^- \in \Im(p^j_{i_0}, Init, \mathfrak{D}_j)$ or $\chi \cdot (\xi, I)^- \cdot \chi'_1 \in \mathfrak{D}_j$ puisque $i < j$ et donc par définition $\xi = \zeta$.

□

Proposition 12.23

Un état final pour une exécution qui n'est jamais **bloquée** *(cas 1.(i) de la SLAM-n) est de la forme $\langle(p_i \cdot \maltese \bullet \emptyset)_{i\in I} \mid \mathfrak{D}\rangle$.*

Démonstration : On considère une exécution de la SLAM-n, $(e_i)_{1\leq i\leq n}$, qui ne rencontre jamais le cas 1.(i) et on suppose que $e_n = \langle(p_i \bullet (\mathbf{E}^{ij})_{j\in J_i})_{i\in I} \mid \mathfrak{D}\rangle$ est donc un état final non-bloqué. Si il existe $i_0 \in I$ tel que $J_{i_0} \neq \emptyset$, soit chaque $\mathbf{E}^{i_0 j}, j \in J_{i_0}$.

□

12.3.4 Backtracking

Dans cette section, nous expliquons comment traiter le backtracking en utilisant les environnements généralisés.

Le principe du bracktracking interactif est d'utiliser le dessein construit pour créer un nouveau test à ajouter à l'environnement de tests tel que, dans le nouvel environnement, on relance un nouveau calcul.

Considérons un état final $\mathcal{S}' = \langle (p_i \cdot \maltese \;\bullet\; \emptyset)_{i \in I} \;\mid\; \mathfrak{D} \rangle$ atteint à partir d'un état initial $\mathcal{S} = \langle (\epsilon \;\bullet\; (\{\mathfrak{E}_j\})_{j \in J}) \;\mid\; \emptyset \rangle$. Si $I \neq \emptyset$, alors $\mathfrak{D}$ est un échec (il contient un $\maltese$ à $\ulcorner p_i \cdot \maltese \urcorner^+$ pour chaque $i \in I$, c'est-à-dire que chaque $p_i, i \in I$ est une dispute conduisant à un $\maltese$ dans $\mathfrak{D}$.

On va utiliser ces chemins $p_i, i \in I$ pour enrichir l'environnement de tests avec de nouveaux desseins.

Définition 12.24 ($\mathfrak{Test}(p)$)

Le dessein $\mathfrak{Test}(p)$ est défini comme suit :
- $\mathfrak{Test}(\epsilon) = \emptyset$;
- $\mathfrak{Test}(\kappa) = \{\kappa^+\}$;
- $\mathfrak{Test}(s \cdot \kappa \cdot \kappa') = \{\ulcorner s \cdot \kappa \cdot \kappa' \urcorner^+, \ulcorner s \cdot \kappa \urcorner^-\} \cup \mathfrak{Test}(s)$.

Proposition 12.25

$\mathfrak{Test}(p_i), i \in I$ est une tranche.
En outre, $\mathfrak{Test}(p_i)$ est le plus petit dessein (comme ensemble de chroniques ordonné par l'inclusion) réalisant l'interaction $p_i \cdot \maltese$ avec le dessein final $\mathfrak{D}$.

De manière à modéliser l'instruction de backtrack, on va utiliser une variante de $\mathfrak{Test}(\_)$. En effet, $\mathfrak{Test}(\_)$ contient à la fois trop de chroniques et trop peu de chroniques pour être utilisé pour le backtracking : un dessein de backtrack ne devrait pas autoriser l'interaction le long de p_i à atteindre le daimon (mais $\mathfrak{Test}(p_i)$ le permet) et il devrait être capable d'interagir avec tout autre dessein que $\mathfrak{D}$ (mais $\mathfrak{Test}(p_i)$ ne le peut pas).

Définition 12.26 ($\mathfrak{Backtrack}(p)$)

$\mathfrak{Backtrack}(p)$ est le dessein obtenu à partir de $\mathfrak{Test}(p)$ comme suit :
- *pour chaque chronique positive $\chi \in \mathfrak{Test}(p)$, $\chi \neq \ulcorner p \urcorner^+$ qui se termine sur une action propre $(\xi, I)^+$, pour tout $i \in I$ et tout $J \in \mathcal{P}_f(\omega)$, si $\chi \cdot (\xi i, J)^- \notin \mathfrak{Test}(p)$, alors on étend χ avec $(\xi i, J)^- \cdot \maltese$.*
- *on retire la chronique $\ulcorner p \urcorner^+$ ainsi que la chronique négative qu'elle étend immédiatement : si $\ulcorner p \urcorner^+ = s \cdot \kappa_1^- \cdot \kappa_2^+$, alors on retire $s \cdot \kappa_1^- \cdot \kappa_2^+$ et $s \cdot \kappa_1^-$ de $\mathfrak{Test}(p)$*

Proposition 12.27 ($\mathfrak{Backtrack}(p)$ – *définition équivalente*)

