λ-calcul différentiel et logique classique : interactions calculatoires

Lionel Vaux

Université de la Méditerranée, Aix-Marseille 2
U.F.R. de Mathématiques

THÈSE

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ AIX-MARSEILLE 2

Spécialité : Mathématiques

présentée et soutenue publiquement par

Lionel Vaux

le 23 novembre 2007

Titre :

λ-calcul différentiel et logique classique : interactions calculatoires

Directeurs de thèse :	Thomas	Ehrhard
	Laurent	Regnier
Rapporteurs :	Pierre-Louis	Curien
	Philip	Scott
Jury :	Pierre-Louis	Curien
	Thomas	Ehrhard
	Jean-Yves	Girard
	Hugo	Herbelin
	Laurent	Regnier
	Philip	Scott

version du 24 octobre 2023

Cette version comprend un certain nombre de corrections apportées après la soutenance. La plupart sont dues à Hugo Herbelin, que je remercie très chaleureusement pour sa lecture attentive. Outre quelques coquilles et maladresses de vocabulaire, le changement le plus important concerne le théorème 2.37, dont la formulation originale était fausse (26 novembre 2007).

Les définitions 2.1, 2.29 et 5.2 des interrupteurs ont été remaniées : le cas de la (co)déréliction manquait (16 avril 2008).

Le théorème 2.6 comportait une erreur grossière maintenant corrigée. Merci à Michele Pagani de me l'avoir signalée, et de m'avoir fourni des indications précieuses pour clarifier ce passage (18 mars 2009).

La définition 6.1 comportait une erreur typographique pas tout à fait anodine. Merci à Ambroise Lafont de me l'avoir signalée (6 mars 2019).

Il manquait un cas de contextualité dans la définition 6.18. Merci à Tom Hirschowitz me l'avoir signalé (25 novembre 2020).

Il y avait une typo très mineure dans la 8.18 (24 octobre 2023).

Les versions successives de ce document sont disponibles pour mémoire sur :
`http://tel.archives-ouvertes.fr/tel-00194149/fr/`.

Générique :

Ce mémoire est composé par TEX avec les macros LATEX. Le texte courant est rendu avec la police *concrete* de Donald Knuth. Les polices mathématiques proviennent principalement de la famille *euler* d'Herman Zapf. Le symbole ⅋ de la logique linéaire est issu du paquet `cmll` d'Emmanuel Beffara. Les diagrammes catégoriques sont générés grâce aux macros de Michael Barr pour le paquet XY-pic de Kristoffer Rose et Ross Moore, qui produit également la plupart des flèches de réduction. Les dessins sont programmés dans le langage `Asymptote` mis au point par Andy Hammerlindl, John Bowman et Tom Prince. Le tout est compilé à l'aide du script `rubber` d'Emmanuel Beffara, le plus souvent sur une machine animée par la distribution Debian GNU/Linux.

Merci à chacun d'eux, ainsi qu'à tous les contributeurs du monde du logiciel libre.

À Vir.

Mercis

Merci à Thomas Ehrhard et Laurent Regnier, qui m'encadrent en duettistes depuis mon stage de DEA. Dès le début, ils m'ont accordé une grande liberté dans l'orientation de mes recherches tout en m'offrant leur aide, conjointe et complémentaire, chaque fois que c'était nécessaire.
Dire que cette thèse, comme le reste de mes travaux, leur doit énormément est un cliché. Mais enfin c'est vrai.

Merci à Pierre-Louis Curien et Phil Scott pour avoir accepté d'être les rapporteurs de cette thèse. Leurs questions et remarques sur une version préliminaire de cette thèse ont largement contribué aux améliorations apportées dans le document final.

Merci également à Hugo Herbelin de son intérêt pour mon travail.

Merci à eux trois, ainsi qu'à Jean-Yves Girard, pour leur participation au jury de ma soutenance : la présence de chacun d'eux est un honneur.

Merci à Thierry Coquand qui m'a fait faire mes débuts en logique. Si mes thèmes de recherche se sont un peu écartés de ces premiers pas, ma rencontre avec lui a été déterminante pour mon orientation vers la théorie de la démonstration.

Merci à Olivier Laurent, pour l'intérêt qu'il a pu montrer pour mon travail, et pour avoir fait de son mémoire de thèse un texte de référence : j'aurais aimé pouvoir prétendre à pareille exhaustivité.

Merci également à Alexandre Miquel pour nos discussions, trop rares mais toujours enrichissantes.

Merci encore à Gilles Dowek pour l'enthousiasme qu'il a manifesté pour mon travail, ainsi que pour nos interactions passées et à venir.

Merci à Michele Pagani, Damiano Mazza et Daniel de Carvalho pour nos échanges sur la logique linéaire en général et les réseaux différentiels en particulier.

Merci à Pierre Hyvernat pour toutes nos discussions scientifiques ou non, pour avoir participé à un courant Lyon–Suède–Marseille dans lequel je me suis engouffré et, surtout, pour son amitié.

Merci à Daniel Hirschkoff, qui est un peu le dénominateur commun des jeunes rigolos passés à l'ENS Lyon avant de migrer vers les laboratoires de logique d'un peu partout.

Merci aux thésards et autres précaires de l'IML, actuels et passés, pour avoir créé un environnement favorable à la recherche et aux échanges de points de vues comme de boissons chaudes. En particulier et dans le désordre, merci à : Nicolas Bedaride, Pierre Boudes, Daniel de Carvalho, Étienne Duchesne, Marc de Falco, Paolo di Giamberardino, Yves Guiraud, Pierre Hyvernat, Marion Le Gonidec, Thierry Monteil, Damiano Mazza, Michele Pagani, etc. Bon courage aux suivants. Merci aux pinces de m'avoir fait connaître un nouvel art de vivre. Et merci à Anne Crumière et Mathieu Sablik pour leur amitié non académique.

Merci à tous les membres de l'équipe LDP et plus généralement de l'IML, pour leur accueil chaleureux et la place qu'ils accordent aux thésards au sein du laboratoire.

Merci à toute ma famille, au sens de plus en plus large que le mot a pris au fil des années, ainsi qu'à la nombreuse bande de joyeux drilles dont l'amitié et le soutien comptent beaucoup pour moi.

Merci à mes sponsors parisiens, Lucien Gaudion et Solène Chaval ; à mon sponsor lyonnais, Damien Pous ; et à mes sponsors marseillais des premiers jours, Roland Hamon et Priscilla Schoukroun.

Un remerciement tout spécial va à mon frère François, qui m'a permis de corriger une erreur dans une version préliminaire de cette thèse.

Merci à celles et ceux que j'ai oublié de remercier.

Merci enfin à Vir, pour sa patience, son soutien et tout ce qu'elle m'apporte jour après jour. Je lui dédie cette rivale, enfin éconduite.

Table des matières

Introduction

À la source de la logique de la programmation, la correspondance de Curry–Howard promeut le point de vue selon lequel *une preuve est un programme* : elle énonce que les termes simplement typés du λ-calcul coïncident avec les preuves en déduction naturelle pour l'implication intuitionniste. La portée de ce slogan ne se limite pas à une simple bijection, associant statiquement le code d'un programme au développement d'une preuve. Il s'agit bien plutôt de remarquer que les étapes d'exécution du programme ainsi généré suivent le déroulement d'un algorithme transformant la structure de la preuve : la β-réduction des λ-termes revient à la procédure d'élimination des coupures en déduction naturelle.

L'étude des invariants de cette dynamique motive l'introduction de sémantiques dénotationnelles. Le principe d'une telle sémantique est d'interpréter les objets finis, inductifs, que sont les preuves ou les termes, par des fonctions entre espaces mathématiquement structurés, de sorte que l'interprétation soit invariante par réduction. Dans la sémantique, les objets peuvent perdre en simplicité, mais le processus complexe et potentiellement long qu'est la réduction est résumé par la composition des fonctions. L'idée est que la structure de l'interprétation doit renseigner sur le quotient introduit sur la syntaxe par la réduction.

La logique linéaire (LL) de Girard [Gir87] est le produit d'une telle analyse. En étudiant la sémantique du λ-calcul dans la catégorie des espaces cohérents, on s'aperçoit que l'interprétation de la flèche intuitionniste $A \to B$ se décompose en $!A \multimap B$, c'est-à-dire l'espace des morphismes linéaires de $!A$ dans B, où $!A$ est en quelque sorte l'espace des « paquets » d'éléments de A (dans une construction qui rappelle l'algèbre tensorielle d'un espace vectoriel). En ce sens, la logique linéaire raffine le calcul des séquents intuitionniste (LJ) en fournissant des informations sur les ressources utilisées : la modalité ! renseigne sur le statut d'argument effaçable et duplicable. Elle fournit également un raffinement de la correspondance de Curry–Howard : le λ-calcul s'interprète naturellement dans le fragment multiplicatif-exponentiel de la logique linéaire (MELL), comme calcul des séquents mais aussi dans la syntaxe moins rigide de réseaux de preuve (voir les thèses de Danos [Dan90] et Regnier [Reg92]).

Il y a deux aspects dans ce raffinement : d'abord la logique linéaire provient d'un retour à la syntaxe depuis une sémantique du λ-calcul ; ensuite la correspondance preuve–programme devient une décomposition des programmes dans les réseaux. Ainsi se produisent les allers-retours entre sémantique dénotation-

nelle et correspondance de Curry–Howard, qui constituent le cœur de la logique de la programmation.

Analyse différentielle des preuves

Dans les mathématiques générales, la linéarité est une notion fondamentale d'algèbre. En logique de la programmation cependant, la linéarité est une notion complètement différente au premier abord : un programme est dit linéaire s'il évalue son argument exactement une fois. On peut préciser ce concept en définissant les *positions linéaires* d'un terme en λ-calcul :

— dans un terme réduit à une variable x, cette occurrence de variable est en position linéaire ;
— dans une abstraction $\lambda x\, s$, les positions linéaires sont celles de s et l'abstraction elle-même ;
— dans une application $(s)\, t$, les positions linéaires sont celles de s et l'application elle-même.

En particulier, l'application est linéaire en la fonction mais pas en l'argument.

D'un point de vue informatique, les positions linéaires sont celles qui guident l'exécution dans la réduction de tête : elles sont exécutées exactement une fois, et non pas copiées ou effacées. Un des apports majeurs de la décomposition de l'implication intuitionniste en logique linéaire est de rendre ce concept saillant. En effet, les positions linéaires sont celles qui ne participent pas à une promotion dans la traduction de Girard : elles se retrouvent à profondeur 0 dans les réseaux, hors de toute boîte.

On peut relier cette notion logique de linéarité à son pendant algébrique. Dans [Ehr01] et [Ehr05], Ehrhard a introduit des sémantiques dénotationnelles de LL dans lesquelles les formules sont interprétées par certains espaces vectoriels topologiques et les preuves traduisant des λ-termes par des fonctions continues définies comme des séries entières sur ces espaces. En contraste avec les espaces de Banach cohérents de Girard [Gir96], ces sémantiques donnent une interprétation satisfaisante des exponentielles, en renonçant à leur uniformité. La linéarité au sens algébrique est généralement vue comme la commutation aux sommes. Or il est bien connu que l'espace des fonctions à valeurs dans un monoïde est un monoïde : la somme des fonctions est définie point par point, par $(f+g)(x) = f(x) + g(x)$. Ceci justifie l'introduction de sommes formelles de termes dans le λ-calcul, avec les identités

$$\begin{aligned} (s+t)\, u &= (s)\, u + (t)\, u \\ \lambda x\, (s+t) &= \lambda x\, s + \lambda x\, t \end{aligned}$$

qui reflètent la définition de la somme des fonctions (et aussi un terme nul 0 vérifiant $(0)\, u = \lambda x\, 0 = 0$). On retrouve que l'application est linéaire en la fonction mais pas en l'argument, en accord avec la notion logique de linéarité.

Cette somme des λ-termes a une nature calculatoire proche d'un opérateur de choix non déterministe. Admettons pour les besoins de la discussion que $s + t$ représente une superposition de s et de t (dans un sens informel). Il est

naturel de considérer que $(s + t)\,u$ a le même comportement que $(s)\,u + (t)\,u$ car choisir d'évaluer s ou t dans $(s + t)\,u$ revient à choisir d'évaluer $(s)\,u$ ou $(t)\,u$: on n'a qu'un seul choix à faire. Par contre, $(u)\,(s + t)$ ne saurait se limiter au comportement de $(u)\,s + (u)\,t$. En effet, l'évaluation de u peut faire appel à plusieurs instances de son argument, qui peuvent alternativement choisir entre s et t. Dans $(u)\,s + (u)\,t$, en revanche, on est limité au cas où u appelle toujours s ou toujours t.

Modèles analytiques. Dans les modèles d'Ehrhard de la logique linéaire, les morphismes considérés sont tous infiniment dérivables. Le λ-calcul différentiel d'Ehrhard et Regnier [ER03] réinterprète cette propriété dans la syntaxe : on peut prendre en compte la dérivation dans le λ-calcul, en accord avec les propriétés de ces modèles, qu'on qualifiera d'*analytiques*. On obtient une syntaxe qui obéit aux lois usuelles du calcul différentiel.

Pour fixer l'intuition, rappelons que si la fonction $f : E \longrightarrow F$ est dérivable, alors sa dérivée f' est une fonction de E dans l'espace des applications linéaires $E \multimap F$. Alors si $a \in E$, $f'(x) \cdot a$ se lit comme la dérivée de f au point x dans la direction a, c'est-à-dire l'application linéaire de $f'(x)$ au vecteur a. La fonction $x \mapsto f'(x) \cdot a$ est donc une fonction de E dans F, qui dépend linéairement de a : on la note $Df \cdot a : E \longrightarrow F$. La dérivée de f devient $Df : E \multimap (E \to F)$: par rapport à la notation f' précédente, ceci revient simplement à changer l'ordre de l'application linéaire et de l'application ordinaire, c'est-à-dire à l'isomorphisme $E \to (E \multimap F) \cong E \multimap (E \to F)$.

En gardant en tête ces considérations sur la dérivée, et à la lumière des modèles analytiques de LL, on peut étendre le λ-calcul comme suit. On introduit une nouvelle construction de terme $Ds \cdot t$, linéaire en s et en t. Si s a le type fonctionnel $A \to B$ et si t est dans A, alors $Ds \cdot t$ est encore de type $A \to B$. L'idée pour définir la réduction est que, dans $D\lambda x\, s \cdot t$, s reçoit exactement une copie linéaire de t, et non plus une boîte librement effaçable et duplicable. On réduit

$$D\lambda x\, s \cdot t \to \lambda x \left(\frac{\partial s}{\partial x} \cdot t \right)$$

où $\frac{\partial s}{\partial x} \cdot t$ est la somme de tous les termes obtenus en remplaçant exactement une occurrence linéaire de x par t (c'est donc le terme nul 0 si x n'est pas libre dans s).

Cette définition est en accord avec l'idée que la somme représente une superposition non déterministe. Mais c'est surtout une relecture à travers la correspondance de Curry–Howard–Girard de l'interaction des règles structurelles et de la déréliction de MELL avec la structure analytique des modèles d'Ehrhard. Cette dernière peut se décomposer en une structure monoïdale sur les exponentielles ($!A$) et un morphisme dual de la déréliction, qui se comporte comme un opérateur de dérivation en 0.

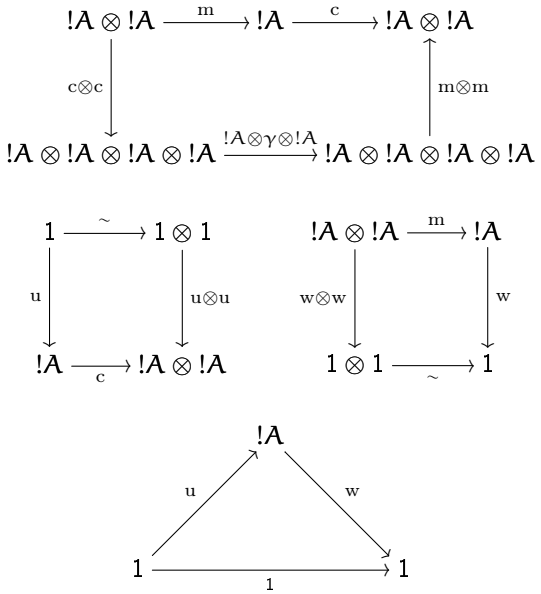
Règles costructurelles. En effet, dans les modèles de LL, la présence d'une somme sur les morphismes induit une structure de $\otimes$-monoïde sur l'exponentielle $!$, qui peut se définir comme un produit de convolution. On peut introduire

$m : !A \otimes !A \longrightarrow !A$ et $u : 1 \longrightarrow !A$ par :

$$\begin{aligned} m &= !\nabla \circ \sigma \\ u &= !\nu \circ s \end{aligned}$$

où ∇ et ν sont la codiagonale et l'unité induites par la somme et le zéro des morphismes, et où σ (resp. s) est l'isomorphisme exponentiel $!A \otimes !A \cong !(A\&A)$ (resp. $1 \cong !\top$). [1]

Les transformations naturelles m et u sont symétriques des interprétations des règles structurelles de MELL : la contraction $c : !A \longrightarrow !A \otimes !A$ et l'affaiblissement $w : !A \longrightarrow 1$. Dans la suite on les qualifiera de *costructurelles*. Les règles structurelles et costructurelles interagissent suivant les lois des *bigèbres*, caractéristiques des algèbres de Hopf :



où γ dénote la commutativité du produit tensoriel (on omet les associativités).

Dérivée. Rappelons qu'un morphisme $f : !A \longrightarrow B$ peut être considéré comme une fonction analytique de A dans B. Alors $f \circ u : 1 \longrightarrow B$ correspond à la valeur de f en 0, et $f \circ m : !(A\&A) \cong !A \otimes !A \longrightarrow B$ correspond à la fonction $(x, y) \mapsto f(x+y)$, où 0 et $+$ sont donnés par la structure additive des morphismes.

Dans ce cadre, la dérivation des fonctions définies sur A est donnée par un morphisme $\partial : A \longrightarrow !A$. L'idée est que $f \circ \partial : A \longrightarrow B$ calcule la dérivée de f

1. Nous suivons les notations de la logique linéaire pour les structures catégoriques de ses modèles : en particulier, & dénote un produit cartésien et $\top$ son unité. Du fait de la présence de sommes, & est ici un biproduit : $\& = \oplus$, d'où la codiagonale. On pourra consulter [Mel03] pour un tour d'horizon des modèles catégoriques de la logique linéaire. Le lecteur pourra également tirer profit de [Fio07] pour une description de l'émergence des règles costructurelles en présence de sommes, et une axiomatisation des modèles analytiques dans ce cadre, comme une variation sur [BCS06].

au point 0, qui est bien une application linéaire. Alors si $a : 1 \longrightarrow A$ est un « vecteur » de A, on pose

$$Df \cdot a = f \circ m \circ ((\partial \circ a) \otimes !A) : 1 \otimes !A \cong !A \longrightarrow B.$$

La définition de la réduction du λ-calcul différentiel par Ehrhard et Regnier provient des propriétés d'un tel opérateur de dérivation, c'est-à-dire de son comportement face aux interprétations des preuves de la logique linéaire.

Ce comportement est présenté dans [ER05], dans un formalisme de réseaux d'interactions (cf. [Laf95]), pour la partie finitaire (sans promotion) de MELL. Dans ces réseaux différentiels, la symétrie associée à la négation linéaire est totale, non seulement au niveau multiplicatif, mais aussi pour les exponentielles : les règles costructurelles (le produit de convolution m et son unité u) sont duales des règles structurelles (contraction c et affaiblissement w), et la dérivée ∂ est duale de la déréliction d. L'élimination des coupures est également symétrique : le réduit d'une coupure entre deux règles est exactement le dual du réduit de la coupure entre les règles duales.

Par la correspondance de Curry–Howard–Girard entre calculs et réseaux, ces réseaux différentiels correspondent à un λ-calcul avec ressources, qui constitue un fragment finitaire du λ-calcul différentiel, où les applications ont pour seul argument le terme nul 0, et dans lequel toutes les constructions sont linéaires : on note $\langle s \rangle (t_1 \cdots t_n) = (D^n s \cdot (t_1, \ldots, t_n))\, 0$. Dans [ER06b], Ehrhard et Regnier ont montré que ce λ-calcul avec ressources permettait d'analyser finement l'application du λ-calcul, en la développant à travers une formule de Taylor sur les termes :

$$\begin{aligned} x^* &= x \\ (\lambda x\, s)^* &= \lambda x\, s^* \\ ((s)\, t)^* &= \sum_{n=0}^{\infty} \frac{1}{n!} \langle s^* \rangle\, t^{*\,n} \end{aligned}$$

qui se comporte bien vis-à-vis de la réduction. En effet, les formes normales des termes qui interviennent dans la somme formelle du membre droit sont des combinaisons linéaires de termes avec ressources dont les supports sont disjoints. On peut donc considérer la somme formelle de ces formes normales : c'est la forme normale du membre droit. Dans [ER06a], Ehrhard et Regnier ont ensuite montré que la forme normale de l'expansion de Taylor d'un terme est précisément celle de l'arbre de Böhm ce terme, en la reliant à l'exécution dans une variante de la machine de Krivine.

Ainsi, la dérivée du λ-calcul différentiel permet une analyse inédite de la β-réduction. Ceci est le reflet d'une décomposition sémantique de la promotion de la logique linéaire, comme le montre la thèse de Carvalho [dC07] : sous certaines conditions, la promotion d'un morphisme linéaire peut se définir à partir de la dérivée et des règles costructurelles, par une formule de Taylor–Ehrhard, dont l'expansion de l'application des λ-termes est une instance.

Logique classique et programmes

Bien qu'initialement formulée pour la déduction naturelle intuitionniste, la correspondance de Curry–Howard peut être étendue à la logique classique. Il est bien connu que la transition d'un système de preuve intuitionniste à un système de preuve classique peut se faire sans adjonction de règles pour l'absurdité, en autorisant des preuves à plusieurs conclusions possibles. En calcul des séquents, le calcul classique LK et le calcul intuitionniste LJ ne diffèrent que par la possibilité pour les séquents de LK de posséder un nombre quelconque de conclusions (au plus une pour LJ). L'interprétation algorithmique de cette extension est claire depuis la proposition de Griffin de typer l'opérateur $\mathcal{C}$ de Felleisen par $\neg\neg A \to A$: les opérateurs de contrôle ($\mathcal{C}$, `call/cc`, *etc.*) sont typés par des formules classiques non prouvables en logique intuitionniste.

Les langages fonctionnels munis d'opérateurs de contrôle ne sont cependant pas tout à fait satisfaisants dans la correspondance preuves–programmes : s'ils introduisent de nouvelles règles de typage, ils ne changent pas le système de déduction. Or la restriction de ce système à au plus une conclusion interdit les transformations les plus naturelles sur les preuves classiques. Cette limitation se reflète dans l'exécution : on est contraint de respecter une stratégie de réduction.

Le $\lambda\mu$-calcul a été introduit dans [Par92] pour pallier ce problème. C'est un calcul pur qui étend le λ-calcul, et dont le système de typage élève la correspondance de Curry–Howard à la logique classique, en changeant le système de déduction plutôt qu'en ajoutant des axiomes. La réduction du $\lambda\mu$-calcul est en effet tirée d'une procédure d'élimination des coupures dans une déduction naturelle sur des séquents classiques (cf. [Par91]). En ce sens le $\lambda\mu$-calcul est un calcul de termes à sorties multiples. C'est la gestion des contextes associés à ces sorties qui constitue le contrôle du flot d'exécution, au sens des opérateurs de contrôle précédents. En particulier, le $\lambda\mu$-calcul fournit une interprétation calculatoire aux règles structurelles sur les conclusions des preuves : celles-ci deviennent de véritables règles de formation des termes, gérant explicitement l'environnement d'exécution au cours de la réduction, et ne se limitent plus à une gestion implicite des contextes de typage.

Curry–Howard–Girard pour la logique classique. Insistons sur le fait que le $\lambda\mu$-calcul n'est pas associé au calcul des séquents LK. Rappelons d'ailleurs que l'élimination des coupures dans LK est fortement non déterministe : une preuve peut admettre des formes normales très différentes. L'étude des traductions de LK dans LL, c'est-à-dire des décorations linéaires des preuves classiques a mené Joinet et Schellinx à définir deux versions *déterministes* de LK enrichies par des annotations orientant l'élimination des coupures : LKT et LKQ [Joi93, Sch94]. Chacun de ces deux systèmes peut se traduire dans LL, en préservant l'élimination des coupures, dans ce qu'on peut voir comme une stratégie de réduction en appel par nom pour LKT et en appel par valeur pour LKQ.

Il est alors possible de redévelopper la correspondance de Curry–Howard–Girard dans ce cadre. Dans sa thèse [Her95], Herbelin a introduit un calcul pur correspondant à LKT : le $\bar{\lambda}\mu$-calcul. En tant que système associé à un calcul des

séquents, la notion d'application d'un terme à un autre (le *modus ponens* de la déduction naturelle) y est remplacé par celle de coupure entre un terme et son environnement d'exécution (ici une pile d'arguments). Curien et Herbelin ont ensuite reformulé et étendu ce calcul dans [CH00] : on obtient un système correspondant à une légère variation sur LK, avec une notion de formule active (au plus une par séquent) gérée par des règles structurelles ; les règles logiques sont restreintes aux formules actives. Ce système non déterministe, le $\bar{\lambda}\mu\tilde{\mu}$-calcul, admet deux restrictions déterministes par l'orientation d'une paire critique : le $\bar{\lambda}\mu\tilde{\mu}_T$-calcul et le $\bar{\lambda}\mu\tilde{\mu}_Q$-calcul correspondent à des variantes de LKT et LKQ. Là encore, les règles structurelles du calcul des séquents reçoivent une interprétation calculatoire en tant que règles de formation des termes et piles, dirigeant l'exécution dans ce qu'on peut voir comme une machine à exécuter des λ-termes.

Polarisation. Une autre approche également fructueuse et éclairante a été initiée par Girard dans [Gir91], avec la définition de LC, un calcul des séquents classique dont l'élimination déterministe des coupures tire parti d'une notion de polarité sur les formules. Bien qu'historiquement déjà présente en logique intuitionniste, la notion de polarité a révélé sa force dans l'étude de la structure des preuves de la logique linéaire. On définit inductivement deux ensembles de formules, dites *polarisées*, échangés par la négation linéaire :

$$\begin{array}{rcl} M, N & ::= & X \mid M \parr N \mid M \mathbin{\&} N \mid ?P \\ P, Q & ::= & X^{\perp} \mid P \otimes Q \mid P \oplus Q \mid !N \end{array}$$

où les premières sont dites négatives et les secondes positives. Le système obtenu à partir de MELL en ne considérant que les formules polarisées suffit à décomposer LJ : la traduction de Girard des formules implicatives est négative. Or le fragment polarisé de LL possède une théorie de la démonstration largement simplifiée par rapport au système complet : entr'autres, les réseaux de preuves y sont faciles à identifier [Reg92, Annexe B], et les conclusions d'une preuves comptent au plus une formule positive.

Partant de LC, Laurent a introduit le système LLP comme une extension du fragment polarisé de LL où, essentiellement, les règles structurelles sont étendues des exponentielles (?P à droite des séquents) aux formules polarisées (négatives à droite des séquents). Comme les formules intuitionnistes peuvent être considérées comme des cas particuliers de formules négatives, ce relâchement revient à autoriser contraction et affaiblissement sur les conclusions, caractéristiques de la logique classique. En contraste avec l'approche LKT/LKQ, qui produisait le pouvoir de preuve classique en changeant la traduction des formules, on étend conservativement la relation entre LJ et LL à une correspondance entre LK et LLP, toujours par la traduction de Girard. Laurent montre alors que LLP possède un caractère canonique dans la décomposition linéaire de la logique classique : on factorise les précédentes déterminisations de LK (déduction naturelle classique, LKQ, LKT, LC, *etc.*) à travers LLP. Ceci offre une éclairage par la logique linéaire des rapports entre logique classique et logique intuitionniste.

Sur le plan sémantique, LLP peut s'interpréter (comme LC) dans les espaces de corrélation de Girard et (comme le $\lambda\mu$-calcul) dans les catégories de contrôle

de Selinger [Sel01], ce qui renforce la pertinence du système, via des sémantiques préexistantes. Laurent et Regnier ont ensuite généralisé la construction des espaces de corrélation à partir des espaces cohérents, comme une construction catégorique spécialisant celle de Lafont [Laf88] : si les exponentielles ?P sont supposées être des ⅋-monoïdes libres, alors une variante de la construction de co-Kleisli restreinte aux ⅋-monoïdes fournit une catégorie de contrôle. Encore une fois, on retrouve l'idée qu'étendre la structure des exponentielles aux formules polarisées produit un système classique.

Perspective de la thèse

En résumé, l'extension différentielle de la logique linéaire introduit une symétrie sur les exponentielles, avec des règles costructurelles, et fournit des moyens d'analyse différentielle des preuves par une notion de dérivée. Par ailleurs, LLP étend la correspondance de Curry–Howard–Girard à la logique classique, en relâchant les règles structurelles, c'est-à-dire en étendant canoniquement la structure des formules exponentielles aux formules polarisées. Ce relâchement se traduit dans les calculs de termes, en faisant apparaître les règles structurelles comme des règles de formation des termes. Il est évidemment tentant de rechercher les rapports qu'entretiennent ces deux enrichissements :

(a) D'abord, sont-ils compatibles ? Peut-on réunir ces deux concepts dans un système qui fasse sens ?

(b) Et surtout, produisent-ils des interactions intéressantes ? Leur conjonction offre-t-elle des moyens nouveaux d'analyser ou d'étendre la correspondance preuve–programme.

Nous avons apporté dans [Vau07b] un premier indice de leur compatibilité, sous la forme d'une coexistence paisible au niveau des calculs de termes : on peut définir un λμ-calcul différentiel qui étend conservativement le λ-calcul différentiel d'Ehrhard–Regnier et le λμ-calcul de Parigot, dans un cadre pur ou typé.

La présente thèse propose une étude des interactions produites par une polarisation à la Laurent de la logique linéaire différentielle d'Ehrhard : celle-ci est principalement caractérisée par l'introduction de règles costructurelles sur les formules positives, symétriques des règles structurelles sur les formules négatives.

Réseaux différentiels polarisés. Nous consacrons la première partie de ce mémoire à la relecture dans un cadre formel commun des correspondances de Curry–Howard–Girard, d'une part entre logique classique et réseaux polarisés, d'autre part entre dérivée en λ-calcul et réseaux différentiels. Au chapitre 1, nous introduisons une variante des réseaux d'interaction de Lafont [Laf95] qui permet de prendre en compte la présence de boîtes et de sommes de réseaux. Dans le chapitre 2, nous reformulons dans ce langage très général les principaux résultats de la littérature, et parfois du folklore, concernant les relations entre calculs de termes et réseaux.

La deuxième partie de notre travail consiste en l'introduction de réseaux différentiels polarisés : la polarisation des réseaux différentiels produit naturellement des règles costructurelles sur les formules positives. Le chapitre 3 présente un modèle dénotationnel commun aux réseaux différentiels et aux réseaux polarisés. Le modèle relationnel multiensembliste de la logique linéaire sert de base à une grande variété de raffinements : entr'autres les espaces cohérents [Gir87] (à l'origine de la logique linéaire), les espaces de corrélation [Gir91] et les espaces de finitude [Ehr05]. On a mentionné le fait que les espaces de correlation correspondent à une structure de $\mathfrak{P}$-monoïde sur les interprétations des formules négatives. Par ailleurs, les espaces de finitude tirent parti d'une symétrie sur les exponentielles caractérisée par une structure de bigèbre. Bucciarelli, Ehrhard et Manzanetto ont introduit dans [BEM07] un objet du modèle relationnel, dont nous montrons qu'il réunit ces propriétés.

Nous définissons enuite au chapitre 4 des réseaux symétriques qui sont une version finitaire des réseaux différentiels polarisés : on les obtient à partir des réseaux différentiels en étendant les règles structurelles aux formules négatives et les règles costructurelles aux formules positives. La définition de la dynamique de ces réseaux est suggérée par la sémantique. Enfin, nous introduisons le système complet des réseaux différentiels polarisés au chapitre 5.

Dans la troisième et dernière partie de ce mémoire, nous nous intéressons à la relation entre ces réseaux et des calculs de termes. Le chapitre 6 confirme que le $\lambda\mu$-calcul différentiel est bien l'expression d'une compatibilité entre extension classique et différentielle : il correspond aux réseaux obtenus en réunissant réseaux différentiels et réseaux polarisés, qui forment un fragment stable par réduction des réseaux différentiels polarisés.

On a vu que les extensions classiques de la correspondance de Curry–Howard avaient pour propriété de faire émerger les règles structurelles comme constructions syntaxiques. Dans le même ordre d'idée, notre polarisation des règles costructurelles permet, par un retour aux calculs purs, de faire apparaître les règles costructurelles (le produit de convolution et son unité) comme des constructions sur les piles dans une extension du $\bar{\lambda}\mu$-calcul : le produit de convolution des piles correspond à un entrelacement de choix non déterministes entre des listes d'arguments successifs. Nous rappelons au chapitre 7 la définition de ce $\bar{\lambda}\mu$-calcul avec produit de convolution d'abord introduit dans [Vau07a], et nous démontrons qu'il correspond bien à l'introduction des règles costructurelles sur les formules positives dans les réseaux polarisés de Laurent. Enfin, le chapitre 8 présente une extension différentielle du $\bar{\lambda}\mu$-calcul avec produit de convolution.

Concurrence. Le comportement calculatoire de la dérivée suggère un rapprochement avec la théorie de la concurrence. Par exemple le λ-calcul avec ressources mentionné plus haut est très proche du calcul avec multiplicités introduit par Boudol [Bou93] : ce dernier est un calcul d'ordre supérieur sensible aux ressources et permettant l'expression de traits de concurrence ce qui le rend observationnellement aussi discriminant que le π-calcul de Milner, Parrow et Walker [MPW92] (voir Boudol et al. [BL95, BL00, BCL99]).

De même, la réduction des réseaux différentiels suggère des aspects concur-

rents : la réduction de la dérivée face aux règles structurelles et à la déréliction revient à router la dérivée vers les hypothèse des règles structurelles jusqu'à communiquer avec la déréliction. Ehrhard et Laurent ont d'ailleurs montré qu'on pouvait coder un π-calcul dans les réseaux différentiels [EL07].

Il semble alors légitime de rechercher une syntaxe de termes au plus proche de ces réseaux, dans le but de produire un calcul d'ordre supérieur pour la concurrence. On peut espérer d'un tel calcul qu'il apporte à la théorie de la concurrence la richesse conceptuelle déjà bien établie de la logique de la programmation. La difficulté réside en ce que les réseaux différentiels utilisés dans [EL07] ne sont en général pas corrects (c'est-à-dire qu'ils ne sont pas directement les traductions de preuves) ni même typés. En particulier, ils sont très éloignés des réseaux interprétant le λ-calcul différentiel ou le λ-calcul avec ressources.

Nous mettons à profit la flexibilité apportée par le produit de convolution pour définir un $\bar{\lambda}\mu$-calcul différentiel qui développe autant que possible l'expressivité des réseaux différentiels polarisés. Dans ce calcul, chaque règle de formation des réseaux correspond à une construction syntaxique spécifique. Nous ne prétendons pas, avec l'introduction du $\bar{\lambda}\mu$-calcul différentiel, apporter une réponse définitive à la question d'une correspondance de Curry–Howard pour la concurrence. Il nous semble toutefois constituer un candidat sérieux : en plus de sa grande proximité formelle avec les réseaux différentiels, la conjonction du caractère d'opérateur de choix du produit de convolution et de la notion de ressources introduite par la dérivée, laisse entrevoir la possibilité d'étendre les caractéristiques du λ-calcul avec ressources de Boudol dans un cadre plus explicite.

Digression

Hors du thème de ce mémoire, mentionnons que notre travail sur le λ-calcul différentiel nous a mené à une réflexion sur le statut de la somme dans les systèmes de réécriture d'ordre supérieur. En effet, puisqu'il existe des sémantiques dénotationnelles du λ-calcul fondées sur des espaces vectoriels, et plus généralement des modules, pourquoi se limiter à des sommes formelles ? Il semble naturel de proposer une extension algébrique du λ-calcul, dans laquelle on forme non plus seulement des sommes, mais des combinaisons linéaires finies de termes, à coefficients dans un corps, un anneau, voire un semi-anneau. C'est d'ailleurs le point de vue adopté dans [ER03].

Cette approche présente cependant plusieurs problèmes, en partie déjà constatés par Ehrhard et Regnier, et auxquels nous avons consacré une étude plus systématique [Vau07c]. Si on souhaite assurer la confluence du système, la réduction doit être définie avec soin : il est nécessaire d'assurer que si $s \to s'$ et si le terme t peut s'écrire $as + u$ où a est un scalaire, alors $t \to as' + u$. Mais en présence de coefficient négatifs, l'infini potentiel inhérent à la β-réduction rend cette notion de réduction triviale : tout terme se réduit en tout autre terme. Par ailleurs, on ne conserve la normalisation forte des termes typés que sous des contraintes fortes sur la structure des coefficients. Or ces difficultés n'ont

rien à voir avec, par exemple, la partie différentielle du λ-calcul différentiel : on les observe dès lors qu'on associe dans un même système combinaisons linéaires et points fixes arbitraires. Dans tous les formalismes syntaxiques abordés dans ce mémoire, on ne considèrera donc que des sommes formelles sans coefficients scalaires ou des multiensembles d'objets.

Notons toutefois qu'il est parfaitement légitime et anodin de considérer des combinaisons linéaires à coefficients quelconques dans des formalismes finitaires : par exemple le λ-calcul avec ressources ou les réseaux différentiels sans boîtes. En effet, comme toutes les constructions y sont linéaires, on ne considère jamais que des combinaisons linéaires formelles d'objets eux-mêmes simples, c'est-à-dire sans sommes.

Première partie

Préliminaires

Chapitre 1

Structures d'interaction

Avant de présenter différentes notions de réseaux, on précise le cadre formel dans lequel on se place. On introduit des structures d'interaction qui généralisent les réseaux d'interaction d'Yves Lafont [Laf95] selon deux axes :

— puisque les réseaux qui nous intéressent comportent des boîtes, sur les portes auxiliaires desquelles des interactions peuvent se produire, on ne pourra contraindre les cellules à posséder exactement un port actif ;

— puisque la réduction des réseaux différentiels fait apparaître des sommes, modélisant une forme de non-déterminisme, on devra considérer des sommes formelles (ou des multiensembles) de réseaux.

Les réseaux d'interaction multiports de Damiano Mazza [Maz06] correspondent à nos structures d'interaction simples, c'est-à-dire sans sommes. La seule nouveauté par rapport à son travail est donc l'extension des notions de connexion, de typage et de réduction au cas de multiensembles de réseaux : on suit en cela les idées utilisées par Ehrhard et Regnier dans [ER05].

Il ne faudra pas chercher l'équivalent des résultats les plus intéressants de Mazza (système de combinateurs universels, séparation, *etc.*) dans cette thèse : les structures d'interaction ne sont pas notre sujet d'étude. Nous tentons plutôt de nous décharger des formalités de la présentation dans ce cadre commun : dans la suite, nous nous concentrerons sur les particularités des instances qui nous intéressent.

1.1 Définition et représentation

1.1.1 Structures d'interaction simples

Définition 1.1 *On appelle signature la donnée d'un ensemble* Σ *de symboles muni de deux fonctions à valeurs dans les entiers naturels* $\mathsf{a}, \bar{\mathsf{a}} : \Sigma \longrightarrow \mathbb{N}$. *On appelle arité (resp. coarité) du symbole* α *le nombre* $\mathsf{a}(\alpha)$ *(resp.* $\bar{\mathsf{a}}(\alpha)$*), et degré de* α *la somme* $\mathsf{d}(\alpha) = \mathsf{a}(\alpha) + \bar{\mathsf{a}}(\alpha) + 1$.

Supposons fixée une telle signature.

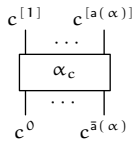
On définit les structures d'interaction comme des circuits construits à partir de cellules : une cellule de symbole α possède $a(\alpha)$ entrées (numérotées de 1 à $a(\alpha)$) et $\bar{a}(\alpha)+1$ sorties (numérotées de 0 à $\bar{a}(\alpha)$). Plus précisément :

Définition 1.2 *Soit* P *un ensemble fini, dit ensemble de ports. Une cellule sur* P *est un couple* $c = (\alpha_c, Po(c))$ *où* α_c *est un symbole et* $Po(c) \in P^{d(\alpha_c)}$ *est une suite finie de ports deux à deux distincts. On note* $a(c) = a(\alpha_c)$, $\bar{a}(c) = \bar{a}(\alpha_c)$ *et* $d(c) = d(\alpha_c)$. *On appelle ports actifs les* $\bar{a}(c)+1$ *premiers éléments de* $Po(c)$ *et ports passifs les suivants. Si* $Po(c) = (p_0, \ldots, p_{\bar{a}(c)}, q_1, \ldots, q_{a(c)})$, *on note* $c^i = p_i$ *et* $c^{[j]} = q_j$. *Le port* c^0, *toujours présent, est appelé port principal de la cellule ; les autres ports actifs sont dits auxiliaires.*

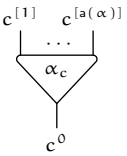
Les ports actifs sont les lieux d'interaction des cellules : lorsqu'on introduira la réduction des structures d'interaction en section 1.2, celle-ci ne se produira qu'entre ports actifs. Par opposition, les ports passifs sont inertes.

Dans les chapitres suivants, on n'utilisera en fait que des cellules de deux sortes : des cellules de coarité 0 (dont le port principal est donc l'unique port actif) qu'on appellera unicellules ; et des cellules-boîtes, qui peuvent interagir sur tous leurs ports, donc d'arité 0. Plutôt que de se limiter à ces deux cas particuliers, il semble mathématiquement sain de présenter le cas général, d'autant plus que la restriction à des structures d'interaction ne contenant que ces deux sortes de cellules ne paraît pas sensiblement simplifier leur développement. Le cas des unicellules sera étudié plus en détail en section 1.4 ; les boîtes seront définies en section 1.5.

Une cellule c sur P peut être vue comme une hyperarête de degré $d(c)$ dans un graphe dont les sommets sont les éléments de P : l'arête est étiquetée par α_c et les sommets adjacents (ports de c) sont ordonnés. On peut la représenter par une forme géométrique



sur le bord de laquelle on fixe l'emplacement des ports comme étant les points d'attache des fils pendants. Quand $\bar{a}(c) = 0$, on la représente en général par



où le port principal est situé sur la pointe.

On a choisi ici de faire figurer les ports actifs en bas et les ports inactifs en haut, de gauche à droite dans les deux cas. Dans le présent chapitre, on se limitera à ces deux formes, mais ceci n'est pas une convention absolue : le seul

point important est de bien préciser l'emplacement des ports chaque fois qu'on introduit une nouvelle forme de cellule.

Continuons à préciser notre notion de circuit.

Définition 1.3 *Soit* P *un ensemble fini. Un fil sur* P *est un multiensemble* $w = [p, q] \in \mathcal{M}_2(P)$. *Si* $p = q$ *on dit que* w *est une boucle, sinon on dit que* w *est un fil propre.*

Définition 1.4 *Une structure d'interaction concrête (SIC)* μ *sur* Σ *est un triplet* $(\mathsf{Po}(\mu), \mathsf{Ce}(\mu), \mathsf{Fi}(\mu))$ *tel que :*

- $\mathsf{Po}(\mu)$ *est un ensemble fini de ports ;*
- $\mathsf{Ce}(\mu)$ *est un ensemble de cellules sur* $\mathsf{Po}(\mu)$ *tel que pour tous* $c \neq c' \in \mathsf{Ce}(\mu)$, $\mathsf{Po}(c) \cap \mathsf{Po}(c') = \emptyset$ *;*
- $\mathsf{Fi}(\mu)$ *est un ensemble de fils sur* $\mathsf{Po}(\mu)$ *;*
- *pour tout port* $p \in \mathsf{Po}(\mu)$, p *apparaît dans au plus une cellule et dans exactement un fil ;*
- *si* p *est le port d'une boucle, i.e.* $[p, p] \in \mathsf{Fi}(\mu)$, *alors* p *n'apparaît dans aucune cellule.*

Les ports qui n'appartiennent ni à une cellule ni à une boucle sont dits libres, les autres sont dits internes : on note $\mathsf{PL}(\mu)$ *l'ensemble des ports libres de* μ *et* $\mathsf{PI}(\mu) = \mathsf{Po}(\mu) \setminus \mathsf{PL}(\mu)$ *l'ensemble des ports internes. Si* $p \in \mathsf{Po}(\mu)$, *on appelle coport de* p *dans* μ *l'unique port* q *tel que* $[p, q] \in \mathsf{Fi}(\mu)$. *On note alors* $q = \overline{p}_\mu$, *ou simplement* $\overline{p}$ *si* μ *est évident.*

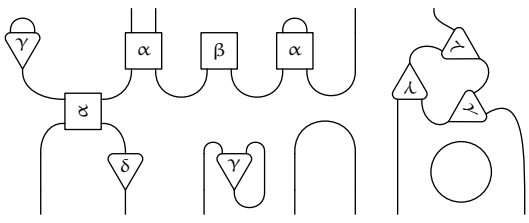
Les fils qui ne contiennent que des ports internes sont appelés fils internes. Les autres sont appelés fils pendants.

On peut voir une SIC μ comme un graphe dont les sommets sont les ports de μ et dont les arêtes sont de deux types : des arêtes simples (les fils de μ) et des hyperarêtes étiquetées par des symboles et dont les sommets adjacents sont ordonnés (les cellules de μ).

Réciproquement, un tel graphe G correspond de manière univoque à une SIC dès que :

- les sommets de G sont organisés en trois classes distinctes :
  - des ports de boucles, de degré 2 et d'hyperdegré 0 ;
  - des ports libres, de degré 1 et d'hyperdegré 0 ;
  - des ports de cellules, de degré 1 et d'hyperdegré 1 ;
- chaque hyperarête étiquetée par α est de degré $d(\alpha)$.

On peut alors représenter graphiquement une SIC : par exemple



représente une SIC sur la signature donnée par

$$a : \begin{cases} \alpha & \mapsto & 2 \\ \beta & \mapsto & 0 \\ \gamma & \mapsto & 2 \\ \delta & \mapsto & 1 \end{cases} \qquad \text{et} \qquad \bar{a} : \begin{cases} \alpha & \mapsto & 1 \\ \beta & \mapsto & 1 \\ \gamma & \mapsto & 0 \\ \delta & \mapsto & 0 \end{cases} .$$

Les fils sont représentés par des... fils et les cellules sont représentées par des formes géométriques comme convenu précédemment. Les ports de cellules sont les points d'attache des fils aux cellules, les ports libres sont les bouts de fils pendants, et les ports de boucles sont implicites. Cette représentation introduit une surcharge : on peut voir une cellule comme une SIC générique ne contenant que cette cellule (il suffit de rajouter un port libre par port de cellule). La correspondance entre ces deux lectures est suffisamment transparente pour qu'on ne l'explicite que rarement dans la suite.

Il est naturel d'identifier deux SIC dont les graphes sont isomorphes. Plus précisément :

Définition 1.5 *Deux SIC* μ *et* μ' *sont isomorphes si elles ne diffèrent que par les noms des ports internes, i.e. s'il existe une bijection* $\varphi : \mathsf{PI}(\mu) \xrightarrow{\sim} \mathsf{PI}(\mu')$ *qui préserve les cellules et les fils.*

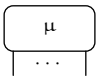
La notion d'isomorphisme entre SIC correspond alors à une notion d'isomorphisme de graphes, avec la restriction que les ports libres sont fixés. Les classes d'isomorphisme définissent une relation d'équivalence entre SIC, très proche de l'α-équivalence, en λ-calcul par exemple : les ports libres correspondent aux variables libres et les ports internes aux variables liées.

Définition 1.6 *On appelle structure d'interaction simple (SIS) toute classe d'isomorphisme de SIC et on note* SIS *l'ensemble des SIS.*

On pourra aussi écrire SIS_Σ quand on voudra expliciter la signature.

Dans la suite on confond systématiquement SIS et SIC : on manipule des SIC en parlant de SIS, ce qui indique implicitement que les choses dites sont valables à isomorphisme près, pour tout instanciation pertinente des ports internes. En particulier, on importe tout le vocabulaire concernant les SIC sur les SIS.

On représentera génériquement une SIS μ comme :



où on fait figurer les fils pendants correspondant aux ports libres.

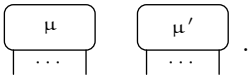
1.1.2 Opérations

Définition 1.7 *Soit* $\mu \in \mathsf{SIS}$. *On appelle sous-SIS de* μ *tout* $\nu \in \mathsf{SIS}$ *tel que* $\mathsf{Po}(\nu) \subset \mathsf{Po}(\mu)$, $\mathsf{Ce}(\nu) \subset \mathsf{Ce}(\mu)$ *et* $\mathsf{Fi}(\nu) \subset \mathsf{Fi}(\mu)$. *On note alors* $\nu \subset \mu$.

En particulier, une sous-SIS de μ est un de ses sous-graphes. Il est à noter que certains ports de cellules dans μ peuvent devenir des ports libres dans $\nu \subset \mu$.

Définition 1.8 *Si μ et μ' sont des SIS telles que $\mathsf{PL}(\mu) \cap \mathsf{PL}(\mu') = \emptyset$ on note $\mu \parallel \mu'$ la SIS obtenue en juxtaposant μ et μ' : on a alors $\mathsf{PL}(\mu \parallel \mu') = \mathsf{PL}(\mu) \cup \mathsf{PL}(\mu')$. On note ε la SIS vide : en particulier $\mathsf{PL}(\varepsilon) = \emptyset$.*

Graphiquement, $\mu \parallel \mu'$ est obtenu en faisant figurer μ et μ' côte-à-côte :



On a bien sûr $\mu \subset \mu \parallel \mu'$ et $\mu \parallel \varepsilon = \mu$.

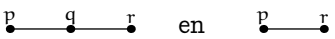
Définition 1.9 *On appelle interface (resp. interface partielle) d'une SIS μ, une énumération $I = (p_1, \ldots, p_n)$ des ports libres de μ (resp. d'un sous ensemble de $\mathsf{PL}(\mu)$). On appelle structure d'interaction simple avec interface partielle (SISIP) la donnée d'une SIS μ et d'une interface partielle I de μ : on la note μ^I. On appelle structure d'interaction simple avec interface (SISI) une SISIP dont l'interface partielle est en fait une interface.*

Les interfaces pertinentes sont le plus souvent évidentes sur les dessins.

Définition 1.10 *Soit G un graphe (éventuellement avec arêtes multiples). On appelle simplifié de G tout graphe obtenu à partir de G par la procédure :*

- *sélectionner deux fils propres $w = [p, q]$ et $w' = [q, r]$;*
- *supprimer q, w et w', et ajouter un fil $[p, r]$.*

C'est-à-dire qu'on réécrit



dès que $p \neq q \neq r$ (mais peut-être $p = r$: dans ce cas, on a formé une boucle). Cette relation de réécriture :

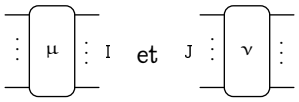
- termine évidemment ;
- élimine tous les ports de degré 2 qui ne sont pas ports de boucles, en laissant les autres degrés inchangés ;
- est fortement confluente à isomorphisme près.

Définition 1.11 *Soient μ^I et ν^J des SISIP telles que I et J soient de même longueur, et $(\mathsf{PL}(\mu) \setminus I) \cap (\mathsf{PL}(\nu) \setminus J) = \emptyset$. On note $\langle \mu^I, \nu^J \rangle$ le résultat de la connexion de μ à ν à travers I et J : c'est la SIS obtenue en identifiant I avec J puis en simplifiant la structure de graphe associée aux fils jusqu'à son unique forme normale.*

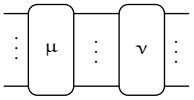
Cette opération est évidemment symétrique : $\langle \mu^I, \nu^J \rangle = \langle \nu^J, \mu^I \rangle$. On omet les interfaces lorsqu'elles sont évidentes ou connues à l'avance. L'étape de simplification est obligatoire, car l'objet obtenu en identifiant I avec J n'est pas une

SIS, sauf si I et J sont vides : dans ce cas $\langle \mu^{\emptyset}, \nu^{\emptyset} \rangle = \mu \parallel \nu$. Il faut en particulier noter que cette opération peut créer des boucles. Par ailleurs, on n'a pas nécessairement $\mu \subset \langle \mu^I, \nu^J \rangle$.

Graphiquement, si on représente

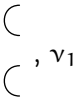


alors on note

$\langle \mu^I, \nu^J \rangle =$ 

.

Notons que $\left\langle \mu^I, \nu_1^J \right\rangle = \left\langle \mu^I, \nu_2^J \right\rangle$ n'implique pas en général $\nu_1 = \nu_2$: par exemple, en posant

$\mu =$  , $\nu_1 =$  et $\nu_2 =$  ,

les deux connexions produisent une boucle.

Lemme 1.12 *Soit* μ^I *une SIS. Si pour tout* $p \in I$, $\overline{p}_\mu \notin I$ *alors* $\left\langle \mu^I, \nu_1^J \right\rangle = \left\langle \mu^I, \nu_2^J \right\rangle$ *implique* $\nu_1 = \nu_2$.

DÉMONSTRATION: Il est déjà clair que ν_1 et ν_2 coïncident en dehors des fils liés à J. Il reste à voir que $\overline{J}_{\nu_1} = \overline{J}_{\nu_2}$. Posons $\overline{I}_\mu = (p_1, \ldots, p_n)$, $J = (q_1, \ldots, q_n)$ et notons μ' la valeur commune de $\left\langle \mu^I, \nu_1^J \right\rangle$ et $\left\langle \mu^I, \nu_2^J \right\rangle$. Pour tout $i \in \{1, \ldots, n\}$, seuls deux cas sont possibles :

(i) Si $\overline{p_i}_{\mu'} = p_j$: par hypothèse, il n'y a pas de fil $[p_i, p_j]$ dans μ donc la simplification de la structure de graphe obtenue en identifiant I et J ne peut toucher aucun des p_i. Nécessairement, $\overline{q_i}_{\nu_1} = \overline{q_i}_{\nu_2} = q_j$.

(ii) Sinon : $\overline{q_i}_{\nu_1} = \overline{q_i}_{\nu_2} = \overline{p_i}_{\mu'}$.

□

Cette propriété est essentielle afin d'assurer le déterminisme de la réduction : voir la définition 1.20.

1.1.3 Structures d'interaction

Définition 1.13 *On appelle structure d'interaction (SI) tout multiensemble fini* μ *de SIS partageant le même ensemble de ports libres, encore noté* $\mathsf{PL}(\mu)$. *On note* SI *l'ensemble des SI.*

On pourra aussi écrire SI_Σ quand on voudra expliciter la signature.

On confond en général une SIS μ avec la SI singleton correspondante. On utilise une notation additive pour les opérations multiensemblistes sur les SI : si μ et ν sont des SI telles que $\mathsf{PL}(\mu) = \mathsf{PL}(\nu)$, on note $\mu + \nu$ leur union

disjointe ; 0 désigne le multiensemble vide (auquel on peut associer n'importe quel ensemble de ports libres). Dans le même ordre d'idée, on notera $n\mu$ pour la structure d'interaction obtenue en ajoutant n fois μ à 0.

La forme générale d'une SI est alors $\mu = \mu_1 + \cdots + \mu_n$ où les μ_i sont des SIS : on écrira $\mu = \overline{\sum}_{i=1}^{n} \mu_i$ dans ce cas.

Définition 1.14 *On appelle structure d'interaction avec interface partielle (SIIP) la donnée d'une SI μ et d'une interface partielle I commune à toutes ses SIS : on note encore μ^I. Idem pour la notion de structure d'interaction avec interface (SII), et on note* SII *l'ensemble des SII. Soient $\mu^I = \overline{\sum}_{i=1}^{n} \mu_i^I$ et $\nu^J = \overline{\sum}_{j=1}^{p} \nu_j^J$ des SIIP telles que I et J soient de même longueur : on pose*

$$\left\langle \mu^I , \nu^J \right\rangle = \overline{\sum_{i,j}} \left\langle \mu_i^I , \nu_j^J \right\rangle \ .$$

Définition 1.15 *Une relation $\tau \subset \mathsf{SII} \times \mathsf{SII}$ est dite contextuelle si :*

- $\mu_1^{I_1} \ \tau \ \mu_2^{I_2}$ *implique* $I_1 = I_2$;
- *et alors* $\left\langle \mu_1^{I_1} , \nu^J \right\rangle \ \tau \ \left\langle \mu_2^{I_1} , \nu^J \right\rangle$ *pour toute SIIP* ν^J.

1.2 Réduction

Définition 1.16 *Une paire active est un couple $((\alpha, i), (\beta, j)) \in (\Sigma \times \mathbb{N})^2$, tel que $0 \leq i \leq \bar{a}(\alpha)$ et $0 \leq j \leq \bar{a}(\beta)$, c'est-à-dire la donnée de deux symboles, et d'un numéro de port actif pour chacun. On notera $\left\langle \alpha^i , \beta^j \right\rangle \in \mathsf{PA}$, où* PA *désigne l'ensemble des paires actives.*

L'idée est que les cellules interagissent via leur ports actifs, ce qui est formalisé par la notion de coupure.