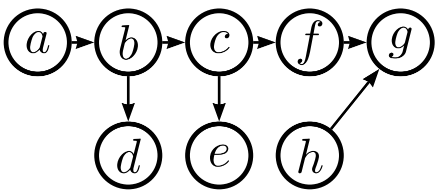
$\mathfrak{Backtrack}(p)$ est le plus petit dessein tel que :
- *$\mathfrak{Backtrack}(p)$ contient toutes les chroniques positives de $\mathfrak{Test}(p)$ excepté $\ulcorner p \urcorner^+$;*
- *si $\chi \in \mathfrak{Backtrack}(p)$ est une chronique positive se terminant sur une action propre $(\xi, I)^+$, alors pour tout $i \in I$ et $J \in \mathcal{P}_f(\omega)$ tels que $\chi \cdot (\xi i, J)^- \notin \mathfrak{Test}(p)$, on a $\chi \cdot (\xi i, J)^- \cdot \maltese \in \mathfrak{Backtrack}(p)$.*

Remarque 12.28

Dans la définition 12.27, $\mathfrak{Backtrack}(p)$ est «très» infini parce que toutes les chroniques $\chi \cdot (\xi i, J)^- \cdot \maltese$ sont ajoutées. Pourtant, si on collectait l'information au cours de la recherche (à l'étape 1. de la SLAM-n, quand on considère l'ensemble des actions initiales positives dans l'environnement), on pourrait ne retenir que les actions qui peuvent être rencontrées et construire un dessein $\mathfrak{Backtrack}(p)$ fini, pourvu que l'environnement de tests initial était constitué seulement de desseins à branchement fini.

Proposition 12.29

$\langle (\epsilon \;\bullet\; (\{\mathfrak{E}_j\})_{j \in J} \cup (\{\mathfrak{Backtrack}(p_i)\})_{i \in I}) \;\mid\; \emptyset \rangle$ est un état initial qui ne calculera plus jamais la dispute $(p_i)_{i \in I}$.



```
adj(a, b).      adj(b, c).
adj(b, d).      adj(c, e).
adj(c, f).      adj(f, g).
adj(h, g).

p(X,Y) :-  adj(X,Z), adj(Z,Y).
```

FIG. 12.7 – Graphe

12.4. Exemples et codages

12.4.1 Un exemple concret : les chemins dans un graphe.

Soit $\mathcal{G}$ le graphe représenté en figure 12.7. Nous souhaitons implémenter la recherche de chemins de longueur 2 dans ce graphe en utilisant une recherche de preuve interactive. Cela correspond aux prédicats montrés en figure 12.7. Par exemple, $p(c, g)$ pourrait être représenté comme la formule MALL :

$$\bigoplus_{x\in\{a,\ldots,h\}} (adj(c, x) \;\&\; adj(x, g))$$

Le graphe et la relation de chemin p doivent être représentés par des contre-desseins, pour servir de tests qui guideront la recherche interactive d'un dessein $\mathfrak{D}$. L'environnement de contre-desseins pourrait être constitué des deux desseins $\mathfrak{E}_1$ et $\mathfrak{E}_2$ de la figure 12.8.

Voici comment $\mathfrak{E}_1$ et $\mathfrak{E}_2$ sont construits : on choisit un lieu pour p dans lequel on placera les desseins de l'environnement (de base $p \vdash$) et le dessein à construire par recherche de preuve interactive (de base $\vdash p$). Supposons que $a, b, \ldots, g, h$ sont des codes d'entiers représentant des nœuds du graphe de la manière habituelle (on peut choisir des entiers arbitraires). pe représentera la formule $(adj(c, e) \;\&\; adj(e, g))$ et $pe1$ et $pe2$ représenteront respectivement $adj(c, e)$ et $adj(e, g)$. $\mathfrak{E}_1$ représentera les arcs ayant leur origine en c et $\mathfrak{E}_2$ représentera les arcs ayant g comme but.

Nous sommes donc dans le cas d'un 2-environnement.

Il y a 8 choix pour la première action du dessein en construction $\mathfrak{D}$, ce qui conduit aux desseins de la figure 12.9 (8 calculs possibles) en fonction du choix de la première action (seulement l'un d'entre eux est un succès).

Voici une brève explication du dialogue :

- L'action $(p, \{x\})^+$ correspond à affirmer qu'il y a un chemin de c à g passant par x ;
- l'ensemble d'actions $\{(p, \{x\})^-\}_{x\in\{a,b,c,\ldots,g,h\}}$ correspond aux arguments potentiels auxquels les desseins $\mathcal{E}_1$ et $\mathcal{E}_2$ sont prêts à répondre ;
- l'action $(px, \{1\})^+$ (ou $(px, \{2\})^+$) correspond à une attaque de $\mathfrak{D}$ par les tests $\mathfrak{E}_i$ signifiant : est-ce que x est adjacent à c (respectivement à g) ;
- l'action $(px, \{1\})^-$ (ou $(px, \{2\})^-$) correspond aux attaques possibles par les $\mathfrak{E}_i$ auxquelles le dessein $\mathfrak{D}$ est prêt à répondre ;
- l'action $(px1, \emptyset)^+$ signifie : «oui ! x est adjacent à c» (respectivement, avec l'indice 2, à g). Le ✠ apparaissant au lieu de l'action précédente signifie que l'erreur dans l'argumentation de $\mathfrak{D}$ se trouve ici.