Définition 1.17 *Une coupure dans une SIS μ est un couple $((c, i), (d, j)) \in (\mathsf{Ce}(\mu) \times \mathbb{N})^2$ tel que, en notant $c = (\alpha, (p_0, \ldots, p_n))$ et $d = (\beta, (q_0, \ldots, q_m))$, on a :*

- $c \neq d$;
- $\left\langle \alpha^i , \beta^j \right\rangle$ *est une paire active ;*
- $\left[c^i, d^j\right] = [p_i, q_j]$ *est un fil de μ.*

On note $\left\langle c^i , d^j \right\rangle$ une telle coupure et on lui associe la SISI (aussi notée $\left\langle c^i , d^j \right\rangle$) définie par :

- *les ports de $\left\langle c^i , d^j \right\rangle$ sont les ports de c et d, plus un port frais $\overline{p_k}$ pour tout $k \neq i$ et $\overline{q_l}$ pour tout $l \neq j$;*
- $\mathsf{Ce}\left(\left\langle c^i , d^j \right\rangle\right) = \{c, d\}$;
- $\mathsf{Fi}\left(\left\langle c^i , d^j \right\rangle\right) = \{[p_i, q_j]\} \cup \{[p_k, \overline{p_k}] \, ; k \neq i\} \cup \{[q_l, \overline{q_l}] \, ; l \neq j\}$;
- *l'interface de $\left\langle c^i , d^j \right\rangle$ est*

$$I_{c,i}^{d,j} = (\overline{p_0}, \ldots, \overline{p_{i-1}}, \overline{p_{i+1}}, \ldots, \overline{p_n}, \overline{q_0}, \ldots, \overline{q_{j-1}}, \overline{q_{j+1}}, \ldots, \overline{q_m}) \ .$$

Remarquons que la SIS $\langle c^i , d^j \rangle$ n'est pas nécessairement une sous-SIS de μ. Par contre, par le lemme 1.12, il n'y a qu'une seule manière d'écrire

$$\mu = \langle \nu^I , \langle c^i , d^j \rangle \rangle$$

au renommage des ports de I près. Le but de la réduction est évidemment d'éliminer les coupures.

Définition 1.18 *Un système de réduction est une fonction partielle* $\bowtie : \mathsf{PA} \dashrightarrow \mathsf{SII}$, *telle que* $\beta^j \bowtie \alpha^i$ *est défini si et seulement si* $\alpha^i \bowtie \beta^j$ *l'est, et vérifiant alors :*

- $\alpha^i \bowtie \beta^j$ *est sans coupure et possède* $d(\alpha) + d(\beta) - 2$ *ports libres ;*
- $\alpha^i \bowtie \beta^j$ *et* $\beta^j \bowtie \alpha^i$ *ont la même SI sous-jacente ;*
- *si* $\alpha^i \bowtie \beta^j$ *est d'interface* $(p_1, \ldots, p_{d(\alpha)-1}, q_1, \ldots, q_{d(\beta)-1})$, *alors* $\beta^j \bowtie \alpha^i$ *est d'interface* $(q_1, \ldots, q_{d(\beta)-1}, p_1, \ldots, p_{d(\alpha)-1})$.

Soit μ *une SIS. Un redex dans* μ *est une coupure* $\langle c^i , d^j \rangle$ *telle que* $\alpha^i_c \bowtie \alpha^j_d$ *soit défini. On notera alors* $c^i \bowtie d^j = \alpha^i_c \bowtie \alpha^j_d$.

Exemple 1.19 *Dans cette section et la suivante, les exemples s'appuient sur la signature des réseaux multiplicatifs de la logique linéaire propositionnelle : ceux-ci rentrent en fait dans le cadre plus simple des réseaux d'interaction de Lafont. On a deux symboles* $\otimes$ *et* $\parr$, *d'arité* 2 *et de coarité* 0. *Il y a quatre paires actives :* $\langle \otimes , \otimes \rangle$, $\langle \otimes , \parr \rangle$, $\langle \parr , \otimes \rangle$ *et* $\langle \parr , \parr \rangle$ *(on omet l'exposant* 0 *qui est le seul possible).*

Le système de réduction habituel est défini uniquement pour les paires actives $\langle \otimes , \parr \rangle$ *et* $\langle \parr , \otimes \rangle$. *On le résume par la règle suivante :*

c'est-à-dire qu'on représente une paire active par une coupure caractéristique, et qu'on donne son réduit. La correspondance entre les ports libres de l'une et l'autre SIS spécifie naturellement l'interface de $\otimes \bowtie \parr$ *: les conditions de symétrie demandées par la définition sont automatiquement respectées.*

On suppose donné un tel système de réduction.

Définition 1.20 *Une étape élémentaire de réduction (EER) est un triplet* (μ, r, μ') *où :*

- $r = \langle c^i , d^j \rangle$ *est un redex de* μ *;*
- $\mu \in \mathsf{SIS}$ *s'écrit* $\mu = \langle \nu^I , r \rangle$ *où* ν^I *est une SISIP ;*
- $\mu \in \mathsf{SI}$ *s'écrit* $\mu' = \langle \nu^I , c^i \bowtie d^j \rangle$.

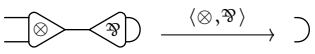
On note alors $\mu \xrightarrow{r} \mu'$.

C'est-à-dire que $\xrightarrow{r}$ est la clôture contextuelle de la relation singleton

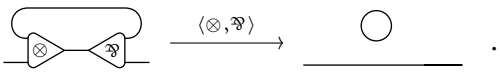
$$\{(\langle c^i , d^j \rangle , c^i \bowtie d^j)\} .$$

Par définition, on a $\mu \xrightarrow{\langle d^j, c^i \rangle} \mu'$ si et seulement si $\mu \xrightarrow{\langle c^i, d^j \rangle} \mu'$. De plus, par le lemme 1.12, μ' est uniquement déterminé par μ et r, c'est-à-dire que $\xrightarrow{r}$ est une fonction partielle. On pourra aussi faire figurer uniquement la paire active correspondante : $\mu \xrightarrow{\langle \alpha^i, \beta^j \rangle} \mu'$. Dans ce cas, plusieurs μ' peuvent convenir. On note enfin $\mu \longrightarrow \mu'$ s'il existe un redex r de μ tel que $\mu \xrightarrow{r} \mu'$.

Exemple 1.21 *Pour les réseaux multiplicatifs, on peut donner les deux réductions suivantes :*



et



.

On étend cette notion de réduction aux SI comme suit :

Définition 1.22 *Un pas de réduction est un couple* (μ, μ') *tel que :*
- $\mu = \sum_{i=1}^{n} \mu_i$ *;*
- $\mu' = \sum_{i=1}^{n} \mu'_i$ *;*
- *pour tout* i, $\mu_i = \mu'_i$ *ou* $\mu_i \longrightarrow \mu'_i$ *;*
- *il existe un* i *tel que* $\mu_i \longrightarrow \mu'_i$.

On note alors $\mu \longrightarrow \mu'$.

En particulier, un pas de réduction à partir d'une SIS n'est rien d'autre qu'une EER.

Définition 1.23 *Si* $n \in \mathbb{N}$, *on note* $\longrightarrow^n$ *l'itérée* n*-ième de* $\longrightarrow$ *:*
- $\mu \longrightarrow^0 \mu'$ *si* $\mu = \mu'$ *;*
- $\mu \xrightarrow{n+1} \mu'$ *s'il existe* μ'' *tel que* $\mu \longrightarrow \mu''$ *et* $\mu'' \longrightarrow^n \mu'$.

On note $\longrightarrow^*$ *(resp.* $\longrightarrow^?$, $\longrightarrow^+$*) la fermeture réflexive et transitive* $\xrightarrow{n \geq 0}$ *(resp. la fermeture réflexive* $\xrightarrow{n \leq 1}$*, la fermeture transitive* $\xrightarrow{n > 0}$*) de* $\longrightarrow$.

1.3 Typage

Définition 1.24 *Un système de types pour* Σ *est constitué des données suivantes :*
- *un ensemble* $\mathcal{T}$*, dit ensemble de types, muni d'une négation involutive, i.e.* $(\cdot)^\perp : \mathcal{T} \xrightarrow{\sim} \mathcal{T}$ *telle que* $A^{\perp\perp} = A$ *pour tout* $A \in \mathcal{T}$ *;*
- *pour tout* $\alpha \in \Sigma$*, une relation* $(\vDash_\alpha) \subset \mathcal{T}^{d(\alpha)}$.

On écrira

$$B_1, \ldots, B_{a(\alpha)} \vDash_\alpha A_0, \ldots, A_{\bar{a}(\alpha)}$$

si et seulement si $(A_0^\perp, \ldots, A_{\bar{a}(\alpha)}^\perp, B_1, \ldots, B_{a(\alpha)}) \in \vDash_\alpha$.

On suppose fixé un tel système de types.

Définition 1.25 *Une SIS typée (SIST) est un couple* (μ^I, τ) *où* μ^I *est une SISI et* $\tau : \mathrm{Po}(\mu) \longrightarrow \mathcal{T}$ *vérifient les propriétés suivantes :*

— *si* $[p, q]$ *est un fil propre de* μ *alors* $\tau(p) = \tau(q)^\perp$ *;*

— *pour toute cellule* $c \in \mathrm{Ce}(\mu)$*,*

$$\tau(c^{[1]}), \ldots, \tau(c^{[a(c)]}) \vDash_{\alpha_c} \tau(c^0), \ldots, \tau(c^{\bar{a}(c)}) \ .$$

Si $I = (p_1, \ldots, p_n)$*, on note alors* $(\mu^I, \tau) \vdash \tau(p_1), \ldots, \tau(p_n)$ *ou simplement* $\mu^I \vdash \tau(p_1), \ldots, \tau(p_n)$*. On appelle la fonction* τ *typage de* μ*, et on dit que* μ^I *est de type* $(\tau(p_1), \ldots, \tau(p_n))$*.*

On représente graphiquement le typage par un étiquetage orienté des fils propres : on note le type d'un des ports du fil, avec une flèche pointant vers ce port. Renverser la flèche revient à pointer vers le coport et donc à nier le type. Cette notation permet également de définir le système de types de manière naturelle, en donnant les typages de SIS réduites à une cellule.

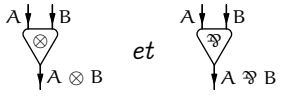
Exemple 1.26 *Pour la logique linéaire multiplicative,* $\mathcal{T}$ *est généré par la grammaire :*

$$A, B ::= X \mid X^\perp \mid A \otimes B \mid A \parr B$$

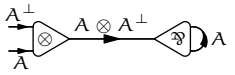
où X *parcourt un ensemble de variables propositionnelles. La négation est définie inductivement à partir des atomes par la loi de Morgan :*

$$(A \otimes B)^\perp = B^\perp \parr A^\perp$$

et l'involutivité. Le système de types est donné par les deux règles :



c'est-à-dire qu'on a $A, B \vDash_\otimes A \otimes B$ *et* $A, B \vDash_\parr A \parr B$ *pour tous types* A *et* B*. Un exemple de SIST est alors :*



Cette notion de typage s'étend naturellement aux SI :

Définition 1.27 *Une SI typée (SIT) est un multiensemble* $\mu = [\mu_1, \ldots, \mu_n]$ *de SIST tel que :*

— *le multiensemble des SISI sous-jacentes est une SII, c'est-à-dire que les* μ_i *partagent une même interface* I *;*

— *les* μ_i *sont du même type : il existe des types* $A_1, \ldots, A_p \in \mathcal{T}$ *fixés tels que* $\mu_i^I \vdash A_1, \ldots, A_p$ *pour tout* i*.*

On écrit alors $\mu \vdash A_1, \ldots, A_p$*.*

Lemme 1.28 *Toute sous-SIS d'une SIS typable* μ *est typable.*

DÉMONSTRATION: C'est direct : il suffit de prendre la restriction du typage aux ports de ν. □

Lemme 1.29 *Si* $\mu = \left\langle \mu_1^{I_1} , \mu_2^{I_2} \right\rangle$ *alors* μ *est typable si et seulement si* μ_1 *et* μ_2 *le sont, de sorte que les types des ports de* I_1 *sont les négations des types des ports de* I_2*.*

DÉMONSTRATION: On peut se réduire au cas des SIS. On étend la notion de typage aux graphes : on associe un type à tout fil propre orienté, de sorte que le type soit nié en inversant l'orientation et avec la contrainte qu'un fil pointant vers un port de cellule a le même type que ce port. On vérifie alors facilement par induction sur le nombre d'étapes de simplification intervenant dans la construction de $\left\langle \mu_1^{I_1} , \mu_2^{I_2} \right\rangle$ que μ est typable si et seulement si le graphe obtenu à partir de $\mu_1 \parallel \mu_2$ en identifiant I_1 et I_2 l'est. □

Bien sûr, ces lemmes paraissent triviaux dès qu'on fait un dessin.

Définition 1.30 *On dit que le système de réduction* $\bowtie$ *est compatible avec le système de types si : pour toute paire active* $\langle \alpha^i , \beta^j \rangle$ *et pour toutes règles de typage*

$$A^{\perp}_{\bar{a}(\alpha)+1}, \ldots, A^{\perp}_{d(\alpha)} \vDash_{\alpha} A_0, \ldots, A_{\bar{a}(\alpha)}$$

et

$$B^{\perp}_{\bar{a}(\beta)+1}, \ldots, B^{\perp}_{d(\beta)} \vDash_{\beta} B_0, \ldots, B_{\bar{a}(\beta)}$$

avec $A_i = B_j^{\perp}$*,*

$$\alpha^i \bowtie \beta^j \vdash (A_0, \ldots, A_{i-1}, A_{i+1}, \ldots, A_{d(\alpha)}, B_0, \ldots, B_{j-1}, B_{j+1}, \ldots, B_{d(\beta)})$$

dès que $\alpha^i \bowtie \beta^j$ *est défini. On dit que* $\bowtie$ *est complet pour le système de types s'il est compatible et si* $\alpha^i \bowtie \beta^j$ *est défini dès qu'un tel typage de la paire active est possible.*

Exemple 1.31 *Le système de types des réseaux multiplicatifs est complet : on vérifie directement qu'il est compatible et les seules coupures typables correspondent à la paire active* $\langle \otimes , \parr \rangle$*.*

Lemme 1.32 *. Supposons que* $\bowtie$ *est compatible avec le système de types. Si* $\mu \vdash A_1, \ldots, A_n$ *alors, pour tout* $\mu' \in \mathsf{SI}$ *tel que* $\mu \longrightarrow^* \mu'$*, il existe un typage de* μ' *tel que* $\mu' \vdash A_1, \ldots, A_n$*. Si de plus* $\bowtie$ *est complet alors une SIT* μ *est en forme normale si et seulement si* μ *ne possède pas de coupure.*

DÉMONSTRATION: Pour le premier point : par transitivité, on se contente de supposer $\mu \longrightarrow \mu'$. Par définition de la réduction sur les SI et du typage des SIT, on se réduit au cas où μ est simple. On conclut par le lemme 1.29.

Pour le second point : c'est direct, car l'hypothèse de complétude implique que toute coupure est réductible. □

1.4 Unicellules

On appelle *symbole fonctionnel* un symbole α tel que $\bar{a}(\alpha) = 0$, et *unicellule* une cellule dont le symbole est fonctionnel : une unicellule ne possède qu'un port actif, qui est son port principal. Le cas où tous les symboles sont fonctionnels est particulier : dans les réseaux d'interaction de Lafont [Laf95], la réduction est toujours fortement confluente. Ici, la présence de sommes et surtout de 0 vient perturber le tableau.

Rappelons que $\longrightarrow^?$ dénote la clôture réflexive de $\longrightarrow$.

Lemme 1.33 *Si Σ ne contient que des symboles fonctionnels alors $\longrightarrow^?$ est fortement confluente : si $\mu \longrightarrow^? \mu'$ et $\mu \longrightarrow^? \mu''$ alors il existe μ''' tel que $\mu' \longrightarrow^? \mu'''$ et $\mu'' \longrightarrow^? \mu'''$.*

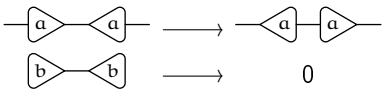
DÉMONSTRATION: Si $\mu = \mu'$ ou $\mu = \mu''$, c'est direct. Sinon, considérons d'abord le cas où μ est une SIS. Il existe des redex r' et r'' de μ tels que $\mu \xrightarrow{r'} \mu'$ et $\mu \xrightarrow{r''} \mu''$. Si $\mu' = \mu''$ on peut choisir $\mu''' = \mu' = \mu''$. Sinon les cellules de r' et r'' sont nécessairement disjointes, car au plus un fil de coupure peut être connecté à une cellule. Alors r' (resp. r'') est laissé intouché dans les SIS de μ'' (resp. μ').

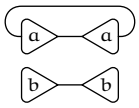
On pose $\mu' = \overline{\sum}_{i=1}^{n} \mu'_i$ et $\mu'' = \overline{\sum}_{j=1}^{p} \mu''_j$. Si par exemple $\mu' = 0$, cela signifie que $\bowtie(r') = 0$. Alors $\mu'' \longrightarrow 0$, car, $\mu''_j \xrightarrow{r'} 0$ pour tout j. Si ni μ' ni μ'' n'est la SI nulle, alors on note $\nu' = \sum_{i=1}^{n} \nu'_i$ et $\nu'' = \sum_{j=1}^{p} \nu''_j$ avec $\mu'_i \xrightarrow{r''} \nu'_i$ pour tout i et $\mu''_j \xrightarrow{r'} \nu''_j$ pour tout j. Alors $\nu' = \nu''$ par associativité et commutativité de la somme des SI, et leur valeur commune convient.

Soit maintenant μ une SI. On pose $\mu = \overline{\sum}_{i=1}^{n} \mu_i$, $\mu' = \sum_{i=1}^{n} \mu'_i$ et $\mu'' = \sum_{i=1}^{n} \mu''_i$, avec $\mu_i \xrightarrow{r'_i} \mu'_i$ ou $\mu_i = \mu'_i$ et $\mu_i \xrightarrow{r''_i} \mu''_i$ ou $\mu_i = \mu''i$ pour tout i. Comme μ_i est simple, le raisonnement précédent fournit μ'''_i tel que $\mu'_i \longrightarrow^? \mu'''_i$ et $\mu''_i \longrightarrow^? \mu'''_i$. Alors $\mu''' = \sum_{i=1}^{n} \mu'''_i$ convient évidemment. □

Malheureusement, ce résultat ne permet pas de conclure grand chose, comparé à la propriété de confluence forte pour $\longrightarrow$: par exemple, on ne peut pas en déduire que les suites de réductions à partir d'un réseau sont toutes de la même longueur. En particulier, on peut trouver un système de réduction d'unicellules pour lequel il existe une structure simple se réduisant à la fois en 0, qui est une forme normale, et en elle-même.

Exemple 1.34 *On pose $\Sigma = \{a, b\}$ où a et b sont fonctionnels, $\bar{a}(a) = 1$ et $\bar{a}(b) = 0$. On donne le système de réduction :*



et on pose $\mu =$  *. Alors* $\mu \longrightarrow \mu$ *et* $\mu \longrightarrow 0$.

On peut toutefois pallier ce problème en interdisant les réductions à 0.

Définition 1.35 *Soit* $\bowtie$ *un système de réduction. On pose* $\alpha^i \overset{\bullet}{\bowtie} \beta^j = \alpha^i \bowtie \beta^j$ *si* $\alpha^i \bowtie \beta^j$ *est défini et non nul, et on laisse* $\alpha^i \overset{\bullet}{\bowtie} \beta^j$ *indéfini sinon. On note* $\multimap$ *la relation de réduction associée à* $\overset{\bullet}{\bowtie}$.

Lemme 1.36 *Si* Σ *ne contient que des symboles fonctionnels alors* $\multimap$ *est fortement confluente.*

DÉMONSTRATION: C'est la même preuve que pour $\longrightarrow^?$, en plus simple. □

Théorème 1.37 *Si* Σ *ne contient que des symboles fonctionnels alors les trois propriétés suivantes sont équivalentes :*

(i) $\multimap$ *normalise faiblement ;*

(ii) $\multimap$ *normalise fortement ;*

(iii) $\longrightarrow$ *normalise fortement.*

DÉMONSTRATION: Les propriétés (i) et (ii) sont équivalentes par le lemme précédent, et se déduisent trivialement de (iii). Réciproquement, si (iii) est fausse, il existe une suite $(\mu_i)_{i\in N}$ de SI telles que $\mu_i \longrightarrow \mu_{i+1}$ pour tout $i \in N$. On peut, sans perte de généralité, supposer que $\mu_i = \mu_i^1 + \mu_i^0$ et $\mu_{i+1} = \mu_i^2 + \mu_i^0$ où :

— μ_i^1 est une SIS,

— $\mu_i^1 \xrightarrow{r_i} \mu_i^2$ où r_i est un redex de μ_i^1.

C'est-à-dire que chaque pas de réduction n'effectue qu'une EER (il suffit de décomposer les éventuels pas comportant plus d'une EER pour obtenir une telle suite). On note $\langle \alpha_i , \beta_i \rangle$ la paire active associée à r_i. Nécessairement, $\{i \in N;\ \alpha_i \bowtie \beta_i \neq 0\}$ est infini : sinon, le nombre de SIS dans μ_i décroît strictement à partir d'un certain rang, ce qui est impossible. On pose $\mu'_0 = \mu_0$. Supposons μ'_i défini :

— si $\alpha_i \bowtie \beta_i = 0$, on pose $\mu'_{i+1} = \mu'_i$;

— sinon on pose μ'_{i+1} l'unique SI telle que $\mu'_i \xrightarrow{r_i} \mu'_{i+1}$ (il est évident que les SIS de μ_i sont toutes dans μ'_i, avec des multiplicités au moins égales).

On extrait alors de $(\mu'_i)_{i\in N}$ une suite infinie de réductions pour $\multimap$, ce qui contredit (ii). □

On termine cette section sur les unicellules en explicitant la notion d'arbre :

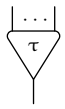
Définition 1.38 *Soit* Σ *un ensemble de symboles fonctionnels. Un arbre sur* Σ *est une SIS munie d'un port libre distingué, dit principal, vérifiant l'une des clauses inductives suivantes :*

— *un fil propre muni d'un port distingué est un arbre ;*

— *si* $\alpha_c \in \Sigma$ *et* $a(c) = n$*, alors, pour tous arbres* $\tau_1, \ldots, \tau_n$*, la SIS obtenue en connectant, pour tout* i*, le port principal de* τ_i *à* $c^{[i]}$*, et* c^0 *à un port libre frais* $\overline{c^0}$*, est un arbre dont le port principal est* $\overline{c^0}$*.*

On appelle ports passifs les ports libres d'un arbre autres que son port principal.

On représente génériquement un arbre τ par



par analogie avec les unicellules.

1.5 Boîtes

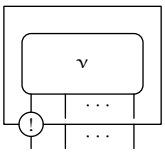
Il est possible d'ajouter des boîtes à des structures d'interaction par une construction inductive. On part des structures de base sur une signature et on considère le point fixe de l'opération consistant à ajouter une nouvelle cellule pour chaque structure. Plus précisément :

Définition 1.39 *Soit* $\mathcal{R}$ *un ensemble de SII possédant toutes au moins un port libre. On note* $\mathcal{R}^\square$ *la signature telle que :*

— *les symboles de* $\mathcal{R}^\square$ *sont les éléments de* $\mathcal{R}$ *(au renommage des ports de l'interface près) ;*

— *pour tout* $\mu \in \mathcal{R}$*,* $a(\mu) = 0$ *et* $\bar{a}(\mu) = |PL(\mu)| - 1$*.*

On représente graphiquement une boîte par :



où les ports de la cellule sont mis en correspondance avec l'interface de la SII. En particulier, on distingue le port libre associé au port principal par l'étiquette ! : les seules boîtes que nous utiliserons seront des boîtes de promotion.

Définition 1.40 *Soit* Σ *une signature. On définit par induction sur* $n \in \mathbb{N}$ *la signature* $\Sigma^{[n]}$ *des structures avec boîtes (resp. la signature* Σ^n *des structures avec boîtes simples) de profondeur au plus* n*. On pose* $\Sigma^{[0]} = \Sigma^0 = \Sigma$*,* $\Sigma^{[n+1]} = \Sigma^{[n]} \cup SI^\square_{\Sigma^{[n]}}$ *et* $\Sigma^{n+1} = \Sigma^n \cup SIS^\square_{\Sigma^n}$*. Enfin, on note* $\Sigma^{[!]} = \bigcup_n \Sigma^{[n]}$ *(resp.* $\Sigma^! = \bigcup_n \Sigma^n$*) la signature des structures avec boîtes (resp. avec boîtes simples).*

On a bien sûr $\Sigma^! \subset \Sigma^{[!]}$. La différence entre $\Sigma^!$ et $\Sigma^{[!]}$ est que les boîtes de $\Sigma^!$ ne contiennent que des SIS. C'est la notion habituelle de boîte dans les réseaux de la logique linéaire (voir la section 2.2). On utilisera la seconde notion $\Sigma^{[!]}$ pour

introduire des boîtes dans les réseaux différentiels (au chapitre 5) : la réduction de ces derniers fait apparaître des sommes, de sorte qu'elle n'est pas stable sur les SIS. Comme on veut pouvoir réduire à l'intérieur des boîtes, il est nécessaire de prendre en compte les sommes. Il est essentiel que les sommes ne « sortent » pas des boîtes : ces dernières sont le lieu de la non-linéarité en logique linéaire.

Définition 1.41 *Soit $\bowtie$ un système de réduction sur les SI de la signature $\Sigma^{[!]}$, et $\longrightarrow$ la relation de réduction associée à $\bowtie$. On définit par induction sur n la réduction à profondeur n, notée $\longrightarrow_{[n]}$:*

- *on pose $\mu \longrightarrow_{[0]} \mu'$ si $\mu \longrightarrow \mu'$;*
- *soit μ une SIS, on pose $\mu \longrightarrow_{[n+1]} \mu'$ si $\mu \longrightarrow \mu'$ ou si $\mu = \langle \mu_0^I , b^J \rangle$ et $\mu' = \langle \mu_0^I , b'^J \rangle$, avec b une boîte de symbole ν^J, b' une boîte de symbole ν'^J et $\nu \longrightarrow_{[n]} \nu'$*
- *soit μ une SI, on pose $\mu \longrightarrow_{[n+1]} \mu'$ si $\mu = \overline{\sum}_{i=1}^{n} \mu_i$ et $\mu' = \sum_{i=1}^{n} \mu'_i$ où, pour tout i, $\mu_i = \mu'_i$ ou $\mu_i \longrightarrow_{[n+1]} \mu'_i$, et ce dernier cas se produit au moins une fois.*

On définit enfin la réduction avec boîtes, en posant $\mu \longrightarrow_{[!]} \mu'$ s'il existe n tel que $\mu \longrightarrow_{[n]} \mu'$.

Si $\bowtie$ est à valeur dans $\mathsf{SIS}_{\Sigma^!}$, alors il est clair que $\longrightarrow_{[!]}$ préserve les structures simples. Notons par ailleurs que $\longrightarrow_{[!]}$ n'est pas associée à un système de réduction, puisqu'une SIS sans coupure (au sens de $\mathsf{SIS}_{\Sigma^{[!]}}$) peut avoir des réduits par $\longrightarrow_{[!]}$ (à profondeur non nulle).

En conséquence, dès qu'on introduit des boîtes, la confluence n'a plus rien d'automatique : non seulement la présence de multiples ports actifs nous fait sortir du cadre des unicellules, mais même à supposer qu'on ait défini un système de réduction sur $\Sigma^{[!]}$ tel que $\longrightarrow$ soit confluente ou fortement confluente, il n'y a aucune raison pour que $\longrightarrow_{[!]}$ conserve cette propriété.

Définition 1.42 *Un système de types avec boîtes est un système de types pour $\Sigma^{[!]}$, tel que la relation de typage de la boîte associée à μ^I ne dépende que des types de μ^I.*

La réduction $\longrightarrow_{[!]}$ préserve le typage dès que $\bowtie$ est compatible avec un tel système de types.

Chapitre 2

Réseaux et calculs

Ce chapitre propose un rapide survol de notions associées aux réseaux de preuve de la logique linéaire, et de leurs liens avec différentes variantes du λ-calcul. Nous ne cherchons pas à être exhaustif mais simplement à préparer le terrain pour la partie suivante : nous montrons comment les structures d'interaction fournissent un cadre formel commun pour l'étude des réseaux polarisés de Laurent et des réseaux d'interaction différentiels d'Ehrhard et Regnier. La plupart des démonstrations de cette section se limitent à des références à des résultats bien connus pour les réseaux de preuve, qu'on adapte, le plus souvent sans difficulté, aux structures d'interaction.

2.1 Types

Les formules de la logique linéaire multiplicative-exponentielle (MELL) propositionnelle sont données par la grammaire :

$$A, B ::= X \mid X^{\perp} \mid A \otimes B \mid A \parr B \mid !A \mid ?A$$

où X parcourt un ensemble de variables propositionnelles. On étend la négation des atomes aux formules par les lois de Morgan :

$$A^{\perp\perp} = A,\ (A \otimes B)^{\perp} = B^{\perp} \parr A^{\perp} \text{ et } (!A)^{\perp} = ?A^{\perp}.$$

Les formules de la logique linéaire polarisée (LLP) forment un sous-ensemble des formules de MELL obtenu par les deux définitions mutuellement inductives suivantes :

$$\begin{aligned} M, N &::= X \mid M \parr N \mid ?P \\ P, Q &::= X^{\perp} \mid P \otimes Q \mid !N. \end{aligned}$$

et on appelle formules *négatives* les premières et formules *positives* les secondes. La négation échange bijectivement formules négatives et positives. On traduit les formules de la logique minimale implicative (c'est-à-dire les types simples du λ-calcul) dans les formules négatives par :

$$\begin{aligned} X^{-} &= X \\ (A \to B)^{-} &= ?(A^{-})^{\perp} \parr B^{-}. \end{aligned}$$

En notant $A \multimap B = A^\perp \parr B$, on retrouve la *traduction négative* de Girard [Gir87, Section 5] : $(A \to B)^- = !A^- \multimap B^-$. Dans la suite on identifiera en général A^- et A, en posant $A \to B = ?A^\perp \parr B$ pour toutes formules A et B de MELL.