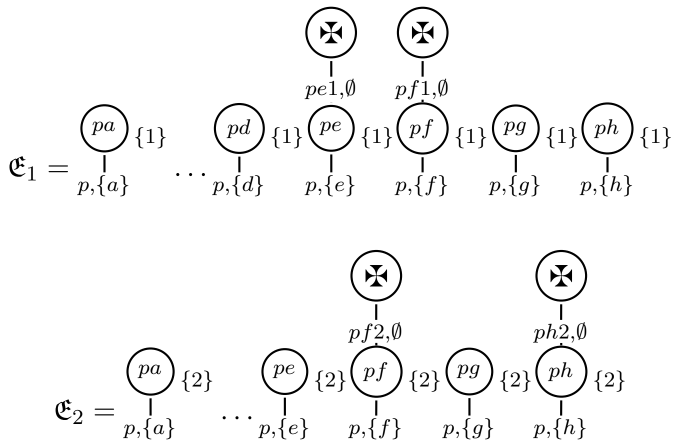
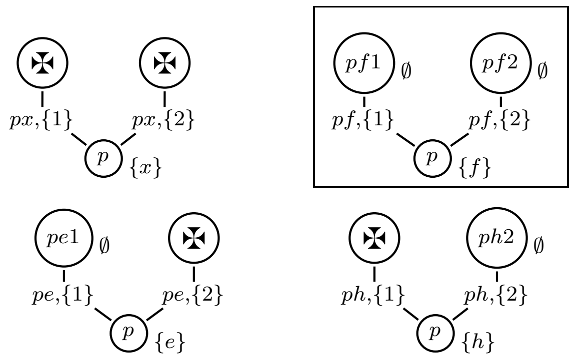


FIG. 12.8 – Les desseins $\mathfrak{E}_1$ et $\mathfrak{E}_2$.



Si $x \in \{a, b, c, d, g\}$.

FIG. 12.9 – Les résultats d'une IPS avec $\mathfrak{E}_1$ et $\mathfrak{E}_2$.

– l'action $(px1, \emptyset)^-$ est la préparation par $\mathfrak{E}_1$ à l'obtention de cet argument utilisé par $\mathfrak{D}$... et le ✠ suivant signifie que dans ce cas, le dessein $\mathfrak{E}_1$ abandonne le test, mais il se pourrait qu'un autre dessein de l'environnement continue à demander des informations, des justifications à $\mathfrak{D}$, entraînant la poursuite de la recherche et la construction d'un dessein plus précis.

12.4.2 Un codage des programmes logiques propositionnels

On considère le fragment propositionnel des clauses de Horn et les programmes logiques définis sur ce fragment :

Définition 12.30 (*Clauses de Horn propositionnelles*)

On se donne un ensemble (fini ou dénombrable) de variables propositionnelles $\mathcal{P}$; on considère deux propositions spéciales, l'échec (`fail`) et la coupure (`!`), qui ne pourront apparaître que négativement dans le programme. On considère les clauses de Horn définies sur $\mathcal{P}$ comme suit. Une clause $\mathcal{C}$ est la donnée d'un ensemble de littéraux dont au plus un est positif :

- **Faits.** *La clause ne contient aucun littéral négatif. On note cela :*

$$\mathcal{C} \triangleq p.$$

- **Règles.** *La clause contient exactement un littéral positif. On note cela :*

$$\mathcal{C} \triangleq p : -q_1, \dots q_n.$$

- **Buts.** *La clause ne contient pas de littéral positif. On note cela :*

$$\mathcal{C} \triangleq ? - q_1, \dots q_n.$$

Un programme est constitué d'un ensemble de faits et de règles.

On définit comme suit l'interprétation d'un pogramme $Prog = \{\mathcal{C}_i, 1 \leq i \leq n\}$.

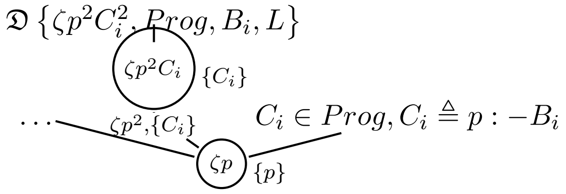
Définition 12.31 (*Énumération des clauses et des variables propositionnelles*)

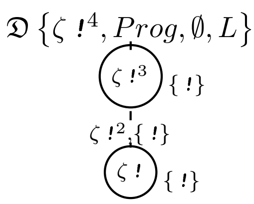
Puisqu'il existe un ensemble fini de clauses dans $Prog$ et un ensemble au plus dénombrable de variables propositionnelles, on peut sans peine définir une injection ι de $Prog \cup \{\mathcal{P}\}$ dans les naturels, que l'on notera implicitement dans la suite : quand on parle de desseins, on notera $\mathcal{C}_i$ ou p pour $\iota(\mathcal{C}_i)$ et $\iota(p)$ respectivement.