En suivant entr'autres [Reg92] et [Dan90], on introduit le type de sortie o comme solution de l'équation :

$$o = ?o^\perp \parr o$$

sur les formules négatives. Notons que ce type o ne saurait être une formule propositionnelle de MELL : il faut l'introduire comme une constante lorsqu'on veut l'utiliser. On considère quatre formules sans atomes construites sur o : o lui-même (type de sortie), $i = o^\perp$ (type d'entrée), $!o$ (type de boîte), $?i$ (type de variable). Les types o et $?i$ sont négatifs et les types i et $!o$ sont positifs. On appelle ensemble des *types* purs l'ensemble $\{o, i, !o, ?i\}$ muni des égalités déduites de $o = ?i \parr o$ et $i = o^\perp$ par les lois de Morgan.

Un système de types basé sur les formules de MELL se restreint directement en un système de types pour les formules polarisées : on type moins de SI. De la même manière, on peut déduire un système de types purs d'un système de types polarisés : il suffit de considérer les instances de règles ne contenant que des types purs. Il n'est par contre pas vrai en général qu'on type alors moins de SI, puisque l'égalité des types purs est plus permissive (voir la section 2.3).

2.2 Réseaux d'interaction pour MELL

Les *structures multiplicatives-exponentielles* (SME) sont les SIS sur la signature $\Sigma^!_{\mathrm{LL}}$ où :

— $\Sigma_{\mathrm{LL}} = \{\otimes, \parr, c, d, w\}$ est un ensemble de symboles fonctionnels ;

— $a(\otimes) = a(\parr) = a(c) = 2$, $a(d) = 1$ et $a(w) = 0$.

Le système de types associé est donné en figure 2.1. Le système de réduction, complet pour le typage, est donné en figures 2.2 et 2.3. On appelle réduction des SME la réduction avec boîtes correspondante $\longrightarrow_{[!]}$. Le lecteur aura reconnu, dans les SME typées, les structures de preuve simplement typées de [Dan90] : il suffit de remplacer les liens axiome et coupure par des fils. Cela revient à considérer les structures de preuve modulo élimination des coupures axiome.

2.2.1 Correction

Définition 2.1 *Soit μ une SME. On appelle interrupteur de μ tout graphe G dont les sommets sont les ports de μ et tel que :*

— *chaque fil de μ est une arête de G ;*

— *pour chaque cellule c de μ, de symbole $\otimes$ ou d, il y a une arête entre le port principal de c et chacun de ses ports passifs ;*

— *pour chaque cellule-boîte c de μ, il y a une arête entre le port principal de c et chacun de ses ports auxiliaires ;*

— *pour chaque cellule c de μ, de symbole $\parr$ ou c, il y a une arête entre le port actif de c et exactement un de ses ports passifs ;*

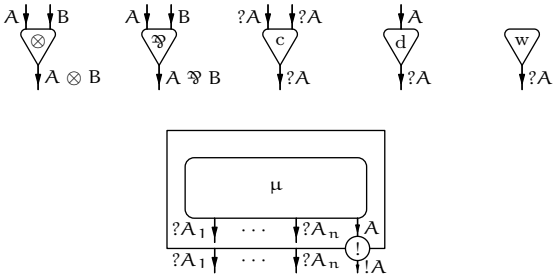


FIGURE 2.1 – Règles de typage des SME

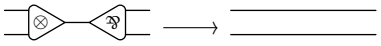


FIGURE 2.2 – Réduction multiplicative

— *il n'y a pas d'arête hors trois cas précédents.*

Définition 2.2 (Critère de Danos) *On définit les réseaux multiplicatifs-exponentiels (RME) par induction sur la profondeur :*

— *un RME de profondeur* 0 *est une SME sans cellule boîte, dont tous les interrupteurs sont acycliques ;*

— *un RME de profondeur* $n+1$ *est une SME dont toutes les boîtes ont pour symbole un RME de profondeur au plus* n*, et dont tous les interrupteurs sont acycliques.*

Lemme 2.3 *Les RME sont stables par réduction.*

DÉMONSTRATION: Il suffit d'inspecter chaque cas de réduction. □

Définition 2.4 *Une SME* μ *est dite faiblement séquentielle si elle peut s'obtenir par les constructions de la figure 2.4, la formation du réseau vide* ε *et la juxtaposition* $\mu_1 \parallel \mu_2$*, en supposant inductivement pour les cas non initiaux que* μ_0*,* μ_1 *et* μ_2 *sont des SME faiblement séquentielles.*

Théorème 2.5 ([Dan90]) *Une SME (non nécessairement typable) est faiblement séquentielle si et seulement si c'est un RME.*

DÉMONSTRATION: On ne donne qu'un schéma de la preuve, qui s'adapte très facilement au cadre des structures d'interaction. La notion-clé est celle de section : on appelle section dans une SIS μ toute unicellule c telle que, pour tout $i \in \{1, \ldots, a(c)\}$, tous les chemins menant de c^0 à $c^{[i]}$ dans l'hypergraphe μ passent par c. On raisonne alors comme suit :

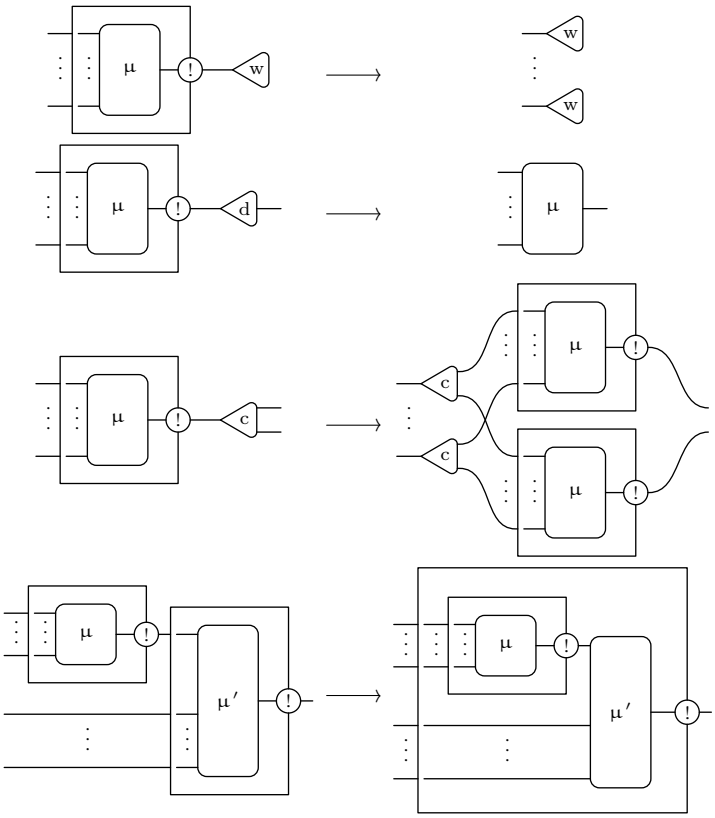


FIGURE 2.3 – Réduction exponentielle

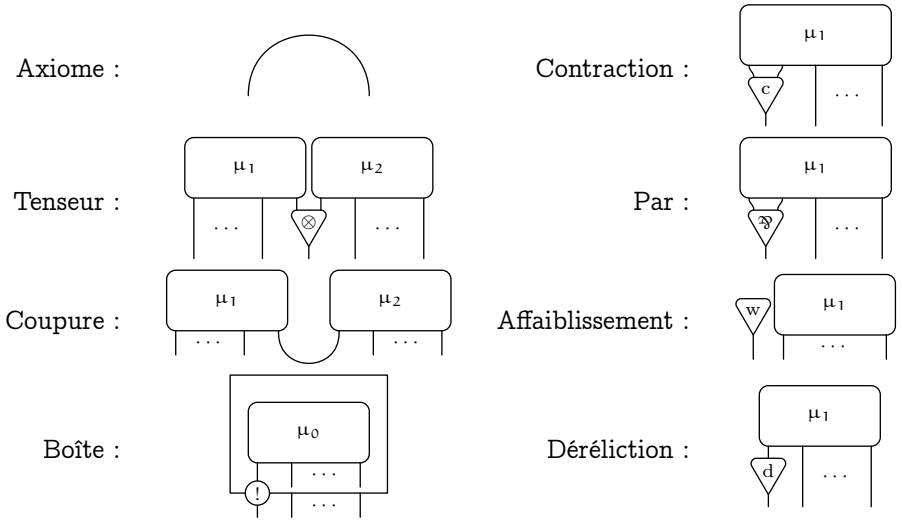


FIGURE 2.4 – Règles de construction des structures séquentielles sur Σ_{LL}.

— il est facile de voir qu'une SME faiblement séquentielle est un RME ;
— on vérifie qu'un RME sans cellule $\parr$ ou c est faiblement séquentiel ;
— on prouve que si un RME μ possède une cellule $\parr$ ou c, alors il en possède une qui est une section (c'est le cœur de la preuve) ;
— on montre alors par récurrence sur le nombre de cellules $\parr$ et c qu'un RME est faiblement séquentiel.

Le résultat de Danos ne concerne que les RME typés, mais le typage n'intervient en fait pas dans la preuve. □

2.2.2 Propriétés

Théorème 2.6 ([Dan90]) *La réduction des RME est confluente.*

DÉMONSTRATION: La preuve de Danos utilise les développements : on annote les coupures présentes dans un réseau, puis on ne réduit que celles-là. [1] On montre que la réduction des développements est nœtherienne, localement confluente et donc confluente. On en déduit la confluence forte de $\longrightarrow^*_{[!]}$, qui est la clôture transitive des développements.

On ne détaille pas au-delà de ce schéma, apparemment simple. La preuve originale est suffisamment longue et complexe pour être intéressante en elle-même. Mentionnons simplement qu'elle est donnée dans le cadre des réseaux purs (voir la section 2.3 pour la notion correspondante dans les RME) mais que seule la correction y joue un rôle.

On pourra d'ailleurs consulter avec profit le travail de Pagani et Tortora

1. La notion de développement est en fait un peu plus complexe, et il faut également annoter comme résidus certaines coupures exponentielles créées par la réduction.

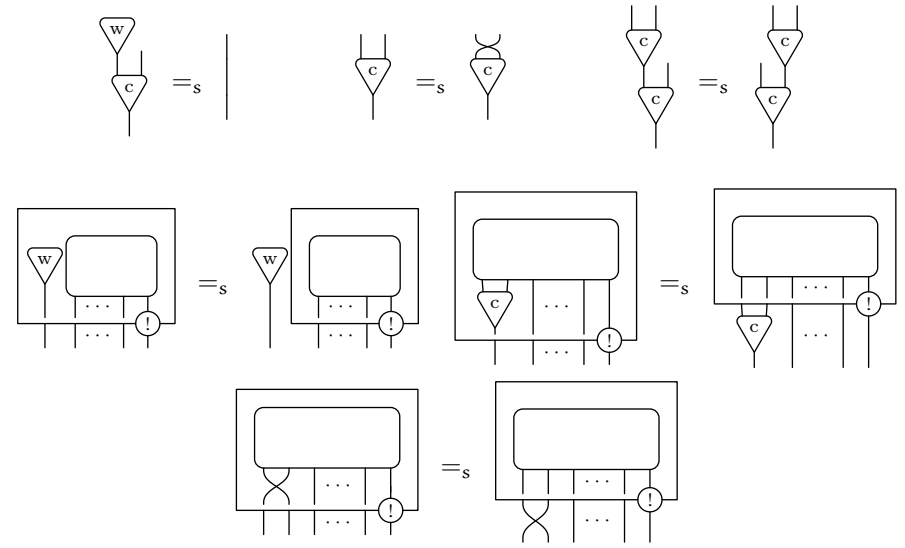


FIGURE 2.5 – Équivalence structurelle sur Σ_{LL}.

de Falco [PdF08] qui généralise ce résultat à la logique linéaire du second ordre avec additifs : ils considèrent explicitement des structures de preuves non typées, et démontrent la confluence à partir de la normalisation forte de la réduction exponentielle. □

Théorème 2.7 *La réduction des RME typés termine.*

DÉMONSTRATION: La preuve originale de Girard par réductibilité [Gir87] dépend d'une propriété de striction prouvée (pour MELL seulement) par Danos dans [Dan90], à partir de la confluence.

Encore une fois, [PdF08] généralise ce résultat à toute la logique linéaire : c'est en fait la seule preuve complète et d'un seul tenant publiée à ce jour. □

2.2.3 Équivalence structurelle

L'équivalence structurelle permet de quotienter ce qu'il reste de séquentialité inutile dans les réseaux : elle correspond à la commutation des règles de contraction, d'affaiblissement et de promotion dans la logique linéaire.

Définition 2.8 *On définit l'équivalence structurelle $=_s$ sur les SME comme la clôture contextuelle, réflexive, symétrique et transitive des transformations de la figure 2.5, considérées à profondeur de boîte quelconque.*

L'équivalence structurelle introduit les identifications nécessaires pour formaliser le fait que la réduction des réseaux simule la β-réduction du λ-calcul,

comme on le montre dans la sous-section suivante. Malheureusement, on ne peut déduire directement les propriétés de confluence et de normalisation forte de la réduction à équivalence structurelle près (c'est-à-dire la relation telle que μ se réduit en ν si et seulement s'il existe μ' et ν' avec $\mu =_s \mu' \longrightarrow_{[!]} \nu' =_s \nu$) de celles de la réduction tout court : voir par exemple [CG99] ou [CKP03] sur le problème de la normalisation forte. Une autre solution, comme dans [Reg92] ou [DR95], est de masquer l'équivalence structurelle en écrasant les cellules correspondantes : ceci revient à considérer uniquement des formes canoniques pour $=_s$. Il faut alors modifier les règles de réduction pour prendre en compte ces restrictions. Alors la réduction des réseaux simule directement la β-réduction

2.2.4 λ-calcul

Le λ-calcul est l'archétype du calcul de termes d'ordre supérieur. On suppose donné un ensemble $\mathcal{V}$ de variables, notées $x, y, z, \ldots$ Les termes sont générés par la grammaire suivante :

$$s, t ::= x \mid \lambda x\, s \mid (s)\, t$$

où l'abstraction $\lambda x\, s$ lie la variable x. Les termes sont alors considérés à α-équivalence près : on renvoie le lecteur à [Kri90] pour des définitions précises de l'α-équivalence, de la substitution d'un terme à une variable (on notera $s\,[x := t]$ pour la *substitution*, sans capture de variable libre, de t à x dans s), et de la plupart des autres notions associées. On note $s \to t$ quand s se réduit en t en une étape de β-réduction : cette dernière est la clôture contextuelle de l'étape élémentaire $(\lambda x\, u)\, v \to u\,[x := v]$ pour tous termes u et v.

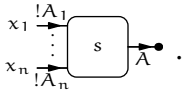
Les types simples du λ-calcul sont les formules de la logique minimale implicative. Rappelons qu'on a plongé ces dernières dans les formules négatives de la logique linéaire par la traduction de Girard (voir la section 2.1). On définit un système de types simples pour le λ-calcul : les jugements de typage sont de la forme $\Gamma \vdash s : A$ où Γ est un environnement de typage, c'est-à-dire une fonction partielle des variables dans les types simples. Les règles de typage sont données dans la première colonne de la figure 2.6.

On traduit le λ-calcul simplement typé dans les RME typés de la manière suivante.

Définition 2.9 *À une dérivation du jugement* $x_1 : A_1, \ldots, x_n : A_n \vdash s : A$, *on associe un RME typé* $[\![s]\!]_{x:A_1,\ldots,x_n:A_n}$ *d'interface* $(x_1, \ldots, x_n, \bullet)$ *tel que*

$$[\![s]\!]_{x:A_1,\ldots,x_n:A_n} \vdash ?A_1^{\perp}, \ldots, ?A_n^{\perp}, A$$

qu'on représente par



Les règles de traduction sont données en figure 2.6.

Cette traduction est simplement une spécialisation pour le λ-calcul de la traduction bien connue de Girard de la logique intuitionniste dans la logique linéaire

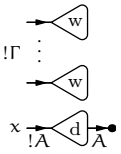
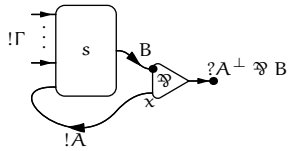
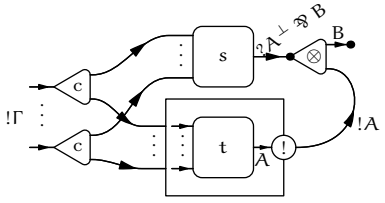
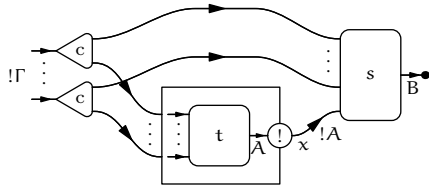
Règles de typage	Traduction des termes typés
$\dfrac{}{\Gamma, x : A \vdash x : A}$	
$\dfrac{\Gamma, x : A \vdash s : B}{\Gamma \vdash \lambda x\, s : A \to B}$	
$\dfrac{\Gamma \vdash s : A \to B \qquad \Gamma \vdash t : A}{\Gamma \vdash (s)\, t : B}$	

FIGURE 2.6 – Règles de typage du λ-calcul et traduction dans les RME typés.

[Gir87]. C'est la version naïve et directe de traductions étudiées par exemple dans [Dan90] et [Reg92]

Lemme 2.10 *Pour tous* λ*-termes* s *et* t *tels que* $\Gamma, x : A \vdash s : B$ *et* $\Gamma \vdash t : B$*, le RME*



se réduit en au moins une étape en un RME structurellement équivalent à la traduction de $\Gamma \vdash s\,[x := t] : B$*.*

DÉMONSTRATION: Facile par induction sur s. □

Théorème 2.11 *La réduction des RME typés simule la* β*-réduction du* λ*-calcul, à équivalence structurelle près : si* $\Gamma \vdash s : A$ *et* $s \to t$ *alors* $\Gamma \vdash t : A$ *et il existe un RME typé* μ *tel que* $[\![s]\!]_\Gamma \longrightarrow_{[!]}^{+} \mu =_s [\![t]\!]_\Gamma$*, où* $\longrightarrow_{[!]}^{+}$ *dénote la fermeture transitive de* $\longrightarrow_{[!]}$*.*

DÉMONSTRATION: On a $\Gamma \vdash t : A$ par réduction du sujet. L'existence de μ se déduit alors par contextualité de $\longrightarrow_{[!]}$ à partir du lemme précédent : il suffit de réduire la coupure $\langle \otimes, \parr \rangle$ correspondant au redex $(\lambda x\, u)\, v$ éliminé par la β-réduction dans s. □

2.3 Réseaux purs

Une *structures multiplicatives-exponentielles pures* (SMEP) est une SME typée sur les types purs. Les SMEP sont stables par réduction, vu le lemme 1.32.

Définition 2.12 (Critère de Regnier) *On définit les réseaux multiplicatifs-exponentiels purs (RMEP) par induction sur la profondeur :*

- *un RMEP de profondeur* 0 *est une SMEP* μ *sans cellule boîte, dont tous les interrupteurs sont acycliques et possèdent une composante connexe de plus qu'il y a de cellules* w *dans* μ *;*
- *un RMEP de profondeur* $n+1$ *est une SMEP* μ *dont toutes les boîtes ont pour symbole un RMEP de profondeur au plus* n*, dont tous les interrupteurs sont acycliques et possèdent une composante connexe de plus que de cellules* w *dans* μ*.*

Lemme 2.13 *Les RMEP sont stables par réduction.*

DÉMONSTRATION: Comme pour le lemme 2.3, il suffit d'inspecter les cas de réduction. □

Définition 2.14 *Une SMEP est dite séquentielle si elle peut s'obtenir inductivement par les constructions de la figure 2.4, en supposant pour les cas non initiaux que* μ_0*,* μ_1 *et* μ_2 *sont des SMEP séquentielles.*

Théorème 2.15 ([Reg92]) *Une SMEP est séquentielle si et seulement si c'est un RMEP.*

DÉMONSTRATION: Encore une fois, il est facile de voir qu'une SMEP séquentielle est un RMEP. Le fait que les types purs soient polarisés joue un rôle crucial dans la démonstration de la réciproque dans [Reg92] : le typage impose des contraintes fortes sur la forme des RMEP. Ces dernières permettent de voyager dans le réseau à la recherche d'une cellule ayant de bonnes propriétés (du même genre que les sections dans la preuve de Danos). Les contraintes du critère de Regnier assurent que ce voyage aboutit toujours, en un nombre fini d'étapes.

Cette remarque que la contrainte de polarisation simplifie grandement l'étude des réseaux et de leurs propriétés est à la base de l'introduction de la logique linéaire polarisée par Laurent (cf. infra). □

Il est clair que si on ajoute aux types simples du λ-calcul une constante o telle que $o = o \to o$ alors tout terme devient typable. On en déduit directement une traduction du λ-calcul pur dans les RMEP, telle que la β-réduction soit simulée par $\longrightarrow_{[!]}$ au sens du théorème 2.11.

2.4 Réseaux polarisés et calculs classiques

Les réseaux polarisés ont été introduits par Laurent dans sa thèse [Lau02]. Ce sont les réseaux de preuve associés à la logique linéaire polarisée (LLP), aussi

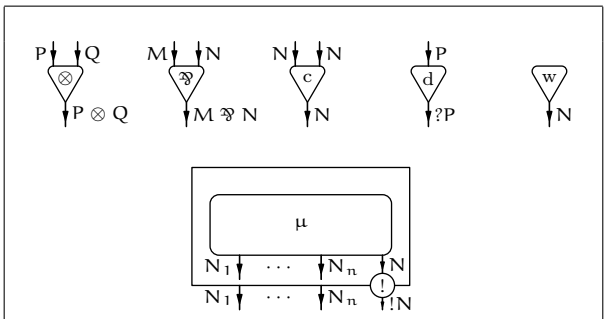


FIGURE 2.7 – Règles de typage des SIP.

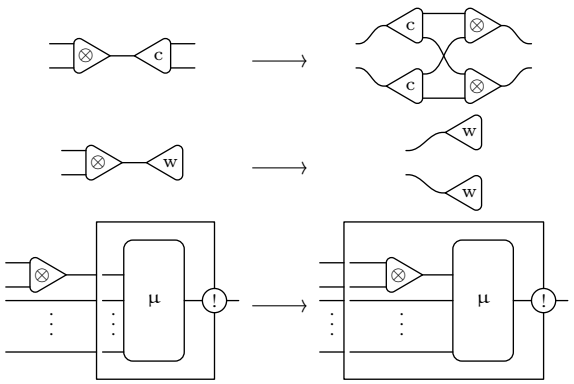


FIGURE 2.8 – Règles de réduction supplémentaires pour les SIP.

introduite dans [Lau02]. On en présente une version limitée aux connecteurs multiplicatifs (sans unités) et exponentiels, dans le formalisme des structures d'interaction.

Les *structures d'interaction polarisées* (SIP) sont encore les SIS sur la signature $\Sigma^{!}_{\mathrm{LL}}$, mais on leur associe cette fois le système de types donné par les règles de la figure 2.7, et on étend le système de réduction des SME par les règles de la figure 2.8. [2] On vérifie aisément que le système de réduction obtenu est complet pour le typage. On appelle structures d'interaction polarisées pures (SIPP) les SIP typées avec les types purs.

2. Les règles de cette figure ne sont pas exactement celles de [Lau02]. En effet, ces dernières étaient données dans un formalisme de réseaux de preuves à la Girard et consistaient en la duplication, l'effacement et l'englobement dans une boîte d'arbres positifs (où un arbre positif est un arbre de tenseurs dont les feuilles sont des boîtes ou des liens axiomes). Les règles que nous proposons en sont une décomposition locale, plus naturelle dans le cadre de réseaux d'interaction.

2.4.1 Correction

Définition 2.16 *Soit μ une SIP typée (ou une SIPP). On appelle graphe de correction de μ le graphe orienté obtenu à partir de μ de la manière suivante :*

- *les sommets sont les cellules, axiomes (fils propres entre deux ports libres) et ports de boucles de μ ;*
- *les arcs sont les fils entre ports de cellules (qu'on attache aux sommets associés aux cellules) et les fils de boucles ;*
- *les arcs associés aux fils propres sont orientés du port de type positif vers le port de type négatif.*

Notons que l'arc associé à un fil propre peut lui-même être une boucle, dans le cas où ses deux ports appartiennent à la même cellule. Dans un graphe orienté, un sommet initial est un sommet sans arête entrante.

Définition 2.17 (Critère de Laurent) *On définit les réseaux d'interaction polarisés (RIP) par induction sur la profondeur :*

- *un RIP de profondeur 0 est une SIP typée sans cellule boîte, dont le graphe de correction est acyclique et possède exactement un sommet initial non w ;*
- *un RIP de profondeur au plus $n+1$ est une SIP typée dont toutes les boîtes ont pour symbole un RIP de profondeur au plus n, et dont le graphe de correction est acyclique et possède exactement un sommet initial non w.*

On définit de manière similaire les réseaux d'interaction polarisés purs (RIPP) à partir des SIPP.

Notons qu'un RIP est toujours typé.

Lemme 2.18 *Les RIP (resp. les RIPP) sont stables par réduction.*

DÉMONSTRATION: Encore une fois, il suffit d'inspecter chaque cas de réduction. □

Théorème 2.19 ([Lau02]) *Une SIP typée (resp. une SIPP) est séquentielle si et seulement si c'est un RIP (resp. un RIPP).*

DÉMONSTRATION: Le preuve de ce théorème raffine et généralise celle de Regnier pour les RMEP. □

2.4.2 Propriétés

On peut traduire injectivement les SIP dans les SME de sorte qu'un pas réduction dans les SIP soit simulé par au moins un pas de réduction dans les SME : si μ est une SIP, on pose μ^b la SME obtenue en plaçant chaque cellule $\otimes$ de μ dans une boîte, et en ajoutant une déréliction d sous le port actif de chaque cellule $\parr$. On vérifie par induction sur le nombre de cellules que μ^b est typée dès que μ l'est.

De cette traduction, dite traduction avec boîtes, Laurent déduit les propriétés suivantes pour les RIP, à partir des propriétés correspondantes pour les RME typés.

Théorème 2.20 *La réduction* $\longrightarrow_{[!]}$ *des RIP (resp. des RIPP) est confluente.*

Théorème 2.21 *La réduction* $\longrightarrow_{[!]}$ *normalise fortement sur les RIP.*

2.4.3 λμ-calcul

Parigot a introduit le λμ-calcul dans [Par92] : c'est le calcul pur associé à une déduction naturelle pour la logique classique. On suppose fixé un ensemble infini dénombrable $\mathcal{N}$ de noms, notés $\alpha, \beta, \gamma, \ldots$ La syntaxe est donnée par les grammaires mutuellement inductives :

$$\begin{aligned} s, t &::= x \mid \lambda x\, s \mid \mu\alpha\, c \mid (s)\, t \\ c &::= [\alpha]\, s \end{aligned}$$

et on appelle termes les premiers objets et commandes les seconds. On considère ces objets modulo α-équivalence : la λ-abstraction $\lambda x\, s$ lie la variable x dans s et la μ-abstraction $\mu\alpha\, c$ lie le nom α dans la commande c. La β-réduction est étendue par

$$(\mu\alpha\, c)\, t \to \mu\alpha\, ((c)_\alpha\, t)$$

où $(c)_\alpha\, t$ est obtenu en remplaçant inductivement chaque sous-commande $[\alpha]\, s$ de c par $[\alpha]\, (s)\, t$ (en évitant la capture de variables ou de noms libres).

Dans la littérature, on trouve le plus souvent la notation $c\, [[\alpha]\, s := [\alpha]\, (s)\, t]$ au lieu de $(c)_\alpha\, t$, ce qui renvoie directement à la définition. Dans la terminologie originale de Parigot [Par92], elle est qualifiée de *substitution structurelle*. Outre sa concision, la notation $(c)_\alpha\, t$ a l'avantage de s'accorder avec l'intuition : $(c)_\alpha\, t$ représente l'application de la sortie nommée α de c à t. Dans la suite, on parlera d'*application nommée*.

On définit un système de types simples pour le λμ-calcul : les jugements de typage sont de la forme $\Gamma \vdash s : A \mid \Delta$ ou $c : (\Gamma \vdash \Delta)$ où Γ (resp. Δ) est un *environnement de variables* (resp. *environnement de noms*), c'est-à-dire une fonction partielle des variables (resp. noms) dans les types. Les règles de typage sont données dans la première colonne de la figure 2.9.

On traduit le λμ-calcul simplement typé dans les RIP de la manière suivante.

Définition 2.22 *À une dérivation du jugement*

$$x_1 : A_1, \ldots, x_n : A_n \vdash s : A \mid \alpha_1 : B_1, \ldots, \alpha_p : B_p$$

on associe un RIP $[\![s]\!]^{\alpha_1:B_1,\ldots,\alpha_p:B_p}_{x_1:A_1,\ldots,x_n:A_n}$ *d'interface* $(x_1, \ldots, x_n, \bullet, \alpha_1, \ldots, \alpha_p)$ *tel que*

$$[\![s]\!]^{\alpha_1:B_1,\ldots,\alpha_p:B_p}_{x_1:A_1,\ldots,x_n:A_n} \vdash ?A_1^\perp, \ldots, ?A_n^\perp, A, B_1, \ldots, B_p$$

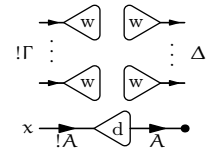
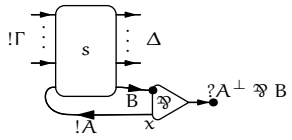
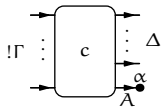
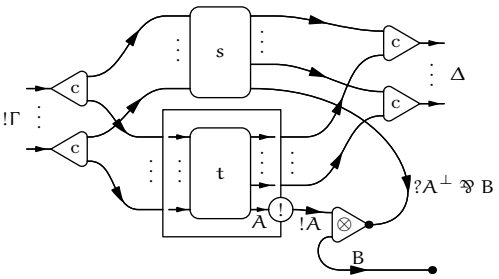
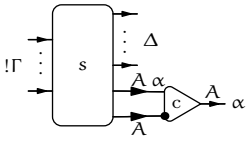
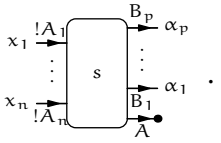
Règles de typage	Traduction des objets typés
$\dfrac{}{\Gamma, x : A \vdash x : A \mid \Delta}$	
$\dfrac{\Gamma, x : A \vdash s : B \mid \Delta}{\Gamma \vdash \lambda x\, s : A \to B \mid \Delta}$	
$\dfrac{c : (\Gamma \vdash \alpha : A, \Delta)}{\Gamma \vdash \mu\alpha\, c : A \mid \Delta}$	
$\dfrac{\Gamma \vdash s : A \to B \mid \Delta \qquad \Gamma \vdash t : A \mid \Delta}{\Gamma \vdash (s)\, t : B \mid \Delta}$	
$\dfrac{\Gamma \vdash s : A \mid \alpha : A, \Delta}{[\alpha]\, s : (\Gamma \vdash \alpha : A, \Delta)}$	

FIGURE 2.9 – Règles de typage du λμ-calcul et traduction dans les RIP typés.

qu'on représente par



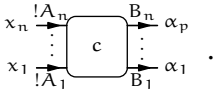
De même, à une dérivation du jugement

$$c : (x_1 : A_1, \ldots, x_n : A_n \vdash \alpha_1 : B_1, \ldots, \alpha_p : B_p)$$

on associe un RIP $[\![c]\!]^{\alpha_1:B_1,\ldots,\alpha_p:B_p}_{x_1:A_1,\ldots,x_n:A_n}$ *d'interface* $(x_1, \ldots, x_n, \alpha_1, \ldots, \alpha_p)$ *tel que*

$$[\![c]\!]^{\alpha_1:B_1,\ldots,\alpha_p:B_p}_{x_1:A_1,\ldots,x_n:A_n} \vdash ?A_1^{\perp}, \ldots, ?A_n^{\perp}, B_1, \ldots, B_p$$

qu'on représente par

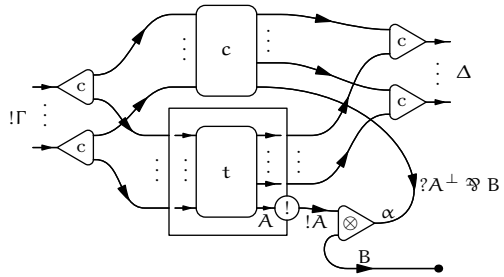


Les règles de traduction sont données en figure 2.9.

Rappelons qu'on a plongé les types simples dans les formules négatives de LLP. Ceci justifie le typage des cellules w et c qu'on a fait pointer vers la droite dans la traduction : A et les formules de Δ sont négatives.

On établit facilement l'analogue du lemme 2.10 dans le cadre du $\lambda\mu$-calcul. De même, on montre le lemme suivant, en même temps que sa variante où on remplace la commande par un terme, par induction sur les objets.

Lemme 2.23 *Pour toute commande* c *et tout terme* t *tels que* $c : (\Gamma \vdash \alpha : A \to B, \Delta)$ *et* $\Gamma \vdash t : A \mid \Delta$*, le RIP*



se réduit en un RIP structurellement équivalent à $[\![(c)_\alpha\, t]\!]^{\alpha:B,\Delta}_{\Gamma}$.

On obtient alors le théorème suivant.

Théorème 2.24 *La réduction des RIP simule la réduction du* $\lambda\mu$*-calcul, à équivalence structurelle près : si* $\Gamma \vdash s : A \mid \Delta$ *et* $s \to t$ *alors* $\Gamma \vdash t : A \mid \Delta$ *et il existe un RIP* μ *tel que* $[\![s]\!]^{\Delta}_{\Gamma} \longrightarrow^{*}_{[!]} \mu =_s [\![t]\!]^{\Delta}_{\Gamma}$.

2.4.4 $\bar{\lambda}\mu$-calcul

Le $\bar{\lambda}\mu$-calcul a été introduit par Herbelin dans sa thèse [Her95]. C'est un calcul pur associé au calcul des séquents classique LKT : LKT est une version déterministe de LK, introduite par Joinet et Schellinx dans leurs thèses [Joi93, Sch94], et qui prend sa source dans l'analyse des liens entre logique classique et logique linéaire.

S'agissant d'un calcul des séquents, l'application (qui se type en déduction naturelle par la règle du *modus ponens*) est remplacée par la notion de coupure. La syntaxe comporte trois catégories d'objets : termes, piles et commandes. L'idée est qu'un terme représente une dérivation d'un séquent avec une conclusion active ; une pile représente une dérivation d'un séquent avec une hypothèse active ; et une commande représente la dérivation obtenue en coupant la conclusion active d'un terme sur l'hypothèse active d'une pile. Les termes, piles et commandes sont donnés par les grammaires mutuellement inductives :

$$\begin{aligned} s &::= x \mid \lambda x\, s \mid \mu\alpha\, c \\ e &::= \alpha \mid s \cdot e \\ c &::= \langle s\,,\, e\rangle \end{aligned}$$

où on considère les termes à α-équivalence près. [3]

La réduction $\rightarrow$ est définie par les deux règles élémentaires suivantes :

$$\begin{aligned} \langle \lambda x\, s\,,\, t \cdot e\rangle &\rightarrow \langle s\,[x := t]\,,\, e\rangle \\ \langle \mu\alpha\, c\,,\, e\rangle &\rightarrow c\,[\alpha := e] \end{aligned}$$

où la substitution est définie de la manière naturelle, en évitant les captures de variables ou de noms.

On définit un système de types simples pour le $\bar{\lambda}\mu$-calcul : les jugements de typage sont de la forme $\Gamma \vdash s : A \mid \Delta$, $\Gamma \mid e : A \vdash \Delta$ ou $c : (\Gamma \vdash \Delta)$. Les règles de typage sont données dans la première colonne de la figure 2.10.

On traduit le $\bar{\lambda}\mu$-calcul simplement typé dans les RIP de la manière suivante.

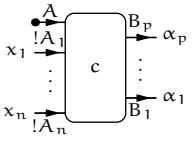
Définition 2.25 *On reproduit la définition 2.22, avec la clause supplémentaire : à une dérivation du jugement*

$$x_1 : A_1, \ldots, x_n : A_n \mid e : A \vdash \alpha_1 : B_1, \ldots, \alpha_p : B_p$$

on associe un RIP $[\![e]\!]^{\alpha_1:B_1,\ldots,\alpha_p:B_p}_{x_1:A_1,\ldots,x_n:A_n}$ *d'interface* $(x_1, \ldots, x_n, \bullet, \alpha_1, \ldots, \alpha_p)$ *tel que*

$$[\![e]\!]^{\alpha_1:B_1,\ldots,\alpha_p:B_p}_{x_1:A_1,\ldots,x_n:A_n} \vdash ?A_1^{\perp}, \ldots, ?A_n^{\perp}, A^{\perp}, B_1, \ldots, B_p$$

qu'on représente par



.

3. On a repris les notations de Curien et Herbelin dans [CH00] pour les constructions du $\bar{\lambda}\mu$-calcul, et non les notations originales de [Her95], plus proches de celles du λ-calcul. Par ailleurs, on ne considère pas ici un système de substitutions explicites, contrairement à Herbelin dans sa thèse.

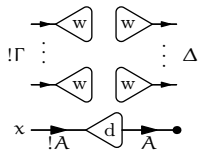
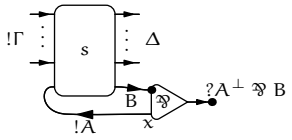
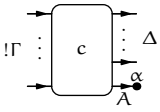
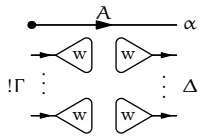
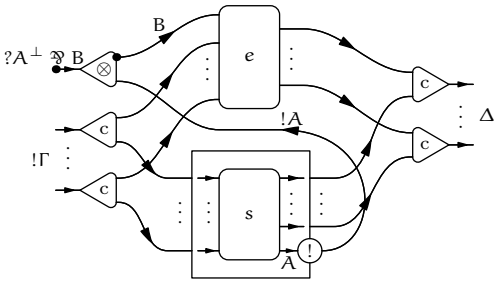
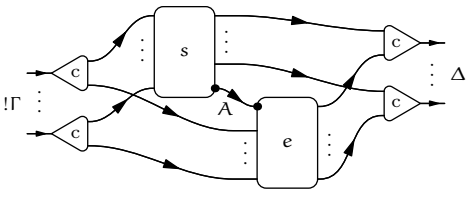
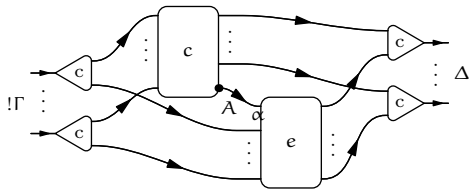
Règles de typage	Traduction des objets typés
$\dfrac{}{\Gamma, x : A \vdash x : A \mid \Delta}$	
$\dfrac{\Gamma, x : A \vdash s : B \mid \Delta}{\Gamma \vdash \lambda x\, s : A \to B \mid \Delta}$	
$\dfrac{c : (\Gamma \vdash \alpha : A, \Delta)}{\Gamma \vdash \mu\alpha\, c : A \mid \Delta}$	
$\dfrac{}{\Gamma \mid \alpha : A \vdash \alpha : A, \Delta}$	
$\dfrac{\Gamma \vdash s : A \mid \alpha : A, \Delta \qquad \Gamma \mid e : B \vdash \Delta}{\Gamma \mid s \cdot e : A \to B \vdash \Delta}$	
$\dfrac{\Gamma \vdash s : A \mid \Delta \qquad \Gamma \mid e : A \vdash \Delta}{\langle s\,,\, e\rangle : (\Gamma \vdash \Delta)}$	

FIGURE 2.10 – Règles de typage du $\bar{\lambda}\mu$-calcul et traduction dans les RIP typés.

Les règles de traduction sont données en figure 2.10.

On montre le lemme suivant, en même temps que ses variantes où on remplace la commande par un terme ou une pile, par induction sur les objets.

Lemme 2.26 *Pour toute commande* c *et toute pile* e *telles que* $c : (\Gamma \vdash \alpha : A, \Delta)$ *et* $\Gamma \mid e : A \vdash \Delta$*, le RIP*



se réduit en un RIP structurellement équivalent à $[\![c\,[\alpha := e]]\!]^{\Delta}_{\Gamma}$.

On obtient alors le théorème suivant, en utilisant le lemme précédent et une variante aisément établie du lemme 2.10.

Théorème 2.27 *La réduction des RIP simule la réduction du* $\bar{\lambda}\mu$*-calcul, au sens du théorème 2.24.*

2.5 Réseaux différentiels et λ-calcul avec ressources

Les réseaux d'interaction différentiels ont été introduits par Ehrhard et Regnier dans [ER05], comme une extension des réseaux d'interaction multiplicatifs, associée aux structures caractéristiques des modèles analytiques de la logique linéaire présentés dans [Ehr01, Ehr05]. Dans ces modèles, les formules de la logique linéaire sont interprétées par certains espaces vectoriels topologiques, et les preuves sont interprétées par des applications linéaires et continues (morphismes).

La sémantique associée à la modalité ! est telle que les morphismes de $!A$ dans B peuvent être définis à partir de séries entières. En particulier, ces morphismes peuvent être dérivés, et on peut définir une multiplication sur $!A$ comme une sorte de produit de convolution proche de celui des distributions [Sch66]. Ce produit permet de définir la dérivée des morphismes de $!A$ dans B à partir de la seule dérivée en zéro.

Les réseaux différentiels sont obtenus en introduisant des cellules duales de la contraction, de l'affaiblissement et de la déréliction, au moyen du produit de convolution, de son unité et de la dérivée en zéro. Les règles de réduction traduisent alors les propriétés de ces opérations, qu'on peut qualifier de costructurelles.

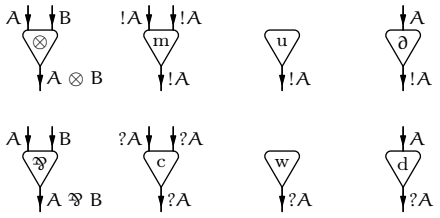


FIGURE 2.11 – Règles de typage des SID.

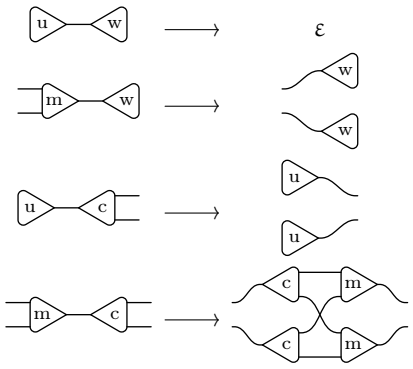


FIGURE 2.12 – Réduction structurelle.

2.5.1 Structures d'interaction différentielles

Définition 2.28 *On considère la signature* Σ_{Diff} *suivante :*

— $\Sigma_{\text{Diff}} = \{\otimes, ⅋, \mathrm{m}, \mathrm{u}, \mathrm{c}, \mathrm{w}, \partial, \mathrm{d}\}$ *est un ensemble de symboles fonctionnels ;*

— $\mathsf{a}(\otimes) = \mathsf{a}(⅋) = \mathsf{a}(\mathrm{m}) = \mathsf{a}(\mathrm{c}) = 2$, $\mathsf{a}(\partial) = \mathsf{a}(\mathrm{d}) = 1$ *et* $\mathsf{a}(\mathrm{u}) = \mathsf{a}(\mathrm{w}) = 0$.

Le symbole m représente le produit de convolution (ou cocontraction), u représente son unité (ou coaffaiblissement), d représente la dérivée en zéro (ou codéréliction).

Les *structures d'interaction différentielles* (SID) sont les SI sur Σ_{Diff}. On leur associe le système de types donné en figure 2.11 et les règles de réduction des figures 2.2 (réduction multiplicative) et 2.12 à 2.14. On pourra trouver dans [ER05] une lecture éclairante de ces nouvelles règles de réduction, en termes d'opérations sur des fonctions analytiques.

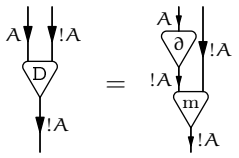
Notons que la réduction de routage (figure 2.14) fait qu'on ne peut plus se restreindre aux structures simples : ces dernières ne sont pas stables par réduction. En revanche, les SID ne comportent pas de boîtes : les résultats de la section 1.4 s'appliquent.

Le lecteur familier avec [BCS06] plutôt qu'avec [ER05] pourra s'étonner de la forme restreinte de la dérivée :

qui ne correspond pas au type attendu pour l'opérateur de dérivation dans une catégorie différentielle

$$\mathrm{D} : A \otimes !A \longrightarrow !A.$$

Ceci provient du fait que l'étude que nous menons concerne particulièrement les règles costructurelles m et u, qui émergent naturellement en présence de sommes sur les morphismes dans les modèles de la logique linéaire. En notant



on retrouve un opérateur de dérivation tel qu'étudié dans [BCS06], à ceci près que nous n'avons pas encore introduit de boîtes de promotion dans les SID. Nous présenterons une notion de réduction impliquant cellules costructurelles, dérivées et boîtes au chapitre 5 : en considérant l'opérateur D que nous venons de définir, le lecteur curieux pourra vérifier qu'on obtient une dynamique en tous points semblable à celle présentée dans la section 4 de [BCS06]. On pourra en particulier voir que la règle de la chaîne pour D ([BCS06, Condition [dC.4]]) se déduit de celle pour ∂ (figure 5.1, page 78).

Par contraste, la notion présentée ici est purement finitaire : on ne considère pas de promotion. En particulier, les modèles des réseaux d'interaction différentiels d'Ehrhard–Regnier ne sont pas nécessairement des catégories différentielles et, inversement, une catégorie différentielle ne suffit pas nécessairement à donner un modèle de ces réseaux : voir la thèse de Carvalho [dC07] pour une sémantique catégorique des réseaux différentiels.

2.5.2 Critère d'acyclicité

Définition 2.29 *Soit* μ *une SID simple (SIDS). On appelle interrupteur de* μ *tout graphe* G *dont les sommets sont les ports de* μ *et tel que :*

- *chaque fil de* μ *est une arête de* G *;*
- *pour chaque cellule* c *de* μ*, de symbole* $\otimes$*,* m*,* d *et* ∂*, il y a une arête entre le port actif de* c *et chacun de ses ports passifs ;*
- *pour chaque cellule* c *de* μ*, de symbole* $\parr$ *ou* c*, il y a une arête entre le port actif de* c *et exactement un de ses ports passifs ;*
- *il n'y a pas d'autre arête hors des trois cas précédents.*

Définition 2.30 *On appelle réseau d'interaction différentiel simple (RIDS) une SIDS dont tous les interrupteurs sont acycliques. On appelle réseau d'interaction différentiel (RID) une SID qui ne contient que des RIDS.*

Lemme 2.31 *L'ensemble des RID est stable par réduction.*

DÉMONSTRATION: Il suffit de vérifier que les réduits d'un RIDS sont tous des RID, par une simple inspection des cas de réduction. □

Théorème 2.32 ([ER05]) *Les RID sont fortement normalisables.*

En fait, Ehrhard et Regnier prouvent ce résultat pour les RID typés dans un système de types faibles, qui généralise à la fois le typage dans les formules de MELL, et le typage pur. Au chapitre 4, on prouvera ce théorème dans le cas plus général des réseaux symétriques, et on verra que l'hypothèse de typage n'est pas nécessaire : la seule raison d'être de cette dernière est d'assurer que toutes les coupures sont réductibles, c'est-à-dire qu'un réseau normal est sans coupure.

2.5.3 Équivalence structurelle

Comme pour les SME, on définit une équivalence structurelle sur les SID. Celle-ci correspond à l'associativité et à la commutativité de la contraction et de la convolution, et à la neutralité de l'affaiblissement (resp. de l'unité) pour la contraction (resp. la convolution).

Définition 2.33 *On définit l'équivalence structurelle* $=_s$ *sur les SIDS comme la clôture contextuelle, réflexive, symétrique et transitive des transformations de la figure 2.15. On l'étend aux SID par :* $\overline{\sum}_{i=1}^{n} \mu_i =_s \overline{\sum}_{i=1}^{n} \nu_i$ *dès que* $\mu_i =_s \nu_i$ *pour tout* i.

2.5.4 λ-calcul avec ressources

L'étude des modèles analytiques de la logique linéaire, à la lumière de la correspondance de Curry–Howard–Girard, a amené Ehrhard et Regnier à introduire le λ-calcul différentiel dans [ER03]. Les RID, cependant, ne comportent pas de boîtes : ils correspondent seulement à une partie finitaire du λ-calcul différentiel, le λ-calcul avec ressources de [ER06b], proche de celui de Boudol [Bou93].

Les termes simples du λ-calcul avec ressources sont donnés par les grammaires mutuellement inductives :

$$\begin{aligned} s &::= x \mid \lambda x\, s \mid \langle s \rangle\, \sigma \\ \sigma &::= 1 \mid s * \sigma \end{aligned}$$

où les objets σ représentent des listes de termes simples, ou hordes. On considère ces listes comme des multiensembles de termes simples en quotientant par la permutativité du produit $*$:

Définition 2.34 *Soit une fonction binaire* $f : X \times Y \longrightarrow Y$. *On dit que* f *est permutative si, pour tous* $x, x' \in X$ *et pour tout* $y \in Y$, $f(x', f(x, y)) = f(x, f(x', y))$.

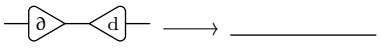


FIGURE 2.13 – Réduction de communication.

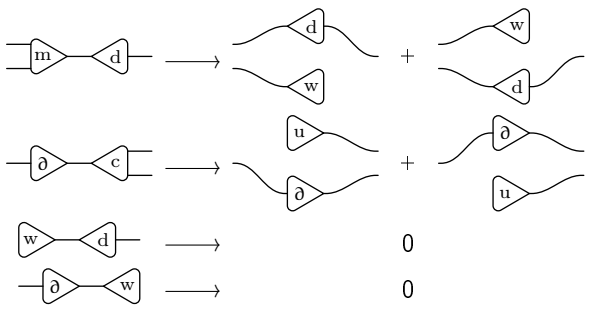


FIGURE 2.14 – Réduction de routage.

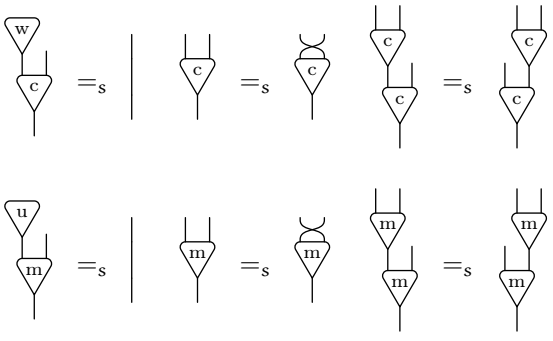


FIGURE 2.15 – Équivalence structurelle des SIDS.

Dire que $*$ est permutative, signifie qu'on identifie $s_1 * (\cdots * (s_n * 1)\cdots)$ avec $s_{\phi(1)} * (\cdots * (s_{\phi(n)} * 1)\cdots)$ pour toute permutation ϕ de $\{1,\ldots,n\}$. L'égalité sur les termes (et les hordes) est donc donnée par l'α-équivalence usuelle et cette permutativité. On notera

$$\sigma = s_1 * (\cdots * (s_n * 1)\cdots) = s_1 * \cdots * s_n = \prod_{i=1}^{n} s_i.$$

Cette notation permet de considérer $*$ comme une opération de monoïde commutatif sur les hordes, avec 1 comme unité, en posant

$$(s_1 * \cdots * s_n) * (s_{n+1} * \cdots * s_{n+p}) = s_1 * \cdots * s_{n+p}.$$

Insistons sur le fait qu'il ne s'agit que d'une notation : le signe $*$ dans $\sigma * \tau$ ne fait pas partie de la syntaxe des hordes, mais désigne une opération binaire, qu'on vient de définir. Les éventuelles ambiguïtés de lecture introduites par cette surcharge sont cependant sans conséquence : quelle que soit la manière de lire une expression formée à partir de termes et de hordes, en utilisant $*$ et 1, on désigne toujours une unique horde de la syntaxe de base.

Les termes du λ-calcul avec ressources sont les sommes finies de termes simples. Formellement, les termes sont générés par la grammaire :

$$S ::= 0 \mid s + S$$

et on les considère modulo permutativité de la somme $+$. On note aussi

$$S = s_1 + (\cdots + (s_n + 0)\cdots) = s_1 + \cdots + s_n = \sum_{i=1}^{n} s_i$$

et on munit l'ensemble des termes d'une structure de monoïde commutatif pour la somme. On s'autorise de la même manière à écrire des sommes de hordes.

Ajoutons enfin que toutes les constructions des termes simples sont linéaires, c'est-à-dire qu'elles commutent avec la somme et 0 : on étend ces constructions à tous les termes par linéarité, en notant par exemple $\lambda x\, 0 = 0$, $(s + t) * \sigma = s * \sigma + t * \sigma$ ou encore $\langle s\rangle\,(\sigma + \tau) = \langle s\rangle\,\sigma + \langle s\rangle\,\tau$. En particulier, l'opération $*$ est distributive sur $+$ pour les hordes. Ceci contraste avec l'habitude selon laquelle l'application d'une fonction à un argument est linéaire en la fonction, mais pas en l'argument.

La réduction est donnée par les deux règles :

$$\begin{aligned} \langle \lambda x\, s\rangle\, 1 &\to s\,[x := 0] \\ \langle \lambda x\, s\rangle\,(t * \sigma) &\to \left\langle \lambda x \left(\frac{\partial s}{\partial x} \cdot t\right)\right\rangle \sigma \end{aligned}$$

où $\frac{\partial s}{\partial x} \cdot t$ est la somme de tous les termes simples obtenus en remplaçant une occurrence de x dans s par t. Par passage au contexte, ces règles définissent bien une relation de réduction des termes, telle que $T \to U$ si et seulement si $T = s + V$ et $U = S' + V$ avec $s \to S'$. Pour le lecteur familier avec le λ-calcul différentiel,

on peut préciser que $\langle s\rangle \prod_{i=1}^{n} s_i$ correspond à $(D^n s \cdot (s_1, \ldots, s_n))\, 0$ qui est un terme simple du λ-calcul différentiel : les règles de réduction précédentes sont alors des cas particuliers de la réduction du λ-calcul différentiel.

Toutefois, afin d'établir une correspondance précise avec la réduction des RID, on considère plutôt une version de $\to$ dite « à grand pas » et notée $\Rightarrow$: celle-ci est définie par passage au contexte de l'unique règle

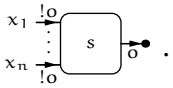
$$\langle \lambda x\, s\rangle\, (t_1 * \cdots * t_n) \Rightarrow \left(\frac{\partial^n s}{\partial x^n} \cdot (t_1, \ldots, t_n)\right) [x := 0].$$

Il est immédiat que $\Rightarrow$ est dans la clôture transitive de $\to$. Par ailleurs, on ne donne pas ici de système de types pour le λ-calcul avec ressources, et on se contente d'une traduction dans les RID purs (RIDP). Notons cependant que tout terme est fortement normalisable : le nombre d'occurrences de variables (en comptant les abstractions) décroît strictement par réduction, au sens de $\to$ comme de $\Rightarrow$.

Définition 2.35 *À un terme simple* s *dont les variables libres sont dans la liste* $X = (x_1, \ldots, x_n)$*, on associe un RIDP* $[\![s]\!]_X$ *d'interface* $(x_1, \ldots, x_n, \bullet)$ *tel que*

$$[\![s]\!]_X \vdash ?i, \ldots, ?i, o$$

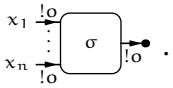
qu'on représente par



De même, à une horde σ *dont les variables libres sont dans* X*, on associe un RIDP* $[\![\sigma]\!]^X$ *d'interface* $(x_1, \ldots, x_n, \bullet)$ *tel que*

$$[\![\sigma]\!]^X \vdash ?i, \ldots, ?i, !o$$

qu'on représente par

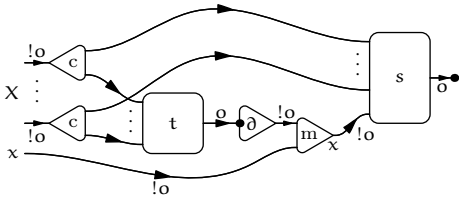


Les règles de traduction sont données en figure 2.16. On étend cette traduction à tous les termes en posant :

$$\left[\!\!\left[\sum_{i=1}^{n} s_i\right]\!\!\right]_X = \sum_{i=1}^{n} [\![s_i]\!]_X .$$

Lemme 2.36 *Pour tous termes* s *et* t *tels que les variables libres de* s *(resp.* t*) soient dans la liste* (X, x) *(resp.* X*) :*

— *le RIDP*



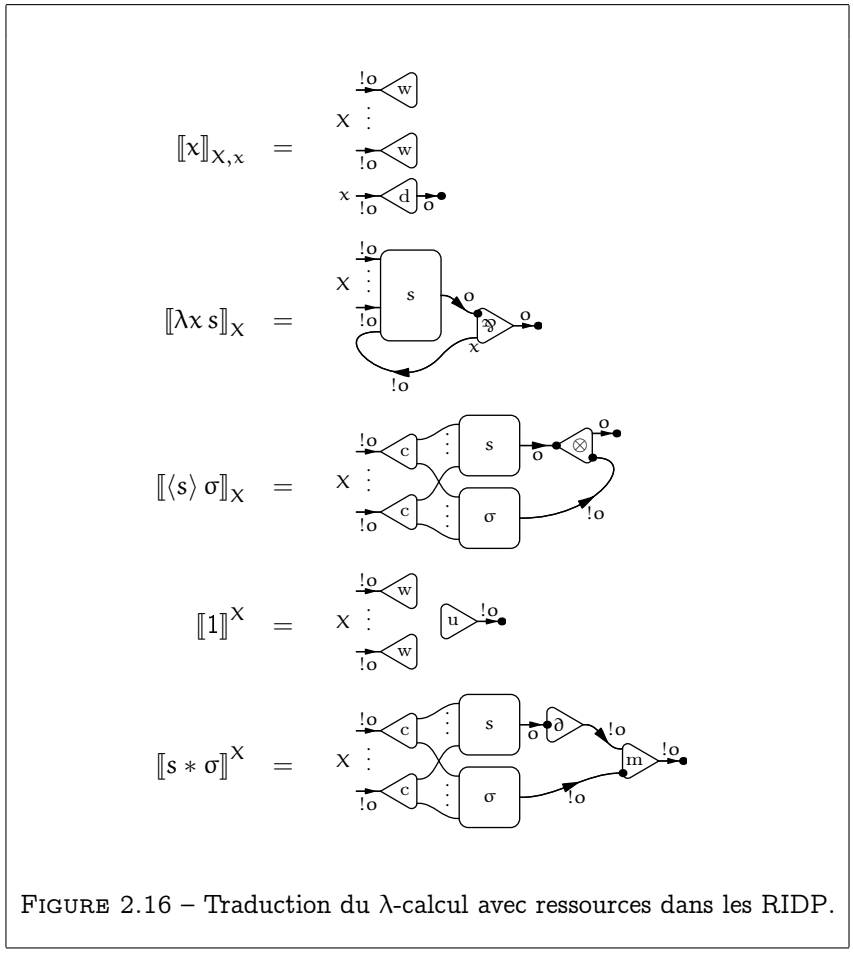
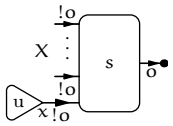


FIGURE 2.16 – Traduction du λ-calcul avec ressources dans les RIDP.

se réduit en au moins une étape en un RIDP structurellement équivalent à $\left[\!\left[\frac{\partial s}{\partial x} \cdot t\right]\!\right]_{X,x}$ *;*

— *le RIDP*



se réduit en au moins une étape en un RIDP structurellement équivalent à $[\![s\,[x := 0]]\!]_X$.