Définition 12.32 ($\mathfrak{D}[\xi, Prog, R, L]$)

Soit $Prog$ un programme logique. On définit l'interprétation de $Prog$ comme $\cup_{p\in\mathcal{P}}\mathfrak{D}[\langle\rangle, Prog, [p], []]$ où $\mathfrak{D}\Big[\xi, Prog, [p_1, \dots, p_l], \Big[n_1^{\xi_1}, \dots, n_k^{\xi_n}\Big]\Big]$ est défini comme :

- $\mathfrak{D}\left\{\xi, Prog, R, L\right\} = \begin{array}{c} \mathfrak{D}'_{\{\xi, Prog, R^{\xi} @ L\}} \\ | \\ \xi, R \end{array}$

- $\mathfrak{D}'\left\{Prog, p^{\zeta} :: L\right\} =$ 

- $\mathfrak{D}'\left\{Prog, \texttt{!}^{\zeta} :: L\right\} =$ 

- $\mathfrak{D}'\left\{Prog, \texttt{fail}^{\zeta} :: L\right\} = \zeta\texttt{fail}\ _{\{\texttt{fail}\}}$

On peut définir le bactracking comme on l'a fait précédemment, ainsi qu'un backtracking qui prenne en compte la coupure à la manière de Prolog.

Conclusion

«Je pourrais vous dire : après tout, c'étaient des pistes à suivre, peu importait où elles allaient ; il importait même que cela n'aille nulle part, en tout cas pas dans une direction déterminée à l'avance ; c'étaient comme des pointillés. À vous de les continuer ou de les infléchir ; à moi, éventuellement, de les poursuivre ou de leur donner une autre configuration. Après tout, ces fragments, on verra bien, vous ou moi, ce qu'on peut en faire.»

Michel Foucault, cours du 7 janvier 1976[1]
Il faut défendre la société, Hautes Études, Gallimard-Seuil

Les contributions de cette thèse sont de trois types :

– Nous avons proposé une solution à la question de la séparation en $\lambda\mu$-calcul. Alors que David et Py [DP01a] avait répondu par la négative à la question de savoir si le $\lambda\mu$-calcul possédait une propriété de séparation, nous avons exhibé une extension naturelle du calcul de Parigot qui possède la propriété de séparation et qui admet une interprétation de streams. Cette extension est d'ailleurs tellement naturelle que certains auteurs la considéraient dans la littérature comme syntaxe de base pour le $\lambda\mu$-calcul sans forcément mettre en évidence les différences entre les deux calculs. Le fait que le $\lambda\mu$-calcul de Parigot ne sépare pas tandis que le $\Lambda\mu$-calcul a la propriété nous amène pourtant à considérer qu'il s'agit véritablement de deux calculs différents. Nous avons ensuite développé la méta-théorie du $\Lambda\mu$-calcul et nous avons ainsi pu établir que le $\Lambda\mu$-calcul préservait la grande majorité des bonnes propriétés de $\lambda\mu$ (confluence, normalisation forte, etc). Renforçant l'interprétation de stream du calcul, nous avons également donné un système de type alternatif qui n'est pas basé sur la logique classique mais qui permet d'observer, au niveau typé, certaines des constructions essentielles à la preuve de séparation, ce qui est peut-être une première étape vers un résultat de séparation typé. Poursuivant dans cette direction, nous avons mené une analyse du $\Lambda\mu$-calcul via des réseaux, ce qui nous a permis de décomposer les réductions du calcul de manière élémentaire et de mettre à jour un calcul connexe, dans lequel la construction de stream a la primauté. Dans un dernier chapitre, nous avons tâché d'éclaircir les relations qu'entretenaient les différentes versions de $\lambda\mu$-calcul et de donner plus de poids à l'interprétation de stream, notamment

[1] Je ne résiste pas à citer également la suite de ce paragraphe du cours de Foucault : *«Je me faisais un peu l'effet d'un cachalot qui saute par-dessus la surface de l'eau, en y laissant une petite trace provisoire d'écume, et qui laisse croire, fait croire, ou veut croire, ou croit peut-être effectivement lui-même, qu'en dessous, là où on ne le voit plus, là où il n'est plus perçu ni contrôlé par personne, il suit une trajectoire profonde, cohérente et réfléchie.»*

en rapportant une proposition de sémantique dénotationnelle pour $\Lambda\mu$. Les liens que le $\Lambda\mu$-calcul entretient avec un calcul introduit par Herbelin et al., le $\lambda\mu\widehat{\mathsf{tp}}$, ainsi que la structure de la machine abstraite proposée dans le dernier chapitre tend à donner du poids au fait que le $\Lambda\mu$-calcul ait des liens avec le contrôle délimité.

- Dans une seconde partie, nous avons revisité la question de la focalisation en proposant une preuve de la focalisation pour la logique linéaire dont le but était d'être axée autour de l'idée de transformation de preuve et du souhait de mettre en évidence les invariants de la focalisation (notamment les graphes de focalisation que nous introduisons dans le chapitre 9) qui sont nécessaires pour étendre le résultat de focalisation à des extensions ou des systèmes voisins de la logique linéaire. Notre méthode des graphes de focalisation a ainsi pu être adaptée pour la logique linéaire avec points fixes étudiée par Baelde et Miller [BM07] et nous avons présenté comment traiter l'une des versions de logique linéaire à complexité bornée. Un autre produit des graphes de focalisation est la notion de multifocalisation que nous avons étudié au chapitre 10 établissant qu'il existait une notion de maximalité parmi les preuves multifocalisées, ce qui fournit un résultat de canonicité permettant de comparer les liens entre preuves maximalement multifocalisée et réseaux de démonstration.
- Nous avons enfin proposé un cadre pour la recherche de preuve interactive, ou encore recherche de preuve par élimination des coupures. Notre objectif est de fournir un système plus homogène pour l'étude de la recherche de preuve que les système habituellement utilisés. La structure générale de notre approche consiste à considérer une recherche de preuve par interaction avec un ensemble de tests : la construction de preuve n'est plus contrainte par un séquent, mais par des tests avec lesquels la preuve en construction doit interagir. Nous avons proposé une machine abstraite pour la construction interactive de desseins, la SLAM, et une manière de faire du backtracking de manière interactive, par élargissement de l'ensemble de tests. Nous avons en outre illustré notre propos de quelques exemples.

Bibliographie

[Abr03] Samson Abramsky. Sequentiality vs. concurrency in games and logic. *Mathematical Structures in Computer Science*, 13(4) :531–565, 2003.

[AC98] Roberto Amadio and Pierre-Louis Curien. *Domains and Lambda Calculi.* Cambridge tracts in theoretical computer science. Cambridge University Press, July 1998.

[AHS07] Zena M. Ariola, Hugo Herbelin, and Amr Sabry. A type-theoretic foundation of delimited continuations. *Higher-Order and Symbolic Computation*, 2007.

[AJ94] Samson Abramsky and Radha Jagadeesan. Games and full completeness for multiplicative linear logic. *Journal of Symbolic Logic*, 59(2) :543–574, 1994.

[AM99] J.-M. Andreoli and R. Maieli. Focusing and proof nets in linear and noncommutative logic. In *International Conference on Logic for Programming and Automated Reasoning (LPAR)*, volume 1581 of *LNAI*. Springer, 1999.

[And90] Jean-Marc Andreoli. *Proposition pour une synthèse des paradigmes de la programmation logique et de la programmation par objets.* Thèse de doctorat, Université Paris VI, June 1990.

[And92] Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *Journal of Logic and Computation*, 2(3) :297–347, 1992.

[AP90] J.-M. Andreoli and R. Pareschi. Linear objects : Logical processes with built-in inheritance. In *Proceeding of the Seventh International Conference on Logic Programming, Jerusalem*, May 1990.

[Bar84] Henk Barendregt. *The Lambda Calculus, its Syntax and Semantics.* Studies in Logic and the Foundations of Mathematics. Elsevier publishers, 1984.

[BBD05] Malgorzata Biernacka, Dariusz Biernacki, and Olivier Danvy. An operational foundation for delimited continuations in the cps hierarchy. *Logical Methods in Computer Science*, 1(2), 2005.

[Ber78] Gérard Berry. Stable models of typed lambda-calculi. In *International Colloquium on Automata, Languages and Programming*, pages 72–89, 1978.

[BHeF01] Kensuke Baba, Sachio Hirokawa, and Ken etsu Fujita. Parallel reduction in type free lambda/mu-calculus. *Electronic Notes in Theoretical Computer Science*, 42, 2001.

[Bie98] Gavin Bierman. A computational interpretation of the $\lambda\mu$-calculus. In *MFCS'98*, volume 1450 of *Lecture Notes in Computer Science*, pages 336–345. Springer, 1998.

[BM07] David Baelde and Dale Miller. Least and greatest fixed points in linear logic. In N. Dershowitz and A. Voronkov, editors, *International Conference on Logic for Programming and Automated Reasoning (LPAR)*, volume 4790 of *Lecture Notes in Computer Science*, pages 92–106, 2007.

[Bö68] Corrado Böhm. Alcune proprietà delle forme $\beta\eta$-normali nel λK-calcolo. *Publicazioni dell'Istituto per le Applicazioni del Calcolo*, 696, 1968.

[CG99] Roberto Di Cosmo and Stefano Guerrini. Strong normalization of proof nets modulo structural congruences. In *RtA '99 : Proceedings of the 10th International Conference on Rewriting Techniques and Applications*, pages 75–89, London, UK, 1999. Springer-Verlag.

[Chu40] Alonzo Church. A formulation of the simple theory of types. *Journal of Symbolic Logic*, 5 :56–68, 1940.

[Chu41] Alonzo Church. *The Calculi of Lambda-Conversion*. Princeton University Press, 1941.

[CK97] Roberto Di Cosmo and Delia Kesner. Strong normalization of explicit substitutions via cut elimination in proof nets. In *LICS '97 : Proceedings of the 12th Annual IEEE Symposium on Logic in Computer Science*, page 35, Washington, DC, USA, 1997. IEEE Computer Society.