DÉMONSTRATION: Facile par induction sur s. □

Théorème 2.37 *La réduction des RIDP simule la réduction du λ-calcul avec ressources, à équivalence structurelle près : si les variables libres du terme s sont dans la liste X, et si $s \Rightarrow t$, alors il existe un RIDP μ tel que* $[\![s]\!]_X \longrightarrow^{+} \mu =_s [\![t]\!]_X$.

DÉMONSTRATION: Il suffit de réduire la coupure $\langle \otimes , \parr \rangle$ correspondant au redex $\langle \lambda x\, u \rangle\, (v_1 * \cdots * v_n)$ considéré, puis d'utiliser le lemme précédent (n fois le premier item, puis exactement une fois le second). □

La preuve précédente permet de mieux comprendre l'intérêt qu'il y a à considérer la réduction à grand pas $\Rightarrow$: celle-ci correspond à la réduction complète d'un redex avec toute sa horde, en accord avec la traduction dans les RIDP.

Deuxième partie

Réseaux différentiels polarisés

Chapitre 3

Un modèle relationnel pour les réseaux purs

Notation : Si X est un ensemble, on note $\mathcal{M}_{\text{fin}}(X)$ l'ensemble des multiensembles finis $x = [\xi_1, \ldots, \xi_n]$ d'éléments $\xi_1, \ldots, \xi_n \in X$ (en prenant en compte les répétitions). On note $[]$ le multiensemble vide.

3.1 Un objet réflexif dans la catégorie des ensembles et relations

Dans [BEM07], Bucciarelli, Ehrhard et Manzanetto introduisent un modèle du λ-calcul pur basé sur le modèle relationnel de la logique linéaire. Il est bien connu (voir par exemple [LS88]) qu'un modèle du λ-calcul pur s'obtient comme un objet réflexif dans une catégorie cartésienne fermée (CCC). Il est également classique qu'un modèle de la logique linéaire produit une CCC par la construction de co-Kleisli sur la comonade associée à l'exponentielle ($!$).

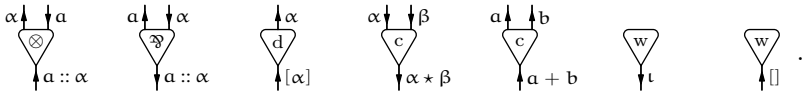
Le modèle relationnel multiensembliste de la logique linéaire est donné par la catégorie Rel, dont les objets sont les ensembles et les morphismes sont les relations binaires, avec comme composition la composition habituelle des relations : si $f : A \longrightarrow B$ et $g : B \longrightarrow C$ alors $g \circ f : A \longrightarrow C$ est définie par

$$(x, z) \in g \circ f \iff \exists y \in B, (x, y) \in f \text{ et } (y, z) \in g.$$

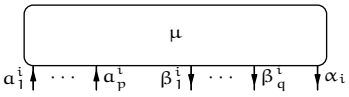
Munie du produit des ensembles comme produit tensoriel et d'un singleton $\bot = \{*\}$ comme objet dualisant, Rel est une catégorie $*$-autonome, dont la dualité ne fait que renverser le sens des flèches (et dont l'unité tensorielle $1 = \bot$). Elle a des produits finis, donnés par l'union disjointe ensembliste. Pour tout objet A, on peut définir le $\otimes$-comonoïde cocommutatif libre $!A$ généré par A, en munissant $\mathcal{M}_{\text{fin}}(A)$ des opérations évidentes.

On obtient ainsi une catégorie de Lafont et donc un modèle de LL ([Bie93]). La co-Kleisli associée à la comonade $!$ est une CCC : c'est la catégorie MRel dont les objets sont les ensembles et telle que $\text{MRel}(A, B) = \mathcal{M}_{\text{fin}}(A) \times B$ (la

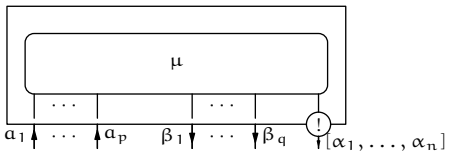
Pour tous $\alpha, \beta \in \mathcal{D}$ et tous $a, b \in \mathcal{M}_{\text{fin}}(\mathcal{D})$, on type



Si, pour tout $i \in \{1, \ldots, n\}$,



alors



avec $a_j = \sum_{i=1}^{n} a_j^i$ et $\beta_k = \prod_{i=1}^{n} \beta_k^i$.

FIGURE 3.1 – Règles de typage relationnel pour les SIP.

composition des flèches se déduit de ce qui précède ; voir [BEM07, Section 4] pour une définition directe).

On explicite maintenant la construction de l'objet réflexif de [BEM07]. On note $\mathcal{M}_{\text{fin}}(X)^{(\omega)}$ l'ensemble des suites $\xi = (\xi(i))_{i\in\omega}$ de multiensembles finis d'éléments de X tels que $\xi(i) = []$ pour presque tout $i \in \omega$.

Définition 3.1 *On définit une famille croissante $(\mathcal{D}_n)_{n\in\mathbb{N}}$ d'ensembles par : $\mathcal{D}_0 = \emptyset$ et $\mathcal{D}_{n+1} = \mathcal{M}_{fin}(\mathcal{D}_n)^{(\omega)}$. On écrit alors $\mathcal{D} = \bigcup_{n\in\mathbb{N}} \mathcal{D}_n$.*

Si $a \in \mathcal{M}_{fin}(\mathcal{D})$ et $\alpha \in \mathcal{D}$, on note $a :: \alpha$ la suite β telle que $\beta(0) = a$ et $\beta(i+1) = \alpha(i)$ pour tout $i \in \omega$. On note ι la suite constante telle que $\iota(i) = []$ pour tout $i \in \omega$.

On a par exemple $\mathcal{D}_1 = \{\iota\}$. L'opération $(a, \alpha) \mapsto a :: \alpha$ est clairement une bijection entre $\mathcal{D}$ et $\mathcal{M}_{\text{fin}}(\mathcal{D}) \times \mathcal{D}$, et vérifie $\iota = [] :: \iota$. Cette bijection fait de $\mathcal{D}$ un objet réflexif extensionnel dans MRel : on obtient un modèle extensionnel du λ-calcul. Dans la suite, on montre que cet objet fournit également un modèle commun aux SIP et SID pures.

On appelle *types relationnels* les éléments de $(\mathcal{D} \cup \mathcal{M}_{\text{fin}}(\mathcal{D})) \times \{0, 1\}$, avec la négation $(\tau, 0) \overset{\perp}{\longleftrightarrow} (\tau, 1)$ si $\tau \in \mathcal{D} \cup \mathcal{M}_{\text{fin}}(\mathcal{D})$. On convient de toujours orienter les fils typés vers le port de type $(\tau, 0)$ et on ne fait figurer que l'étiquette τ.

3.2 Interprétation des réseaux polarisés

On impose une structure de monoïde commutatif sur $\mathcal{D}$ de la manière suivante.

Définition 3.2 *Pour tous $\alpha, \beta \in \mathcal{D}$, on note $\alpha \star \beta$ la suite telle que, pour tout $i \in \omega$, $(\alpha \star \beta)(i) = \alpha(i) + \beta(i)$ où $+$ dénote l'union des multiensembles.*

On récupère les propriétés de l'union des multiensembles : $(\mathcal{D}, \star, \iota)$ est un monoïde commutatif, intègre et simplifiable.

On donne un système de types relationnels pour les SIP par les règles de la figure 3.1. Il est clair qu'une SIP relationnellement typée admet un typage pur : il suffit de remplacer les types :

$$\begin{aligned}(\alpha, 0) &\rightsquigarrow o\\ (\alpha, 1) &\rightsquigarrow i\\ (a, 0) &\rightsquigarrow ?i\\ (a, 1) &\rightsquigarrow !o.\end{aligned}$$

On vérifie de plus que le système de réduction est compatible avec le typage. On peut faire mieux :

Définition 3.3 *Soit μ une SIP et I une interface de μ. On note*

$$\mu^I : \tau_1, \ldots, \tau_n \vdash \sigma_1, \ldots, \sigma_m$$

si

$$\mu^I \vdash (\tau_1, 1), \ldots, (\tau_n, 1), (\sigma_1, 0), \ldots, (\sigma_m, 0).$$

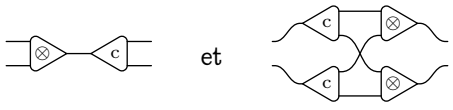
Lemme 3.4 *Pour toutes SIP μ et ν telles que $\mu \longrightarrow \nu$, et toute interface I de μ, on a*

$$\mu^I : \tau_1, \ldots, \tau_n \vdash \sigma_1, \ldots, \sigma_m$$

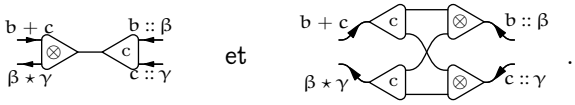
si et seulement si

$$\nu^I : \tau_1, \ldots, \tau_n \vdash \sigma_1, \ldots, \sigma_m.$$

DÉMONSTRATION: Par contextualité, il suffit de le vérifier quand μ est une coupure purement typée et ν est son réduit. Par exemple, comme $a :: \alpha = (b :: \beta) \star (c :: \Gamma)$ si et seulement si $a = b + c$ et $\alpha = \beta \star \gamma$, les deux SIP



admettent exactement les mêmes types, tous de la forme



.

□

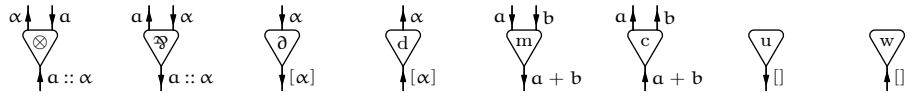


FIGURE 3.2 – Règles de typage relationnel pour les SIDS.

Définition 3.5 *Soit* μ^I *une SIP avec interface. On appelle sémantique relationnelle de* μ^I *et on note* $[\![\mu]\!]_I$ *l'ensemble*

$$\left\{((\tau_1,\ldots,\tau_n),(\sigma_1,\ldots,\sigma_m));\ \mu^I : \tau_1,\ldots,\tau_n \vdash \sigma_1,\ldots,\sigma_m\right\}.$$

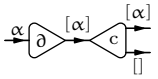
Par le lemme précédent, la sémantique relationnelle est préservée par réduction.

Ce résultat ne devrait pas surprendre le lecteur familier avec LLP. En effet, non seulement $\mathcal{D}$ est un objet réflexif dans MRel, mais c'est aussi, une fois muni de sa structure de monoïde, un objet réflexif dans la catégorie de contrôle des $\parr$-monoïdes $\mathbb{K}_{\mathsf{Rel}}$, telle que définie dans [LR03].

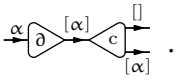
Rappelons que si $\mathcal{C}$ est une catégorie de Lafont, et si ! est la comonade associée à la formation des $\otimes$-comonoïdes cocommutatifs colibres, alors $\mathbb{K}_{\mathcal{C}}$ est la catégorie dont les objets sont les $\parr$-monoïdes commutatifs de $\mathcal{C}$ et dont les morphismes sont définis par $\mathbb{K}_{\mathcal{C}}(M,N) = \mathcal{C}(!M,N)$. La composition est définie comme pour la co-Kleisli. Le théorème 3.1 de [LR03] établit que $\mathbb{K}_{\mathcal{C}}$ est une catégorie de contrôle, et fournit donc un modèle de LLP (ainsi qu'un modèle du $\lambda\mu$-calcul d'après [Sel01]). Dans cette sémantique, les formules négatives sont interprétées par des $\parr$-monoïdes et, dualement, les formules positives sont interprétées par des $\otimes$-comonoïdes.

3.3 Interprétation des réseaux différentiels

On peut aussi donner un système de types relationnels pour les SIDS, par les règles de la figure 3.2. Encore une fois, on vérifie qu'une SIDS relationnellement typée admet un typage pur. Par contre, la réduction de routage n'est pas compatible avec le typage relationnel sur les SID. Considérons par exemple la coupure $\langle \partial\ ,\ c \rangle$, qu'on peut typer :

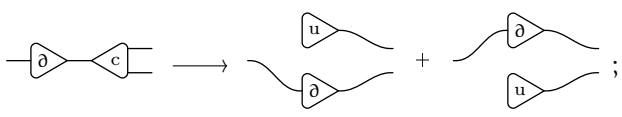


ou

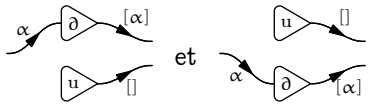


.

Rappelons que :



or ce réduit n'est pas typable, car les deux SIDS qui le composent n'ont pas les mêmes types. Remarquons en effet que :



sont les seules formes de typage possibles. On retrouve bien les typages possibles du redex dans les SIDS du réduit, mais séparément. Ceci suggère la définition suivante.

Définition 3.6 *Soit* μ *une SIDS et* I *une interface de* μ. *On note*

$$\mu^I : \tau_1, \ldots, \tau_n \vdash \sigma_1, \ldots, \sigma_m$$

si

$$\mu^I \vdash (\tau_1, 1), \ldots, (\tau_n, 1), (\sigma_1, 0), \ldots, (\sigma_m, 0).$$

On appelle sémantique relationnelle de μ^I *et on note* $[\![\mu]\!]_I$ *l'ensemble*

$$\left\{((\tau_1, \ldots, \tau_n), (\sigma_1, \ldots, \sigma_m));\ \mu^I : \tau_1, \ldots, \tau_n \vdash \sigma_1, \ldots, \sigma_m\right\}.$$

On étend cette définition à toutes les SID : si $\mu = \overline{\sum}_{i=1}^{n} \mu_i$ *alors on pose* $[\![\mu]\!]_I = \bigcup_{i=1}^{n} [\![\mu_i]\!]_I$

Lemme 3.7 *Si* $\mu \longrightarrow \nu$ *alors* $[\![\mu]\!]_I = [\![\nu]\!]_I$.

DÉMONSTRATION: Il suffit de vérifier que si μ est simple et $\nu = \overline{\sum}_{i=1}^{n} \nu_i$ alors

$$\mu^I : \tau_1, \ldots, \tau_n \vdash \sigma_1, \ldots, \sigma_m$$

si et seulement s'il existe $i \in \{1, \ldots, n\}$ tel que

$$\nu_i^I : \tau_1, \ldots, \tau_n \vdash \sigma_1, \ldots, \sigma_m.$$

C'est facile à voir pour les cas où μ est un redex, et le résultat s'étend par contextualité. □

Encore une fois, ce résultat rentre dans un cadre catégorique plus général :

(a) Rel est une catégorie $*$-autonome (avec $\otimes = \times$, $A^\perp = A$ et $I = \perp = \{*\}$) enrichie sur les monoïdes commutatifs (on prend pour somme des morphismes la réunion). Si $f : A \longrightarrow B$ est un morphisme dans Rel, $f^\perp : B \longrightarrow A$ est donné par

$$f^\perp = \{(\beta, \alpha);\ (\alpha, \beta) \in f\}.$$

(b) Pour tout objet X, $!A$ est une *bigèbre* : en posant

$$\mu_A = \Delta_A^\perp = \{((a, b), a + b);\ a, b \in !A\} : !A \otimes !A \longrightarrow !A$$

et

$$\eta_A = \epsilon_A^\perp = \{(*, [])\} : I \longrightarrow !A$$

$(!A, \mu_A, \eta_A)$ est un monoïde commutatif, $(!A, \Delta_A, \epsilon_A)$ est un comonoïde cocommutatif, et μ_A et η_A sont des morphismes de comonoïdes (ou, de manière équivalente, Δ_A et ϵ_A sont des morphismes de monoïdes). [1]

1. Ceci correspond à la commutation des diagrammes page 4.

(c) En posant

$$\mathrm{der}_A = \mathrm{cod}_A^{\perp} = \{([\alpha], \alpha);\ \alpha \in A\} : !A \longrightarrow A$$

on a

$$\begin{aligned} \mathrm{der}_A \eta_A &= \emptyset \\ \epsilon_A \mathrm{cod}_A &= \emptyset \\ \mathrm{der}_A \mu_A &= (\mathrm{der}_A \times \epsilon_A) \cup (\epsilon_A \times \mathrm{der}_A) \\ \Delta_A \mathrm{cod}_A &= (\mathrm{cod}_A \times \eta_A) \cup (\eta_A \times \mathrm{cod}_A \\ \mathrm{der}_A \mathrm{cod}_A &= A. \end{aligned}$$

D'après [dC07], ceci définit un modèle des réseaux différentiels. En effet : la clause (a) permet d'interpréter le fragment multiplicatif de la logique linéaire ; la clause (b) permet d'interpréter la convolution m, l'unité u, la contraction c et l'affaiblissement w, en modélisant la réduction structurelle (figure 2.12) ; la clause (c) permet d'interpréter la dérivée ∂ et la déréliction d en modélisant les réductions de routage et de communication (figures 2.13 et 2.14).

3.4 Vers des réseaux différentiels polarisés

Les structures d'interaction ont permis de présenter réseaux polarisés et réseaux différentiels dans un cadre formel commun. On a de plus exhibé un modèle dénotationnel pour les variantes pures de ces réseaux, à partir du modèle relationnel de la logique linéaire.

Partant, diverses voies sont offertes pour explorer les rapports qu'entretiennent, d'une part, les notions associées à la logique classique (réseaux polarisés, $\lambda\mu$-calcul, $\bar{\lambda}\mu$-calcul, et leurs sémantiques) et, d'autre part, les notions associées à l'analyse différentielle des preuves (réseaux différentiels, λ-calcul différentiel, λ-calcul avec ressources, et leurs modèles).

3.4.1 Compatibilité et interaction

Il est possible de considérer des SID avec boîtes, et de donner (en les déduisant par exemple de [Ehr05]) des règles de réduction pour les coupures $\langle \partial\ ,\ b^n \rangle$, $\langle \mathrm{m}\ ,\ b^n \rangle$ et $\langle \mathrm{u}\ ,\ b^n \rangle$, où b est une cellule boîte et n est un numéro de port auxiliaire ($1 \leq n \leq \bar{a}(b)$). Ces règles font partie du folklore entourant les réseaux différentiels. Toutefois, elles n'ont jamais été publiées en tant que telles à notre connaissance. On les donnera explicitement au chapitre 5, (figures 5.1 et 5.2).

On peut alors considérer le système obtenu en réunissant réseaux polarisés et réseaux différentiels : sur la signature $(\Sigma_{\mathrm{LL}} \cup \Sigma_{\mathrm{Diff}})^{[!]}$, les règles de typage sont celles des SIP et celles des SID restreintes aux formules polarisées, et les règles de réduction sont la réunion de celles des SIP et celles des SID avec boîtes. Dans ce système, aucun nouveau type de coupure n'est introduit : les extensions polarisées des règles structurelles de LLP et les cellules costructurelles des réseaux différentiels ne se « voient » pas, à cause du typage.

On verra dans la troisième partie de ce mémoire que ce système correspond au λμ-calcul différentiel introduit dans [Vau07b], qui est une extension commune au λ-calcul différentiel d'Ehrhard–Regnier et au λμ-calcul de Parigot. Cette relation témoigne d'une compatibilité entre les traits de logique classique et les moyens d'analyse différentielle des preuves. En tant qu'extensions orthogonales de la logique linéaire, l'introduction de règles structurelles généralisées et celle des règles costructurelles et de la dérivée sont compatibles justement parce qu'elles n'ont pas l'occasion d'interagir.

Ceci ne signifie toutefois pas que les règles costructurelles sont insensibles à la polarisation. Cédons ici à la tentation de la métaphore : si le système esquissé ci-dessus correspond à la somme de la polarisation et de l'extension différentielle (somme directe, puisqu'on a vu que ces concepts étaient orthogonaux), alors il est légitime de se demander quel en est le produit. Avant d'avancer une réponse à cette interrogation, nous proposons la synthèse suivante, sous forme de slogans :

(a) L'extension polarisée de la logique linéaire revient à étendre la structure de $\otimes$-comonoïde des formules *bien sûr* !N (resp. la structure de $\parr$-monoïde formules *pourquoi pas* ?P) à toutes les formules positives (resp. négatives).

(b) L'extension différentielle de la logique linéaire revient à :
   — étendre la structure de $\otimes$-comonoïde des formules *bien sûr* !N (resp. la structure de $\parr$-monoïde des formules *pourquoi pas* ?P) en une structure de $\otimes$-bigèbre (resp. $\parr$-bigèbre) ;
   — introduire la dérivée, comme duale de la déréliction.

Notons que le premier item du slogan (b) ne s'appuie pas sur le second : pour les besoins de la discussion et faute d'un meilleur vocable, on qualifie d'extension algébrique ce premier point. Cette extension algébrique est un préliminaire nécessaire à l'extension différentielle : sans la convolution ni son unité, on ne saurait définir le comportement de la dérivée. Ce constat est très visible dans la relecture catégorique des modèles du λ-calcul différentiel fournie par [BCS06], et c'est une articulation essentielle du travail de Fiore dans [Fio07] : les règles costructurelles émergent naturellement à partir de la somme des morphismes, et c'est dans ce cadre que se définit l'interprétation de la dérivée.

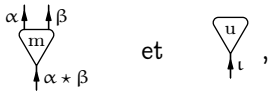
En résumé, la polarisation étend le champ des règles structurelles et l'algébrisation leur donne des symétriques. Nous proposons le point de vue suivant : le produit de la polarisation et de l'analyse différentielle des preuves réside dans une extension polarisée des règles costructurelles, c'est-à-dire dans l'introduction d'une structure de $\otimes$-bigèbre sur les types positifs. On consacre le reste de cette section à préciser ce concept.

3.4.2 Bigèbres sur les types polarisés

Partant des SI sur $(\Sigma_{\mathrm{LL}} \cup \Sigma_{\mathrm{Diff}})^{[!]}$, munies des règles de typage et de réduction des SIP et des SID avec boîtes, posons comme fil conducteur le relâchement du typage des cellules m et u aux formules positives :

.

La réduction structurelle des SID (figure 2.12, page 48) est encore compatible avec ce typage. En étendant la sémantique relationnelle par symétrie :

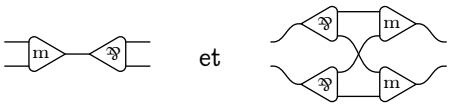


la reduction structurelle correspond à la structure de bigèbre de $\mathcal{D}$, dont la multiplication et la comultiplication sont données par $\star$, et l'unité et la coünité par ι.

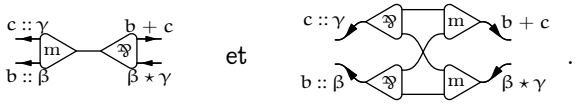
Le relâchement du typage introduit également deux nouvelles coupures typables : $\langle m\,,\,⅋\rangle$ et $\langle u\,,\,⅋\rangle$. Comment définir leurs réduits ? Rappelons que le ⅋ se type relationnellement :

.

Or on a $a :: \alpha = (b :: \beta) \star (c :: \gamma)$ si et seulement si $\alpha = \beta \star \gamma$ et $a = b + c$. C'est-à-dire que les réseaux

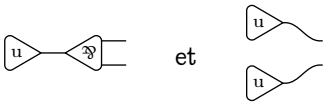


sont identifiés par la sémantique, vu qu'on type génériquement



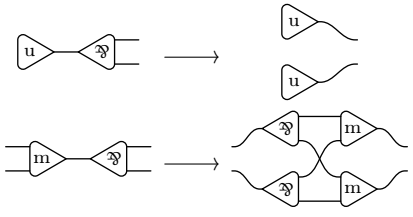
.

De la même manière, on voit que

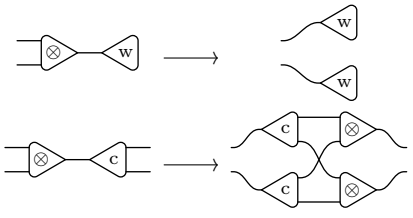


ont la même sémantique.

On propose donc les règles de réduction supplémentaires

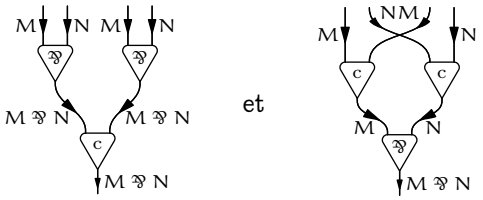


qui font écho aux règles symétriques dans les SIP

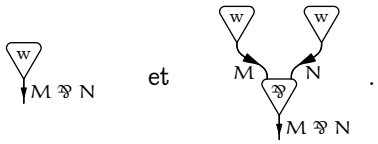


.

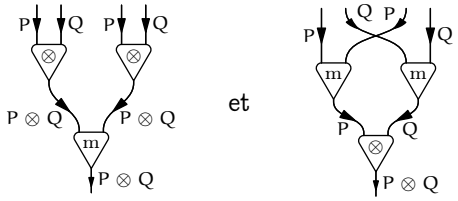
Remarquons d'ailleurs qu'on peut lire ces dernières règles comme une réalisation du fait que



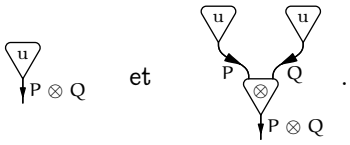
ont le même comportement dans la réduction, de même que



Cela reflète le point de vue sémantique selon lequel la structure de ⅋-monoïde sur $M \parr N$ se déduit canoniquement de celle sur M et N. De manière analogue, les règles de réduction qu'on vient d'introduire munissent



de la même dynamique, ainsi que



On obtient ainsi des réseaux différentiels polarisés, dont la réduction est suggérée à la fois par des considérations sur la nature calculatoire de la polarisation, et par des moyens sémantiques. Dans le chapitre suivant, on en donne d'abord une version finitaire, qui correspond à l'extension par polarisation des SID sans boîtes. On présente ensuite le système complet avec boîtes de promotion, au chapitre 5.

Chapitre 4

Réseaux symétriques

On définit une extension des réseaux différentiels d'Ehrhard–Regnier, par une polarisation à la Laurent. Le système obtenu conserve la symétrie des réseaux différentiels :

— la dualité associée à la négation linéaire échange $\parr$ et $\otimes$, cellules structurelles et costructurelles, déréliction et dérivée ;

— les règles de réduction, déduites du chapitre précédent, sont symétriques selon cette dualité.

La seule notion qui brise cette symétrie est la correction : comme dans [ER05] on introduit un critère à la Danos–Regnier, qui assure la normalisation forte, sans condition de typage.

4.1 Structures symétriques

On appelle structures symétriques simples (SSS) les SIS sur Σ_{Diff} et *structures symétriques* (SS) les SI correspondantes, lorsqu'elles sont associées au système de typage donné en figure 4.1, et au système de réduction donné par la réunion des règles de réduction des RID :

— réduction multiplicative (figure 2.2, page 33) ;

— réduction structurelle (figure 2.12, page 48) ;

— réduction de communication (figure 2.13, page 51) ;

— réduction de routage (figure 2.14, page 51) ;

et des règles de la figure 4.2. La réduction est complète pour le typage, et confluente au sens de la section 1.4 (page 26).

On note $\xrightarrow{\text{struct}}$ (resp. $\xrightarrow{\text{comm}}$, $\xrightarrow{\text{route}}$) la relation de réduction restreinte aux règles de réduction structurelle (resp. de communication, de routage). On appelle *réduction simple*, notée $\xrightarrow{\text{simple}}$, le système restreint aux figures 2.2, 2.12 et 4.2. On appelle *réduction d'effacement*, notée $\xrightarrow{\text{eff}}$, le système réduit aux trois premières règles de la figure 2.12 et aux première et troisième règles de la figure 4.2.