[CLN98] Roberto Di Cosmo, Jean-Vincent Loddo, and Stephane Nicolet. A game semantics foundation for logic programming (extended abstract). In *PLILP/ALP*, pages 355–373, 1998.

[CMS08] Kaustuv Chaudhuri, Dale Miller, and Alexis Saurin. Canonical sequent proofs via multi-focusing. In Juhani Karhumäki and Luke Ong, editors, *IFIP International Conference on Theoretical Computer Science*, September 2008.

[Cur52] Haskell B. Curry. The permutability of rules in the classical inferential calculus. *Journal of Symbolic Logic*, 17(4) :245–248, 1952.

[Dan90] Vincent Danos. *La Logique Linéaire appliquée à l'étude de divers processus de normalisation (principalement du λ-calcul)*. Thèse de doctorat, Université Paris VII, 1990.

[DF90] Olivier Danvy and Andrzej Filinski. Abstracting control. In *LISP and Functional Programming*, pages 151–160, 1990.

[dG94] Philippe de Groote. On the relation between the $\lambda\mu$-calculus and the syntactic theory of sequential control. In *LPAR'94*, volume 822 of *Lecture Notes in Artificial Intelligence*, 1994.

[dG98] Philippe de Groote. An environment machine for the $\lambda\mu$-calculus. *Mathematical Structures in Computer Science*, 8, 1998.

[DM08] Olivier Delande and Dale Miller. A neutral approach to proof and refutation in MALL. In F. Pfenning, editor, *23rd Symposium on Logic in Computer Science*, pages 498–508. IEEE Computer Society Press, 2008.

[DNR04] René David, Karim Nour, and Christophe Raffalli. *Introduction à la logique – Théorie de la démonstration*. Sciences Sup. Dunod, 2004.

[DP00] Kosta Došen and Zoran Petrić. The maximality of the typed lambda caluclus and of cartesian closed categories. *Publications de l'Institut Mathématique*, 68(82) :1–19, 2000. Nouvelle série.

[DP01a] René David and Walter Py. $\lambda\mu$-calculus and Böhm's theorem. *Journal of Symbolic Logic*, 2001.

[DP01b] Kosta Došen and Zoran Petrić. The typed Böhm theorem. *Electronic Notes in Theoretical Computer Science*, 50(2), 2001. in Proceedings of BOTH 2001.

[dR94] Simona Ronchi della Rocca. Basic lambda-calculus. Notes de cours de Advanced School on Typed Lambda Calculus and Functional Programming, September 1994.

[DY99] Olivier Danvy and Zhe Yang. An operational investigation of the CPS hierarchy. In *Eightennth European Symposiuim on programming*, Lecture Notes in Computer Science, pages 224–242. Springer, 1999.

[Fag02] Claudia Faggian. Travelling on designs. In *Computer Science Logic*, pages 427–441, 2002.

[FH92] Matthias Felleisen and Robert Hieb. The revised report on the syntactic theories of sequential control and state. *Theoretical Computer Science*, 103(2) :235–271, september 1992.

[Gen69] Gerhard Gentzen. Investigations into logical deductions. In M. E. Szabo, editor, *The Collected Papers of Gerhard Gentzen*, pages 68–131. North-Holland, Amsterdam, 1969.

[Gir72] Jean-Yves Girard. *Interprétation fonctionnelle et élimination des coupures de l'arithmétique d'ordre supérieur*. Thèse de doctorat d'etat, Université Paris VII, 1972.

[Gir86] Jean-Yves Girard. The system F of variable types, fifteen years later. *Theoretical Computer Science*, 45 :159–192, 1986.

[Gir87a] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50 :1–102, 1987.

[Gir87b] Jean-Yves Girard. *Proof theory and logical complexity*, volume 505 of *Studies in Proof Theory*. Bibliopolis, Napoli, 1987.

[Gir91] Jean-Yves Girard. A new constructive logic : classical logic. *Mathematical Structures in Computer Science*, 1(3) :255–296, 1991.

[Gir01] Jean-Yves Girard. Locus solum. *Mathematical Structures in Computer Science*, 11 :301–506, 2001.

[Gir06] Jean-Yves Girard. *Le Point Aveugle : Cours de logique. Tome 1, Vers la perfection ; Tome 2, Vers l'imperfection*. Hermann, 2006.

[GLT89] Jean-Yves Girard, Yves Lafont, and Paul Taylor. *Proofs and Types*. Number 7 in Cambridge tracts in theoretical computer science. Cambridge University Press, 1989.

[Gri90a] Timothy Griffin. A formulae-as-types notion of control. In *Principles of Programming Languages*. IEEE Computer Society Press, 1990.

[Gri90b] Timothy Griffin. A formulae-as-types notion of control. In *POPL'90*, pages 47–58, 1990.

[GRW08] Chrysida Galanaki, Panos Rondogiannis, and William W. Wadge. An infinite-game semantics for well-founded negation in logic programming. *Annals of Pure and Applied Logic*, 151(2) :70–88, 2008.

[Gug07] Alessio Guglielmi. A system of interaction and structure. *tocl*, 8(1) :1–64, January 2007.

[Gö30] Kurt Gödel. Die vollständigkeit der axiome des logischen funktionenkalküls. *Monatshefte für Mathematik und Physik*, 37, 1930.

[Hei67] Jean Van Heijenoort. *From Frege to Gödel : A Source Book in Mathematical Logic, 1879-1931*. Harvard University Press, 1967.

[HG08] Hugo Herbelin and Silvia Ghilezan. An approach to call-by-name delimited continuations. In *Principles of Programming Languages*. ACM Sigplan, January 2008.

[How80] William A. Howard. The formulae-as-type notion of construction, 1969. In J. P. Seldin and R. Hindley, editors, *To H. B. Curry : Essays in Combinatory Logic, Lambda Calculus, and Formalism*, pages 479–490. Academic Press, New York, 1980.

[HS02] Martin Hofmann and Thomas Streicher. Completeness of continuation models for lambda-mu-calculus. *Information and Computation*, 179(2) :332–355, December 2002.

[JNS05] Radha Jagadeesan, Gopalan Nadathur, and Vijay Saraswat. Testing concurrent systems : An interpretation of intuitionistic logic. In *Proceedings of FSTTCS*, 2005.

[Jol00] Thierry Joly. *Codages, séparabilité et représentation de fonctions en λ-calcul simplement typé et dans d'autres systèmes de types*. Thèse de doctorat, Université Paris VII, January 2000.

[Kle52] Stephen Cole Kleene. Permutabilities of inferences in Gentzen's calculi LK and LJ. *Memoirs of the American Mathematical Society*, 10 :1–26, 1952.

[Kle67] Stephen Cole Kleene. *Mathematical Logic*. John Wiley & Sons, Inc., 1967.

[Kle04] Stephen Cole Kleene. *Logique Mathématique*. Jacques Gabay, 2004. Traduction de [Kle67] par Jean Largeault.

[Kri90] Jean-Louis Krivine. *Lambda-calcul : Types et Modèles*. Masson, 1990.

[Kri92] Jean-Louis Krivine. Un interpréteur du λ-calcul. manuscript, disponible à `http://www.pps.jussieu.fr/~krivine/articles/interprt.pdf`, 1992.

[Laf95] Yves Lafont. From proof nets to interaction nets. In Jean-Yves Girard, Yves Lafont, and Laurent Regnier, editors, *Advances in Linear Logic*, volume 222 of *London Mathematical Society Lecture Note Series*, pages 225–247. Cambridge University Press, 1995.

[Lau02] Olivier Laurent. *Étude de la polarisation en logique*. Thèse de doctorat, Université Aix-Marseille II, March 2002.

[Lau03a] Olivier Laurent. Krivine's abstract machine and the $\lambda\mu$-calculus (an overview). note manuscrite, September 2003.

[Lau03b] Olivier Laurent. Polarized proof-nets and $\lambda\mu$-calculus. *Theoretical Computer Science*, 290(1) :161–188, 2003.

[Lau04a] Olivier Laurent. On the denotational semantics of the untyped lambda-mu calculus. Unpublished note, February 2004.

[Lau04b] Olivier Laurent. Polarized games. *Annals of Pure and Applied Logic*, 130(1–3) :79–123, December 2004.

[LC00] Jean-Vincent Loddo and Roberto Di Cosmo. Playing logic programs with the alpha-beta algorithm. In *LPAR*, pages 207–224, 2000.

[LM07] Chuck Liang and Dale Miller. Focusing and polarization in intuitionistic logic. In J. Duparc and T. A. Henzinger, editors, *Computer Science Logic*, volume 4646 of *Lecture Notes in Computer Science*, pages 451–465. Springer, 2007.

[Lod02] Jean-Vincent Loddo. *Généralisation des Jeux Combinatoires et Applications aux Langages Logiques*. PhD thesis, Université Paris VII, 2002.

[Mil96] Dale Miller. Forum : A multiple-conclusion specification logic. *Theoretical Computer Science*, 165(1) :201–232, September 1996.

[Mil99] Dale Miller. Sequent calculus and the specification of computation. In Ulrich Berger and Helmut Schwichtenberg, editors, *Computational Logic*, volume 165 of *Nato ASI Series*, pages 399–444. Springer, 1999.

[MNPS91] Dale Miller, Gopalan Nadathur, Frank Pfenning, and Andre Scedrov. Uniform proofs as a foundation for logic programming. *Annals of Pure and Applied Logic*, 51 :125–157, 1991.

[MS06] Dale Miller and Alexis Saurin. A game semantics for proof search : Preliminary results. In *Proceedings of the Mathematical Foundations of Programming Semantics (MFPS05)*, number 155 in Electronic Notes in Theoretical Computer Science, pages 543–563, 2006.