On observe immédiatement que seul le typage sépare les cellules duales ($\otimes/\parr$, $\mathrm{m/c}$, $\mathrm{u/w}$, ∂/d), dont les interactions sont rigoureusement symétriques :

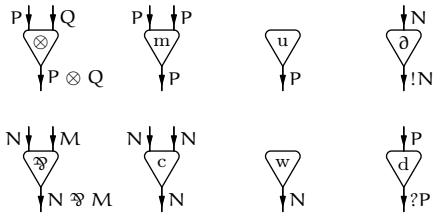


FIGURE 4.1 – Règles de typage des structures symétriques.

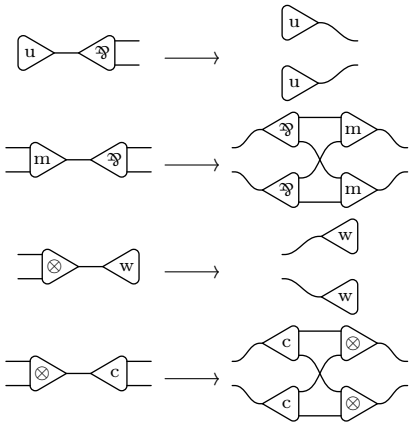


FIGURE 4.2 – Réduction polarisée des SSS.

on pourrait donc se contenter de quatre symboles au lieu des huit présentés, en fusionnant les symboles duaux deux-à-deux. Dans un cadre typé, on obtiendrait une notion de réduction identique. Cette remarque d'Yves Lafont vaut également pour les SID d'Ehrhard et Regnier.

On s'abstient toutefois de procéder à une telle identification : comme annoncé plus haut, on introduit dans la suite un critère de correction purement géométrique, qui assure la normalisation forte de la réduction, y compris dans un cadre non typé ; or ce critère donne des rôles très différents aux symboles duaux.

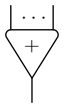
On n'a par contre pas introduit de symboles distincts pour les unités multiplicatives 1 et $\bot$: si on ajoutait des constantes 1 et $\bot$ aux formules de la logique linéaire, les règles correspondantes se comporteraient respectivement comme des cellules u sur le type (positif) 1 et w sur le type (négatif) $\bot$, et ce à la fois dans la réduction, pour la correction ou la séquentialité, et dans la sémantique (en tout cas pour la sémantique relationnelle que nous venons de présenter).

4.2 Réduction simple

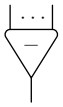
La réduction simple est la réduction qui ne fait pas intervenir les cellules déréliction (d) et dérivée (∂). On étudie les effets de la réduction simple d'un point de vue global, en considérant des arbres de cellules.

Définition 4.1 *On appelle arbre positif (resp. négatif ; de convolution ; de contraction) un arbre construit sur les symboles* $\otimes$, m *et* u *(resp.* $\parr$, c *et* w *;* m *et* u *;* c *et* w*), selon la définition 1.38 (page 27).*

On représente génériquement un arbre positif par

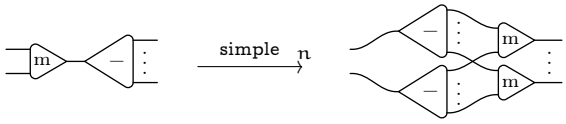


et un arbre négatif par

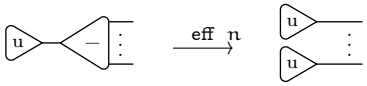


.

Lemme 4.2 *Par réduction simple (resp. d'effacement), le produit (resp. l'unité) duplique (resp. efface) un arbre négatif :*



et

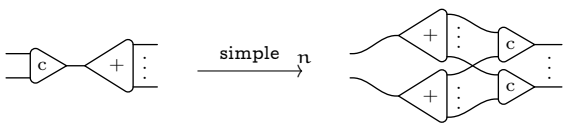


où n *est le nombre de cellules de l'arbre négatif.*

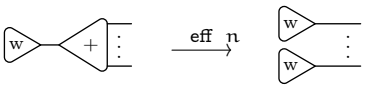
DÉMONSTRATION: Ces résultats sont aisément établis par induction sur l'arbre négatif. □

Le lemme suivant est une version duale du précédent et se démontre de la même manière :

Lemme 4.3 *Par réduction simple (resp. d'effacement), la contraction (resp. l'affaiblissement) duplique (resp. efface) un arbre positif :*



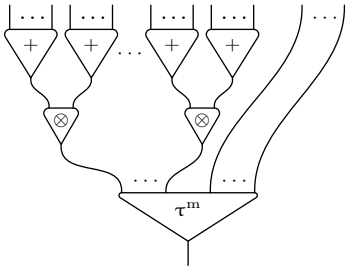
et



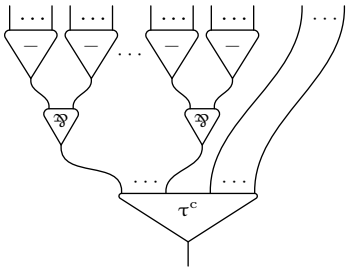
où n *est le nombre de cellules de l'arbre positif.*

On établit des résultats similaires, pour les cellules multiplicatives $\otimes$ et $\parr$.

Lemme 4.4 *La forme générale d'un arbre positif est :*



où τ^{m} *est un arbre de convolution. De même, la forme générale d'un arbre négatif est :*



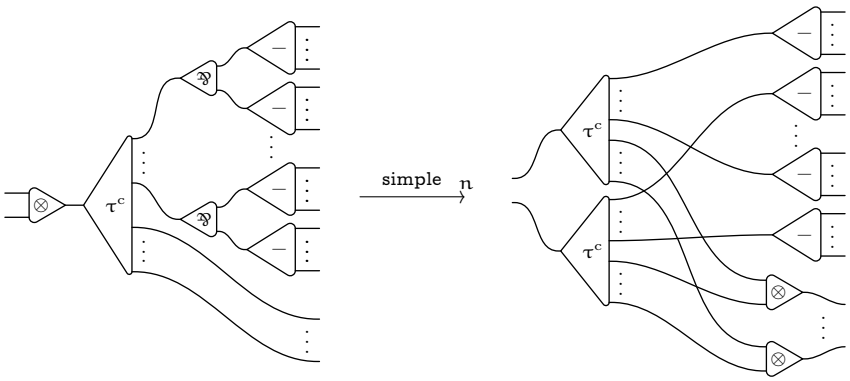
où τ^{c} *est un arbre de contractions.*

DÉMONSTRATION: Facile par induction. [1] □

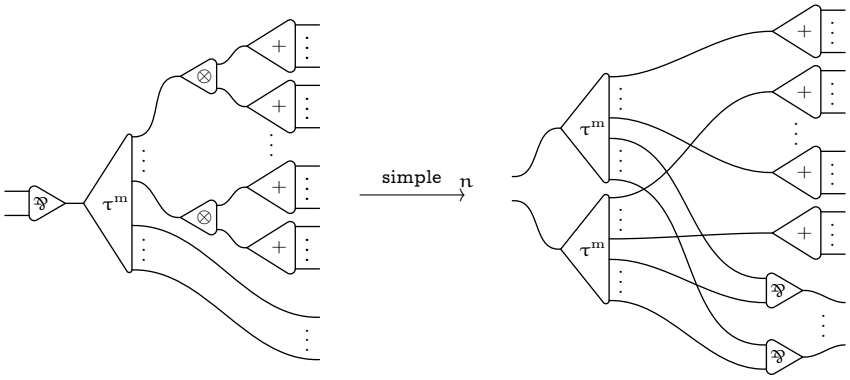
Lemme 4.5 *Par réduction simple, la SSS obtenue en coupant une cellule* $\otimes$ *(resp.* $\parr$*) sur un arbre négatif (resp. positif) se réduit en une SSS sans*

1. On utilise en particulier le fait que la représentation générique des arbres peut cacher des croisements de fils. Il ne s'agit pas ici de faire intervenir une notion d'équivalence structurelle : on n'a simplement pas imposé d'ordre particulier sur les ports passifs dans la définition des arbres (définition 1.38).

coupure. Plus précisément, avec les notations du lemme précédent :



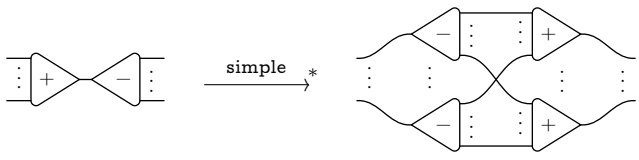
et



où n *est le nombre de cellules de* τ^c *ou de* τ^m.

DÉMONSTRATION: La preuve est une simple induction sur τ^m ou τ^c. □

Lemme 4.6 *Par réduction simple, la SSS obtenue en coupant un arbre positif sur un arbre négatif, se réduit en une SSS sans coupure. Plus précisément :*



ou les arbres positifs (resp. négatifs) représentés ne sont pas nécessairement tous identiques.

DÉMONSTRATION: Ce résultat est établi par induction sur, par exemple, l'arbre positif, en utilisant le lemme 4.5 ou le lemme 4.2 à chaque pas d'induction, suivant que la cellule racine est un ⊗ ou non. □

4.3 Normalisation des réseaux symétriques

On reproduit à l'identique la définition 2.30 des RID.

Définition 4.7 *On appelle réseau symétrique simple (RSS) une SSS dont tous les interrupteurs sont acycliques. On appelle réseau symétrique (RS) une SS qui ne contient que des RSS.*

Évidemment, un arbre est toujours un RSS. L'exemple canonique de RSS qui n'est pas un arbre est l'identité linéaire :  .

Lemme 4.8 *L'ensemble des RS est stable par réduction.*

DÉMONSTRATION: Il suffit de vérifier que les réduits d'un RSS sont tous des RS, ce qui se vérifie par une simple inspection des cas de réduction. □

On démontre la normalisation forte des RS, indépendamment de tout typage. La preuve est adaptée de celle de [ER05] : on donne une stratégie de réduction qui normalise, puis on en déduit la normalisation forte via la confluence forte. Cette dernière étape est cassée dans la preuve originale : on a vu que la confluence forte n'était pas vérifiée en général. On répare ce défaut grâce au théorème 1.37 (page 27).

On démontre d'abord la normalisation forte de la réduction simple.

Lemme 4.9 *La réduction simple normalise fortement sur les RS.*

DÉMONSTRATION: Comme les règles de la réduction simple préservent les SIS, on a $\xrightarrow{\text{simple}} \subset \multimap$, et plus précisément $\xrightarrow{\text{simple}} = \xrightarrow[\bullet]{\text{simple}}$. Donc, par le théorème 1.37, il suffit de montrer la normalisation faible. On note $\|\mu\|$ le nombre de coupures simples dans le RSS μ. On démontre par induction sur $\|\mu\|$ que $\mu \xrightarrow{\text{simple}}{}^{*} \mu'$, où μ' est un RSS sans coupures simples.

Si $\|\mu\| = 0$, $\mu' = \mu$. Sinon, soit $\langle c^+ , c^- \rangle$ une coupure simple de μ, où c^+ est une cellule positive et c^- une cellule négative. On note alors τ^+ l'arbre positif maximal enraciné en c^+ et τ^- l'arbre négatif maximal enraciné en c^-. Par le lemme 4.6, $\mu \xrightarrow{\text{simple}}{}^{*} \mu'$ où μ' est obtenu en remplaçant $\langle \tau^+ , \tau^- \rangle$ par un réseau sans coupure, dans lequel les ports passifs de τ^+ (resp. τ^-) deviennent les ports principaux d'arbres négatifs (resp. positifs). En vertu de la correction de μ, il n'y a pas de fil entre un port passif de τ^+ et un port passif de τ^- : par maximalité de τ^+ et τ^-, $\|\mu'\| = \|\mu\| - 1$ et on conclut par l'hypothèse d'induction.

Le résultat s'étend trivialement aux RS. □

Définition 4.10 *On introduit une stratégie de réduction, comme une restriction de $\longrightarrow^{*}$. On pose $\nu \longrightarrow\!\!\!\!\!\rightarrow \nu'$ si l'un des cas suivants s'applique :*

— *ν est un RSS contenant au moins une coupure simple, et ν' est sa forme normale par réduction simple ;*

— ν *est un RSS ne contenant pas de coupure simple et il existe un* ν'' *tel que* $\nu \xrightarrow{\text{comm}} \nu''$ *et* ν' *soit la forme normale par réduction simple de* ν'' *;*
— ν *est un RSS ne contenant pas de coupure simple et il existe un* $\nu'' \neq 0$ *tel que* $\nu \xrightarrow{\text{route}} \nu''$ *et* ν' *soit la forme normale par réduction simple de* ν'' *;*
— $\nu = \overline{\sum}_{i=1}^{n} \nu_i$ *et* $\nu' = \sum_{i=1}^{n} \nu'_i$ *où*
  — $\nu_i = \nu'_i$ *si* ν_i *est normal,*
  — $\nu_i \longrightarrow\!\!\!\!\!\rightarrow \nu'_i$ *sinon,*
  — *et les* ν_i *ne sont pas tous en forme normale.*

Lemme 4.11 *La relation* $\longrightarrow\!\!\!\!\!\rightarrow$ *est dans la fermeture transitive de* $\multimap\!\!\bullet\!\!\rightarrow$*. De plus, un RS* μ *est en forme normale pour* $\multimap\!\!\bullet\!\!\rightarrow$ *si et seulement s'il est en forme normale pour* $\longrightarrow\!\!\!\!\!\rightarrow$*.*

DÉMONSTRATION: C'est direct d'après la définition. □

On définit maintenant une « hauteur » sur les SS, qui décroît strictement suivant $\longrightarrow\!\!\!\!\!\rightarrow$.

Définition 4.12 *Soit* μ *une SSS. On note* $|\mu|_\partial$ *le nombre de cellules* ∂ *dans* μ*, et* $|\mu|_{m,c}$ *le nombre de cellules* m *et* c *dans* μ*.*

On se contente de compter les cellules ∂, et non les cellules d : la seule EER qui modifie $|\mu|_\partial$ est la réduction de communication, et cette dernière est symétrique en d et ∂.

Lemme 4.13 *Si* μ *est une SSS et* $\mu \xrightarrow{\text{simple}} \mu'$ *alors* $|\mu'|_\partial = |\mu|_\partial$*.*

DÉMONSTRATION: La réduction simple préserve les SSS et ne touche pas les cellules ∂. □

Lemme 4.14 *Si* μ *est un RSS normal pour la réduction simple,* $\mu \xrightarrow{\text{route}} \mu'$*,* ν' *est un RSS de* μ' *et* ν'' *est la forme normale de* ν' *par réduction simple, alors* $|\nu''|_{m,c} \leq |\nu'|_{m,c} = |\mu|_{m,c} - 1$*.*

DÉMONSTRATION: L'égalité $|\nu'|_{m,c} = |\mu|_{m,c} - 1$ est directe. Pour l'inégalité $|\nu''|_{m,c} \leq |\nu'|_{m,c}$, il suffit d'observer que les seules coupures simples éventuellement créées dans ν' sont des coupures d'effacement. □

Définition 4.15 *Pour toute SSS* μ*, on définit*

$$h(\mu) = \begin{cases} (0,0) & \text{si } \mu \text{ est en forme normale,} \\ (|\mu|_\partial, |\mu|_{m,c}) & \text{sinon.} \end{cases}$$

Si maintenant $\mu = \overline{\sum}_{i=1}^{n} \mu_i$ *est une SS, on note* $h(\mu) = \max\{h(\mu_i); 1 \leq i \leq n\}$ *où* $\mathbb{N}^2$ *est ordonné lexicographiquement.*

Lemme 4.16 *Si* μ *est un RSS en forme normale simple et si* $\mu \longrightarrow\!\!\!\!\!\rightarrow \mu'$ *alors* $h(\mu) > h(\mu')$*.*

DÉMONSTRATION: Comme μ est en forme normale simple seuls deux cas sont possibles.

Cas 1 : $\mu \xrightarrow{\text{comm}} \mu''$ et μ' est la forme normale simple de μ''. Alors μ' et μ'' sont des RSS et $|\mu'|_\partial = |\mu''|_\partial = |\mu|_\partial - 1$.

Cas 2 : $\mu \xrightarrow{\text{route}} \mu'' \neq 0$ et μ' est la forme normale simple de μ''. Si on note $\mu' = \overline{\sum}_{i=1}^{n} \mu'_i$, pour tout i on a $|\mu'_i|_{m,c} < |\mu|_{m,c}$ par le lemme 4.14. De plus, $|\mu'_i|_\partial = |\mu|_\partial$ par le lemme 4.13.

□

Théorème 4.17 *La réduction des RS normalise fortement.*

DÉMONSTRATION: Par le théorème 1.37, il suffit de montrer la normalisation faible de $—\bullet\rightarrow$. Par le lemme 4.11, celle-ci découle de la normalisation faible de $\longrightarrow\!\!\!\!\!\rightarrow$. On remarque que si $\mu \longrightarrow\!\!\!\!\!\rightarrow \mu'$ alors μ' est sans coupure simple : il suffit donc de montrer que tout RS sans coupure simple admet une forme normale pour $\longrightarrow\!\!\!\!\!\rightarrow$. Or par le lemme 4.16 et la définition de $\longrightarrow\!\!\!\!\!\rightarrow$ sur les RS, $\mu \longrightarrow\!\!\!\!\!\rightarrow \mu'$ implique $h(\mu) > h(\mu')$ dès que μ est sans coupure simple : on conclut par bonne fondation de l'ordre lexicographique sur $\mathbb{N}^2$. □

En particulier, si μ est un RS typé dans un système de types tel que la réduction soit complète (comme le typage polarisé ou le typage pur qui s'en déduit), alors μ se réduit en un unique RS sans coupures.

Chapitre 5

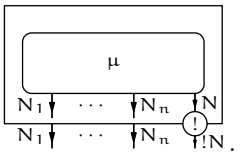
Réseaux différentiels polarisés

Dans ce chapitre, on introduit les boîtes de promotion dans les réseaux symétriques, ainsi que les réductions associées. Le système des réseaux différentiels avec boîtes fait partie du folklore, mais n'a jamais été publié à notre connaissance. Le système introduit ici en est une extension par polarisation : on appelle réseaux différentiels polarisés les réseaux correspondant.

5.1 Structures différentielles polarisées

Définition 5.1 *On appelle structure différentielle polarisée (SDP) toute SI sur la signature* $\Delta = \Sigma_{\text{Diff}}^{[!]}$. *On note* SDP *l'ensemble des SDP.*

On étend le système de types des SS aux SDP : si μ est une SDP telle que $\mu \vdash N_1, \ldots, N_n, N$ alors on type :



Contrairement aux cas des SIP (figure 2.7, page 40), on ne force plus μ à être simple.

On étend la réduction des SS aux SDP en ajoutant les règles de réduction avec boîtes données en figures 2.3 (réduction exponentielle des RME, page 34), 5.1 et 5.2. On vérifie que le système de réduction obtenu est complet pour le typage.

5.2 Correction

Définition 5.2 *Soit* μ *une SDP simple. On appelle interrupteur de* μ *tout graphe* G *dont les sommets sont les ports de* μ *et tel que :*

— *chaque fil de* μ *est une arête de* G *;*

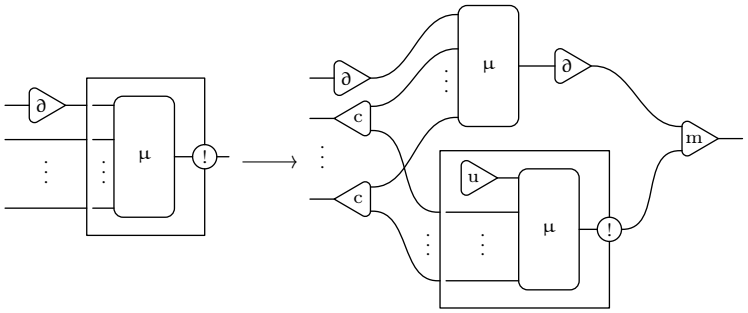


FIGURE 5.1 – Loi de la chaîne.

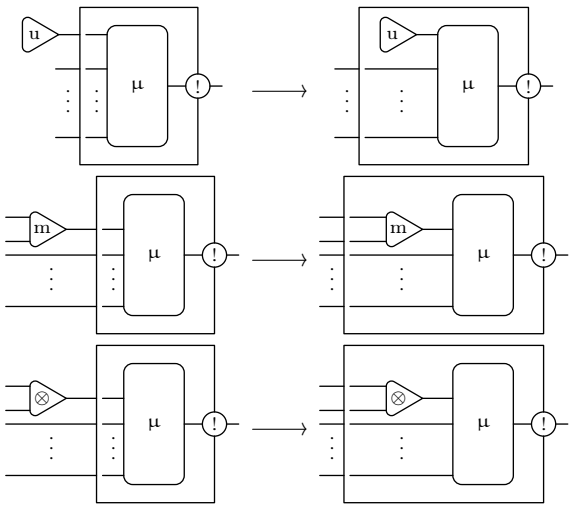


FIGURE 5.2 – Commutations.

— *pour chaque cellule* c *de* μ*, de symbole* $\otimes$, m, d *ou* ∂*, il y a une arête entre le port principal de* c *et chacun de ses ports passifs ;*
— *pour chaque boîte* c *de* μ*, il y a une arête entre le port principal de* c *et chacun de ses ports auxiliaires ;*
— *pour chaque cellule* c *de* μ*, de symbole* $\parr$ *ou* c*, il y a une arête entre le port actif de* c *et exactement un de ses ports passifs ;*
— *il n'y a pas d'autre arête hors des cas précédents.*

Définition 5.3 *On définit les réseaux différentiels polarisés (RDP) par induction sur la profondeur :*
— *un RDP de profondeur* 0 *est un RS ;*
— *un RDP de profondeur au plus* $n+1$ *est une SDP* $\mu = \overline{\sum}_{i=1}^{n} \mu_i$ *telle que, pour chaque SDP simple* μ_i*, tous les interrupteurs de* μ_i *sont acycliques et toutes les boîtes de* μ_i *ont pour symbole un RDP de profondeur au plus* n*.*

Lemme 5.4 *Les RDP sont stables par réduction.*

DÉMONSTRATION: Il suffit d'inspecter les cas de réduction. □

Définition 5.5 *On définit inductivement les SDP faiblement séquentielles :*
— *une SDP simple est faiblement séquentielle si elle peut s'obtenir inductivement par l'une des règles de construction de la figure 5.3, par la formation de la SDP vide* ε *ou par juxtaposition* $\mu_1 \parallel \mu_2$*, en supposant pour les cas non initiaux que* μ_1 *et* μ_2 *sont des SDP simples faiblement séquentielles et* μ_0 *est une SDP faiblement séquentielle ;*
— *une SDP* $\mu = \overline{\sum}_{i=1}^{n} \mu_i$ *est faiblement séquentielle si les* μ_i *sont faiblement séquentielles.*

Théorème 5.6 *Une SDP est faiblement séquentielle si et seulement si c'est un RDP.*

DÉMONSTRATION: C'est la même preuve que pour le théorème 2.5. □

5.3 Polarisation et dérivée

Le théorème 5.6 est un peu décevant pour qui a goûté à la magie de la polarisation en logique linéaire. En effet, on a vu que dans le cas de LLP (et donc de MELL restreinte aux formules polarisées), on pouvait donner un critère simple pour la séquentialité : les RIP sont exactement les SIP séquentielles (théorème 2.19, page 41). On va voir que la force simplificatrice de la polarisation est mise à mal par la présence de la dérivée ∂.

Définition 5.7 *On définit inductivement les SDP séquentielles :*
— *une SDP simple est séquentielle si elle peut s'obtenir inductivement par l'une des règles de construction de la figure 5.3, en supposant pour les cas non initiaux que* μ_1 *et* μ_2 *sont des SDP simples séquentielles et* μ_0 *est une SDP séquentielle ;*

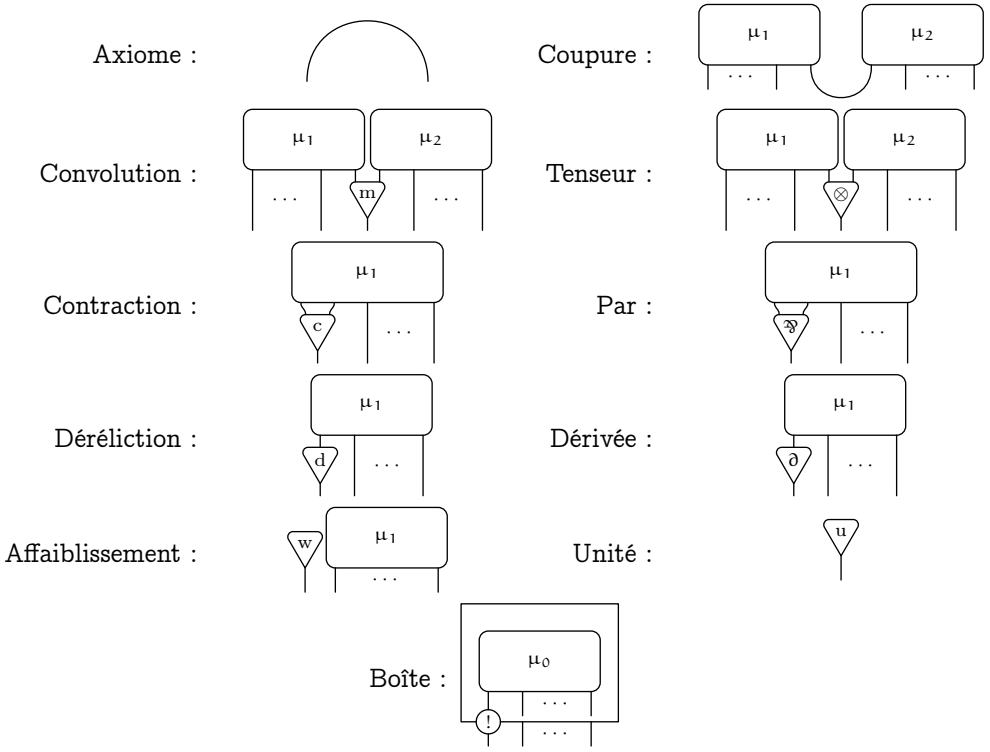


FIGURE 5.3 – Règles de construction des SDP séquentielles.

— *une SDP* $\mu = \overline{\sum}_{i=1}^{n} \mu_i$ *est séquentielle si les* μ_i *sont séquentielles.*

Cette définition suggère en particulier celle d'un calcul des séquents différentiel polarisé, avec une procédure d'élimination des coupures déduite des règles de réduction des SDP. Alors les SDP séquentielles typées sont exactement les traductions des preuves de ce calcul des séquents.

Notons déjà que la notion de prouvabilité a peu d'importance dans le contexte de réseaux différentiels : non seulement la SI nulle 0 est typable quel que soit la signature et le système de types, mais de manière plus frappante, la SDP séquentielle

a pour conclusion n'importe quelle formule positive P. Ce phénomène apparaît déjà dans les réseaux différentiels d'origine, puisque la SID séquentielle

a pour conclusion n'importe quelle formule exponentielle $!A$ de MELL. C'est-à-dire qu'on a quitté le domaine de la prouvabilité pour celui de l'analyse qualitative des preuves : u représente une preuve sans contenu, dont l'interrogation (par une déréliction) mène à l'échec (le réseau nul).

Reprenons les définitions 2.16 et 2.17.

Définition 5.8 *Soit* μ *une SDP simple typée. On appelle graphe de correction de* μ *le graphe orienté obtenu à partir de* μ *de la manière suivante :*

— *les sommets sont les cellules, axiomes (fils propres entre deux ports libres) et ports de boucles de* μ *;*

— *les arcs sont les fils entre ports de cellules (qu'on attache aux sommets associés aux cellules) et les fils de boucles ;*

— *les arcs associés aux fils propres sont orientés du port de type positif vers le port de type négatif.*

Définition 5.9 *On définit la correction des SDP typées par induction sur la profondeur :*

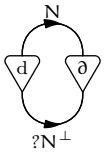
— *une SDP simple typée est correcte à profondeur* 0 *si son graphe de correction est acyclique et admet un unique sommet initial non* w *;*

— *une SDP simple typée est correcte à profondeur au plus* $n+1$ *si toutes ses boîtes ont pour symbole une SDP typée correcte à profondeur au plus* n*, et si son graphe de correction est acyclique et admet un unique sommet initial non* w*.*

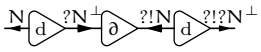
— *une SDP typée est correcte si toutes les SDP simples qui la constituent le sont.*

Remarquons que l'espoir de reproduire le théorème 2.19 dans le cadre des SDP est vain. En effet, on construit facilement des SDP correctes non séquentielles :

par exemple



n'est même pas un RDP simple. Dans le sens de la réciproque, il existe également des SDP séquentielles et incorrectes : par exemple, le graphe de correction de



possède deux sommets initiaux non w (les deux d). Le problème réside en ce que la dérivée, en autorisant des changements de polarité à volonté, brise une propriété fondamentale de LLP : la présence d'au plus une formule positive dans les conclusions d'une preuve.