[MS07] Dale Miller and Alexis Saurin. From proofs to focused proofs : a modular proof of focalization in linear logic. In J. Duparc and T. A. Henzinger, editors, *Computer Science Logic*, volume 4646 of *LNCS*, pages 405–419. Springer-Verlag, 2007.

[Nai86] Lee Naish. *Negation and Control in Prolog*, volume 238 of *Lecture Notes in Computer Science*. Springer, 1986.

[Nai95] Lee Naish. Pruning in logic programming. Technical Report 95/16, Department of Computer Science, University of Melbourne, Melbourne, Australia, june 1995.

[Nou02] Karim Nour. Non deterministic classical logic : The lambda-mu^{++}-calculus. *Mathematical Logic Quarterly*, 48(3) :357–366, 2002.

[Ong96] Luke Ong. A semantic view of classical proofs. In *Logic In Computer Science*, 1996.

[OS97] Luke Ong and Charles Stewart. A Curry-Howard foundation for functional computation with control. In *Principles of Programming Languages*, pages 215–227, 1997.

[Par91] Michel Parigot. Free deduction : An analysis of "computations" in classical logic. In *Proceedings of the Russian Conference on Logic Programming*, volume 592 of *Lecture Notes in Artificial Intelligence*, pages 361–380, London, UK, 1991. Springer-Verlag.

[Par92] Michel Parigot. $\lambda\mu$-calculus : an algorithmic interpretation of classical natural deduction. In *Proceedings of the 1992 International Conference on Logic Programming and Automated Reasonning*, volume 624 of *Lecture Notes in Computer Science*, London, UK, 1992. Springer-Verlag.

[Par93] Michel Parigot. Classical proofs as programs. In *Proceedings of the Third Kurt Gödel Colloquium on Computational Logic and Proof Theory*, volume 713 of *LNCS*, pages 263 – 276, London, UK, 1993. Springer-Verlag.

[Par97] Michel Parigot. Proofs of strong normalisation for second order classical natural deduction. *Journal of Symbolic Logic*, 62(4) :1461–1479, december 1997.

[PR04] David Pym and Eike Ritter. *Reductive Logic and Proof-search : proof theory, semantics, and control*, volume 45. Oxford Logic Guides, Oxford, 2004.

[PR05] David Pym and Eike Ritter. A games semantics for reductive logic and proof-search. In Dan Ghica and Guy McCusker, editors, *GaLoP 2005 : Games for Logic and Programming Languages*, pages 107–123, 2005.

[Pra65] Dag Prawitz. *Natural Deduction*. Almqvist & Wiksell, Uppsala, 1965.

[PS08] Michele Pagani and Alexis Saurin. Stream associative nets and $\Lambda\mu$-calculus. Technical Report 6431, INRIA, January 2008.

[Py98] Walter Py. *Confluence en $\lambda\mu$-calcul*. Thèse de doctorat, Université de Savoie, 1998.

[Reg92] Laurent Regnier. *Lambda-Calcul et Réseaux*. Thèse de doctorat, Université Paris VII, 1992.

[Sau05] Alexis Saurin. Separation with streams in the $\Lambda\mu$-calculus. In *Logic In Computer Science*, pages 356–365, Chicago, 2005. IEEE Computer Society Press.

[Sau07] Alexis Saurin. Typing streams in the $\Lambda\mu$-calculus : extended abstract. In *Proceedings of the Short Papers Session at LPAR 2007*, October 2007.

[Sau08a] Alexis Saurin. On the relations between the syntactic theories of $\lambda\mu$-calculi. In *17th EACSL Annual Conference on Computer Science Logic 2008 (CSL 2008)*, Lecture Notes in Computer Science. Springer, September 2008.

[Sau08b] Alexis Saurin. Towards ludics programming : Interactive proof search. In *24th International Conference on Logic Programming (ICLP2008)*, Lecture Notes in Computer Science. Springer, December 2008. À paraître.

[Sco76] Dana S. Scott. Data types as lattices. *SIAM Journal of Computing*, 5(3) :522–587, 1976.

[Sel01] Peter Selinger. Control categories and duality : on the categorical semantics of the lambda-mu calculus. *Mathematical Structures in Computer Science*, 11(2) :207–260, April 2001.

[Smu68] Raymond M. Smullyan. *First-Order Logic*. Springer-Verlag, New York Inc., 1968.

[SR98] Thomas Streicher and Bernhard Reus. Classical logic, continuation semantics and abstract machines. *Journal of Functional Programming*, 8(6) :543–572, 1998.

[Sta83] Richard Statman. λ-definable functionals and $\beta\eta$ conversion. *Archiv für Mathematische Logik und Grundlagenforschung*, 22 :1–6, 1983.

[Str05] Thomas Streicher. A denotational semantics for Saurin's $\Lambda\mu$-calculus. communication privée, Mars 2005.

[Ter03] Terese. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2003.

[TS96] Anne S. Troelstra and Helmut Schwichtenberg. *Basic Proof Theory*. Cambridge University Press, 1996.

[vE86] Maarten H. van Emden. Quantitative deduction and its fixpoint theory. *Journal of Logic Programming*, 3(1) :37–53, 1986.

Index

Table des matières détaillée