Ceci nous amène à penser qu'il sera difficile, y compris dans le cas typé, de trouver un critère géométrique simple caractérisant les SDP séquentielles sans rajouter d'information à la structure. Dans de futures investigations sur le sujet, on pourra se tourner vers les artifices habituels :

(a) La solution la plus grossière consiste ajouter des boîtes pour la dérivée. C'est très insatisfaisant, car ces dernières ne jouent aucun rôle au cours de la réduction. Et rien ne garantit (sauf à imposer à ces boîtes le même genre de typage que pour celles de la promotion, ce qui est une restriction très forte) que ceci autorise un critère à la Laurent.

(b) On peut chercher à obtenir une caractérisation purement géométrique à la Danos–Regnier, valable dans le cas non typé, en ajoutant de la structure aux réseaux (boîtes, sauts, affaiblissements restreints aux axiomes, *etc.*), par exemple dans la veine de [JdNM07] ou[GM01]. Dans ce cas, la dérivée n'est plus un problème car, comme la déréliction, elle a la géométrie d'un fil. La difficulté sera plutôt de s'assurer que le critère se comporte bien vis-à-vis de la réduction.

On peut toutefois déjà proposer le résultat suivant :

Théorème 5.10 *Une SDP typée,* sans cellule ∂*, est séquentielle si et seulement si elle est correcte.*

DÉMONSTRATION: La preuve de Laurent s'adapte sans peine : le produit de convolution m est une cellule positive et se comporte comme un $\otimes$ à la fois par rapport au critère et pour la définition de la séquentialité ; de même pour u et l'unité du tenseur dans LLP. □

5.4 Des calculs de termes pour les réseaux différentiels polarisés

On peut hybrider le $\lambda\mu$-calcul de Parigot et le λ-calcul différentiel d'Ehrhard et Regnier, de manière purement syntaxique. Nous avons effectué ce travail dans

[Vau07b], où nous avons en particulier donné un sens calculatoire à la dérivée d'une μ-abstraction. On présente à nouveau le λμ-calcul différentiel au chapitre 6, et on montre que celui-ci correspond à des réseaux différentiels avec boîtes et règles structurelles sur les formules négatives, c'est-à-dire aux RDP où le produit et l'unité de convolution sont restreints aux types exponentiels !N.

Il est alors légitime de rechercher la contrepartie calculatoire de m et u sur les types positifs non exponentiels. L'article [Vau07a] présentait une variante du $\bar{\lambda}\mu$-calcul d'Herbelin, avec un produit de convolution sur les piles. Le lien avec le produit de convolution des réseaux y est établi via une sémantique dans $\mathcal{D}$ donnée dans l'article. Au chapitre 7, on rappelle la définition du calcul avec convolution et on en donne directement la traduction dans les réseaux différentiels polarisés : celle-ci ne fait pas usage de la dérivée ∂, ce qui témoigne de l'expressivité des règles costructurelles en elles-mêmes.

Enfin, il est possible de définir un calcul faisant intervenir de toute la richesse de la réduction des réseaux différentiels polarisés : le $\bar{\lambda}\mu$-calcul différentiel que nous présentons au chapitre 8 fait un usage intensif du produit de convolution sur les piles pour donner un statut similaire aux arguments intuitionnistes (les boîtes) et linéaires (introduits par les dérivées).

Afin d'établir formellement les résultats de simulation annoncés, il est nécessaire d'introduire une notion d'équivalence structurelle sur les SDP. Celle-ci généralise l'équivalence structurelle des SME (définition 2.8, page 36) et des SID (définition 2.33, page 50), en y ajoutant l'extensionnalité multiplicative, la commutation du $\parr$ avec la promotion, ainsi que les équations vues en section 3.4.2 (page 65).

Définition 5.11 *On définit l'équivalence structurelle $=_s$ sur les SDP comme la clôture contextuelle, réflexive, symétrique et transitive des transformations des figures 2.5 (page 36), 2.15 (page 51) et 5.4, toutes considérées à profondeur de boîte quelconque.*

L'introduction de l'extensionnalité multiplicative est rendue nécessaire par la traduction de la dérivée du λ-calcul différentiel (voir page 3) dans les réseaux : comme celle-ci ne fait pas décroître le type du terme dérivé, il est nécessaire de le déconstruire par un $\otimes$ pour accéder à la variable abstraite, puis de le reconstruire par un $\parr$ (voir la traduction du λμ-calcul différentiel en section 6.3.2, page 99).

La commutation du $\parr$ avec la boîte promotion est, elle, une commutation déjà parfaitement acceptable dans les réseaux polarisés de Laurent, au même titre que celle concernant la contraction (figure 2.5, page 36). Elle n'est cependant pas nécessaire pour assurer la correction de la traduction du λμ-calcul et des autres formalismes classiques envisagés dans [Lau02]. Notons toutefois que, de même que pour les quatre équations concernant la commutation des multiplicatifs avec les règles structurelles et costructurelles, une de ses orientations

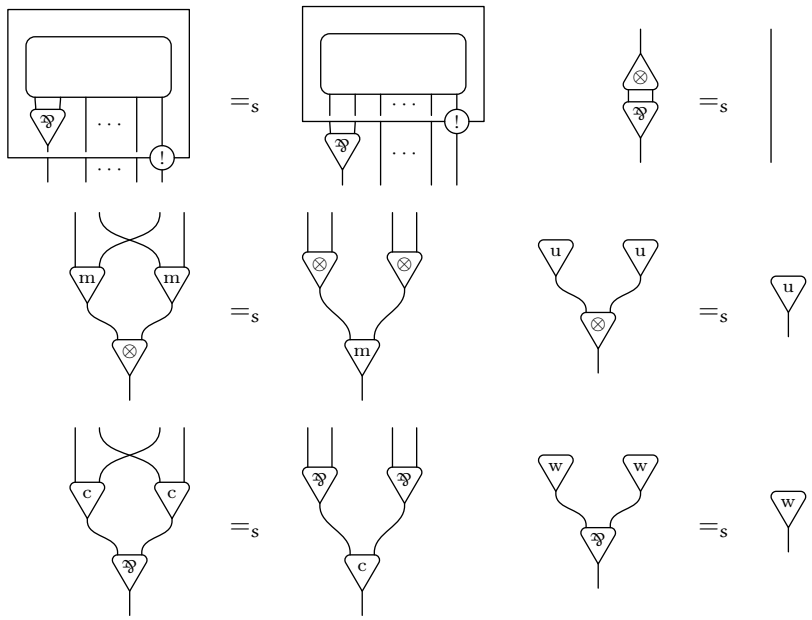
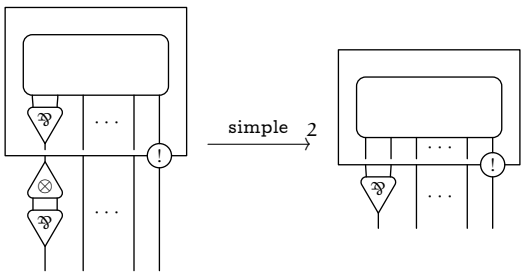
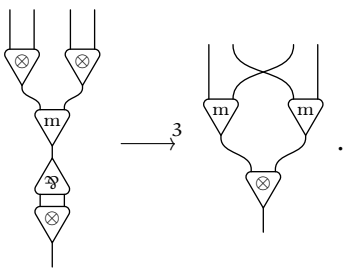


FIGURE 5.4 – Règles supplémentaires pour l'équivalence structurelle des SDP.

peut se déduire de l'extensionnalité multiplicative par la réduction : on a



et, par exemple,



.

Ce qui motive l'introduction de ces égalités en plus de l'extensionnalité multiplicative, c'est qu'elle permet d'éviter les réductions à rebours dans les résultats de simulation (par exemple le théorème 6.32, page 103).

Troisième partie

Interprétations λ-calculesques

Chapitre 6

λμ-calcul différentiel

Le λμ-calcul différentiel est une extension commune à la fois du λ-calcul différentiel d'Ehrhard et Regnier, et du λμ-calcul de Parigot. Ce chapitre révise l'article [Vau07b], en éliminant les combinaisons linéaires de termes qui venaient parasiter le discours.

6.1 Syntaxe

Le λμ-calcul différentiel est obtenu à partir du λμ-calcul, en introduisant une nouvelle construction syntaxique tirée du λ-calcul différentiel : si s et t sont des termes, on peut construire le nouveau terme $Ds \cdot t$ qui représente la dérivée de s selon t. L'ensemble des termes est par ailleurs muni d'une structure de monoïde commutatif, qui permet de sommer des termes : on note $+$ la loi correspondante et 0 le terme nul. Toutes les constructions sont linéaires (c'est-à-dire commutent avec la somme), sauf l'application qui est seulement linéaire en la fonction : on a par exemple $\lambda x\,(s+t) = \lambda x\,s + \lambda x\,t$ et $(s+t)\,u = (s)\,u + (t)\,u$ mais pas $(u)\,(s+t) = (u)\,s + (u)\,t$. On implémente ces propriétés en donnant d'abord une syntaxe de base, sur laquelle l'égalité entre termes se ramène à l'α-équivalence et la permutativité de la somme et de la dérivée, puis en introduisant des notations plus générales.

Comme on ne manipule que des sommes de termes, sans coefficients, il est clair que les constructions syntaxiques que nous introduisons sont bien définies, et que la notion d'égalité entre termes ainsi obtenue ne recèle pas plus de pièges que la simple α-équivalence. Ceci permet de se concentrer sur les nouveautés syntaxiques introduites par la différentiation, en contraste avec les difficultés soulignées dans [Vau07c].

6.1.1 Morphologie

Définition 6.1 *On définit les ensembles Λ_0 (pré-termes ρ), Λ (termes simples $s, t, u, \ldots$), Λ^+ (termes $S, T, U, \ldots$), $\mathcal{C}$ (commandes simples c, d) et $\mathcal{C}^+$*

(commandes C, D*) par les grammaires mutuellement inductives suivantes :*

$$\begin{aligned} \rho &::= x \mid \lambda x\, s \mid (s)\, T \mid \mu\alpha\, c \\ s &::= \rho \mid Ds \cdot t \\ c &::= [\alpha]\, s \\ S &::= 0 \mid s + S \\ C &::= 0 \mid c + C \end{aligned}$$

Comme pour le λ-calcul avec ressources de la section 2.5.4, on considère les termes et les commandes modulo permutativité de la somme et des dérivées successives : on identifie par exemple $s + (s' + S)$ et $s' + (s + S)$, de même que $D\,(Ds \cdot t) \cdot u$ et $D\,(Ds \cdot u) \cdot t$. On appelle égalité permutative la relation d'équivalence ainsi définie sur chacun des ensembles précédents. L'égalité permutative préserve les variables libres et liées, elle est donc compatible avec l'α-conversion. La clôture transitive commune de l'α-équivalence et de l'égalité permutative est trivialement une relation d'équivalence : dans toute la suite, on considérera comme égaux deux objets identifiés par cette relation.

6.1.2 Notations

On note $D^n s \cdot (u_1, \ldots, u_n)$ le terme simple $D(\cdots(Ds \cdot u_1)\cdots) \cdot u_n$. En particulier, $D^1 s \cdot u = Ds \cdot u$. On a directement

$$\begin{aligned} & D^n D^p s \cdot (u_1, \ldots, u_p) \cdot (u_{p+1}, \ldots, u_{p+n}) \\ &= D^{n+p} s \cdot (u_1, \ldots, u_{n+p}) \\ &= D^p D^n s \cdot (u_{p+1}, \ldots, u_{p+n}) \cdot (u_1, \ldots, u_p) \end{aligned}$$

par l'égalité permutative. De plus, tout terme simple s'écrit de manière unique $D^n \rho \cdot (u_1, \ldots, u_n)$ à commutation des u_i près.

On appelle objet simple tout terme simple et toute commande simple, et objet tout terme et toute commande. On utilisera la minuscule grecque θ pour dénoter un objet simple et la majuscule Θ pour un objet. En général, si un objet simple θ et un objet Θ apparaissent dans le même contexte, il est sous-entendu qu'ils appartiennent au même genre syntaxique (terme ou commande).

Soient $\theta_1, \ldots, \theta_n$ des objets simples et Θ un objet, on abrège $\theta_1 + (\cdots + (\theta_n + \Theta)\cdots)$ en $\theta_1 + \cdots + \theta_n + \Theta$. Si θ est un objet simple, on pourra aussi écrire θ pour l'objet correspondant $\theta + 0$. Ainsi, tout objet Θ peut s'écrire sans ambiguïté $\Theta = \theta_1 + \cdots + \theta_n$ ou même $\Theta = \sum_{i=1}^{n} \theta_i$. Supposons que $\Theta = \theta_1 + \cdots + \theta_n$ et $\Theta' = \theta'_1 + \cdots + \theta'_p$: on note $\Theta + \Theta'$ l'objet $\theta_1 + \cdots + \theta_n + \theta'_1 + \cdots + \theta'_p$.

À ces conventions d'écriture près, $+$ devient une opération associative et commutative sur les termes (resp. sur les commandes) et l'objet 0 est neutre. En particulier, on écrira $n\Theta$ où $n \in \mathbb{N}$, pour le terme obtenu en ajoutant n fois Θ à 0. Si on note $\delta_{x,y}$ le symbole de Kronecker, on a $\delta_{x,y}\Theta = \Theta$ si $x = y$ et 0 sinon.

On étend maintenant les constructions de la syntaxe de base par linéarité (sauf pour la position d'argument d'une application).

Définition 6.2 *Supposons* $s_1, \ldots, s_n \in \Lambda$, $t_1, \ldots, t_p \in \Lambda$, $c_1, \ldots, c_q \in \mathcal{C}$ *et* $S \in \Lambda^+$. *Alors on pose :*

$$\begin{aligned}
\lambda x \left(\sum_{i=1}^{n} s_i \right) &= \sum_{i=1}^{n} \lambda x\, s_i \\
\left(\sum_{i=1}^{n} s_i \right) S &= \sum_{i=1}^{n} (s_i)\, S \\
\mu\alpha \left(\sum_{k=1}^{q} c_k \right) &= \sum_{k=1}^{q} \mu\alpha\, c_k \\
[\alpha] \left(\sum_{i=1}^{n} s_i \right) &= \sum_{i=1}^{n} [\alpha]\, s_i \\
D \left(\sum_{i=1}^{n} s_i \right) \cdot \left(\sum_{j=1}^{p} t_j \right) &= \sum_{i=1}^{n} \sum_{j=1}^{p} Ds_i \cdot t_j .
\end{aligned}$$

Cette définition introduit un conflit de notations : par exemple, $\lambda x\, s$ peut se lire à la fois comme un pré-terme de la syntaxe de base et, modulo l'identification $s = s + 0$, comme le terme dénoté par $\lambda x\, (s + 0)$ dans la définition précédente. Cette ambiguïté est cependant inoffensive, car le terme correspondant est toujours le même.

6.1.3 Substitutions

On note $\mathrm{VL}(\Theta)$ l'ensemble des variables libres de Θ et $\mathrm{NL}(\Theta)$ l'ensemble de ses noms libres.

Il y a quatre opérations de base sur les objets : la *substitution* $\Theta[x := T]$, la *dérivée partielle* $\frac{\partial \Theta}{\partial x} \cdot T$, l'*application nommée* $(\Theta)_\alpha T$ et la *dérivée nommée* $D_\alpha \Theta \cdot T$. Chacune est associée à une manière d'utiliser un argument, et correspond typiquement à la forme réduite d'un redex (voir la définition de la réduction en section 6.2).

La substitution $\Theta[x := T]$ est la substitution usuelle de λ-calcul étendue à tous les objets :

Définition 6.3 *La substitution* $\Theta[x := T]$ *est définie inductivement par :*

$$\begin{aligned}
y[x := T] &= \begin{cases} T \ \textit{si}\ x = y \\ y \ \textit{sinon} \end{cases} \\
(\lambda y\, s)[x := T] &= \lambda y\, (s[x := T]) \\
((s)\, u)[x := T] &= (s[x := T])\, (u[x := T]) \\
(\mu\alpha\, c)[x := T] &= \mu\alpha\, (c[x := T]) \\
([\alpha]\, s)[x := T] &= [\alpha]\, (s[x := T]) \\
(Ds \cdot u)[x := T] &= D\, (s[x := T]) \cdot (u[x := T]) \\
0[x := T] &= 0 \\
(\theta + \Theta)[x := T] &= \theta[x := T] + \Theta[x := T]
\end{aligned}$$

où $x \neq y$ *et* $y \notin \mathrm{VL}(T)$ *dans le cas de la* λ*-abstraction, et* $\alpha \notin \mathrm{NL}(T)$ *dans le cas de la* μ*-abstraction.*

La dérivée partielle $\frac{\partial \Theta}{\partial x} \cdot T$ est celle de [ER03], qu'on étend sans problème en présence de la μ-abstraction et des commandes. C'est la somme de tous les objets obtenus en remplaçant exactement une occurrence linéaire de x dans Θ par T. Dans la mesure où toutes les occurrences de x ne sont pas nécessairement en position linéaire, on linéarise à la volée : dans le cas de l'application, on extrait une copie linéaire de l'argument avant d'y effectuer la substitution. Formellement :

Définition 6.4 *La dérivée partielle* $\frac{\partial \Theta}{\partial x} \cdot T$ *est définie inductivement par :*

$$\begin{aligned}
\frac{\partial y}{\partial x} \cdot T &= \delta_{x,y} T \\
\frac{\partial \lambda y\, s}{\partial x} \cdot T &= \lambda y \left(\frac{\partial s}{\partial x} \cdot T \right) \\
\frac{\partial (s)\, U}{\partial x} \cdot T &= \left(\frac{\partial s}{\partial x} \cdot T \right) U + \left(Ds \cdot \left(\frac{\partial U}{\partial x} \cdot T \right) \right) u \\
\frac{\partial \mu\alpha\, c}{\partial x} \cdot T &= \mu\alpha \left(\frac{\partial c}{\partial x} \cdot T \right) \\
\frac{\partial [\alpha]\, s}{\partial x} \cdot T &= [\alpha] \left(\frac{\partial s}{\partial x} \cdot T \right) \\
\frac{\partial\, Ds \cdot u}{\partial x} \cdot T &= D \left(\frac{\partial s}{\partial x} \cdot T \right) \cdot u + Ds \cdot \left(\frac{\partial u}{\partial x} \cdot T \right) \\
\frac{\partial\, 0}{\partial x} \cdot T &= 0 \\
\frac{\partial\, \theta + \Theta}{\partial x} \cdot T &= \frac{\partial \theta}{\partial x} \cdot T + \frac{\partial \Theta}{\partial x} \cdot T
\end{aligned}$$

où $x \neq y$ *et* $y \notin \mathrm{VL}(T)$ *dans le cas de la* λ*-abstraction, et* $\alpha \notin \mathrm{NL}(T)$ *dans le cas de la* μ*-abstraction.*

L'application nommée $(\Theta)_\alpha T$ est celle du λμ-calcul, étendue à tous les objets. Rappelons que dans la terminologie originale de Parigot [Par92], elle est qualifiée de *substitution structurelle* : dans la littérature, on trouve le plus souvent la notation $S\,[[\alpha]\,U := [\alpha]\,(U)\,T]$ ou $S\,[[\alpha]\,(U)\,T/[\alpha]\,U]$. En effet, $(\Theta)_\alpha T$ est obtenu en remplaçant inductivement toute sous-commande $[\alpha]\,U$ dans Θ par $[\alpha]\,(U)\,T$.

Notre notation $(\Theta)_\alpha T$ a l'avantage de s'accorder avec l'intuition : $(\Theta)_\alpha T$ est Θ appliqué à T à travers le nom α. La traduction du λμ-calcul dans les réseaux polarisés renforce ce point de vue : comparer le lemme 2.23 (page 44) à la traduction de l'application dans la figure 2.9 (page 43). On verra d'ailleurs que certaines propriétés établies plus loin concernant l'application nommée $(\Theta)_\alpha T$ rappellent fortement des propriétés similaires de l'application : voir par exemple les lemmes de commutation 6.10 à 6.13. Cette notation est par ailleurs cohérente avec celle de la seule nouvelle opération du λμ-calcul différentiel : la dérivée nommée, qu'on introduira ensuite.

Définition 6.5 *L'application nommée* $(\Theta)_\alpha T$ *est définie inductivement par :*

$$\begin{aligned}
(x)_\alpha T &= x \\
(\lambda x\, s)_\alpha T &= \lambda x\, ((s)_\alpha T) \\
((s)\, u)_\alpha T &= ((s)_\alpha T)\, ((u)_\alpha T) \\
(\mu\beta\, c)_\alpha T &= \mu\beta\, ((c)_\alpha T) \\
([\beta]\, s)_\alpha T &= \begin{cases} [\alpha]\, ((s)_\alpha T)\, T & \textit{si } \alpha = \beta \\ [\beta]\, (s)_\alpha T & \textit{sinon} \end{cases} \\
(Ds \cdot u)_\alpha T &= D\, ((s)_\alpha T) \cdot ((u)_\alpha T) \\
(0)_\alpha T &= 0 \\
(\theta + \Theta)_\alpha T &= (\theta)_\alpha T + (\Theta)_\alpha T
\end{aligned}$$

où $\alpha \neq \beta$ *et* $\beta \notin \mathrm{NL}(T)$ *dans le cas de la* μ*-abstraction, et* $x \notin \mathrm{VL}(T)$ *dans le cas de la* λ*-abstraction.*

La dérivée nommée est la seule opération nouvelle. On peut tenter de l'introduire par analogie :

— la dérivée est une version linéaire de l'application, qui ne fournit qu'une copie linéaire de son argument ;
— la dérivée partielle, version linéaire de la substitution, remplace exactement une occurrence linéaire d'une variable par son argument ;
— la dérivée nommée, version linéaire de l'application nommée, dérive exactement un sous-terme en position linéaire, sous un nom donné, suivant son argument.

Formellement :

Définition 6.6 *La dérivée nommée* $D_\alpha \Theta \cdot T$ *est définie inductivement par :*

$$\begin{aligned}
D_\alpha x \cdot T &= 0 \\
D_\alpha \lambda x\, s \cdot T &= \lambda x\, (D_\alpha s \cdot T) \\
D_\alpha (s)\, u \cdot T &= (D_\alpha s \cdot T)\, u + (Ds \cdot (D_\alpha u \cdot T))\, u \\
D_\alpha \mu\beta\, c \cdot T &= \mu\beta\, (D_\alpha c \cdot T) \\
D_\alpha\, [\beta]\, s \cdot T &= \delta_{\alpha,\beta}\, [\alpha]\, (Ds \cdot T) + [\beta]\, (D_\alpha s \cdot T) \\
D_\alpha\, (Ds \cdot u) \cdot T &= D\, (D_\alpha s \cdot T) \cdot u + Ds \cdot (D_\alpha u \cdot T) \\
D_\alpha 0 \cdot T &= 0 \\
D_\alpha (\theta + \Theta) \cdot T &= D_\alpha \theta \cdot T + D_\alpha \Theta \cdot T
\end{aligned}$$

où $\alpha \neq \beta$ *et* $\beta \notin \mathrm{NL}(T)$ *dans le cas de la* μ*-abstraction, et* $x \notin \mathrm{VL}(T)$ *dans le cas de la* λ*-abstraction.*

Comme pour l'application nommée avec l'application, la plupart des propriétés de la dérivée nommée rappellent des propriétés similaires de la dérivée : voir en particulier les lemmes 6.10 à 6.13.

Lemme 6.7 *Si* $x \in \mathcal{V}$ *et* Θ *est un objet tel que* $x \notin \mathrm{VL}(\Theta)$*, alors pour tout terme* S*,* $\Theta[x := S] = \Theta$*. Si* α *est un nom et* Θ *est un objet tel que* $\alpha \notin \mathrm{NL}(\Theta)$*, alors pour tout terme* S*,* $(\Theta)_\alpha S = \Theta$*.*

DÉMONSTRATION: Facile par induction sur Θ. □

Lemme 6.8 *Si* $x \in \mathcal{V}$ *et* Θ *est un objet tel que* $x \notin \mathrm{VL}(\Theta)$*, alors pour tout terme* S*,* $\frac{\partial \Theta}{\partial x} \cdot S = 0$*. Si* $\alpha \in \mathcal{N}$ *et* Θ *est un objet tel que* $\alpha \notin \mathrm{NL}(\Theta)$*, alors pour tout terme* S*,* $D_\alpha \Theta \cdot S = 0$*.*

Lemme 6.9 *Si* $x \in \mathcal{V}$*,* $\alpha \in \mathcal{N}$*,* Θ *est un objet et* $S, T \in \Lambda^+$*, alors*

$$\begin{aligned}
\frac{\partial \Theta}{\partial x} \cdot 0 &= 0 \\
\frac{\partial \Theta}{\partial x} \cdot (S + T) &= \frac{\partial \Theta}{\partial x} \cdot S + \frac{\partial \Theta}{\partial x} \cdot T \\
D_\alpha \Theta \cdot 0 &= 0 \\
D_\alpha \Theta \cdot (S + T) &= D_\alpha \Theta \cdot S + D_\alpha \Theta \cdot T.
\end{aligned}$$

DÉMONSTRATION: Les deux lemmes précédents se démontrent facilement par induction sur Θ. Le cas de l'application est intéressant dans chacune des preuves, en ce qu'il fait apparaître l'intérêt de la linéarisation introduite dans les définitions 6.4 et 6.6. □

6.1.4 Commutations

Cette section est consacrée aux commutations entre opérations : on donne des conditions suffisantes pour pouvoir échanger deux opérations successives sur un objet, sans changer l'effet produit. Chaque item de chaque lemme de commutation est prouvé séparément par induction sur les objets. Les preuves sont typiquement longues, ennuyeuses et faciles. Le lecteur consciencieux est invité à s'en convaincre en prouvant l'item de son choix.

Pour toutes variables x, y, tous noms α, β, tout objet Θ et tous termes U et V, les lemmes suivants sont vérifiés.

Lemme 6.10 *Si* $x \neq y$ *et* $x \notin \mathrm{VL}(V)$ *alors*

$$\Theta[x := U][y := V] = \Theta[y := V][x := U[y := V]]$$

et

$$\left(\frac{\partial \Theta}{\partial x} \cdot U\right)[y := V] = \frac{\partial \Theta[y := V]}{\partial x} \cdot (U[y := V]).$$

Si $\alpha \notin \mathrm{NL}(V)$ *alors*

$$((\Theta)_\alpha U)[x := V] = (\Theta[x := V])_\alpha (U[x := V])$$

et

$$(D_\alpha \Theta \cdot U)[x := V] = D_\alpha (\Theta[x := V]) \cdot (U[x := V]).$$

Lemme 6.11 *Si* $x \neq y$ *et* $x \notin \mathrm{VL}(U) \cup \mathrm{VL}(V)$ *alors*

$$\frac{\partial \Theta[x := U]}{\partial y} \cdot V = \left(\frac{\partial \Theta}{\partial y} \cdot V\right)[x := U] + \left(\frac{\partial \Theta}{\partial x} \cdot \left(\frac{\partial U}{\partial y} \cdot V\right)\right)[x := U].$$

Si $x \notin \mathrm{VL}(V)$ *alors*

$$\frac{\partial}{\partial y}\left(\frac{\partial \Theta}{\partial x} \cdot U\right) \cdot V = \frac{\partial}{\partial x}\left(\frac{\partial \Theta}{\partial y} \cdot V\right) \cdot U + \frac{\partial \Theta}{\partial x} \cdot \left(\frac{\partial U}{\partial y} \cdot V\right).$$

Si $\alpha \notin \mathrm{NL}(U) \cup \mathrm{NL}(V)$ *alors*

$$\frac{\partial (\Theta)_\alpha U}{\partial x} \cdot V = \left(\frac{\partial \Theta}{\partial x} \cdot V\right)_\alpha U + \left(\mathrm{D}_\alpha \Theta \cdot \left(\frac{\partial U}{\partial x} \cdot V\right)\right)_\alpha U.$$

Si $\alpha \notin \mathrm{NL}(V)$ *alors*

$$\frac{\partial \mathrm{D}_\alpha \Theta \cdot U}{\partial x} \cdot V = \mathrm{D}_\alpha\left(\frac{\partial \Theta}{\partial x} \cdot V\right) \cdot U + \mathrm{D}_\alpha \Theta \cdot \left(\frac{\partial U}{\partial x} \cdot V\right).$$

Lemme 6.12 *Si* $x \notin \mathrm{VL}(V)$ *alors*

$$(\Theta[x := U])_\alpha V = ((\Theta)_\alpha V)[x := (U)_\alpha V]$$

et

$$\left(\frac{\partial \Theta}{\partial x} \cdot U\right)_\alpha V = \frac{\partial (\Theta)_\alpha V}{\partial x} \cdot ((U)_\alpha V).$$

Si $\alpha \neq \beta$ *et* $\alpha \notin \mathrm{NL}(V)$ *alors*

$$((\Theta)_\alpha U)_\beta V = \left((\Theta)_\beta V\right)_\alpha \left((U)_\beta V\right)$$

et

$$(\mathrm{D}_\alpha \Theta \cdot U)_\beta V = \mathrm{D}_\alpha (\Theta)_\beta V \cdot \left((U)_\beta V\right).$$

Lemme 6.13 *Si* $x \notin \mathrm{VL}(U) \cup \mathrm{VL}(V)$ *alors*

$$\mathrm{D}_\alpha \Theta[x := U] \cdot V = (\mathrm{D}_\alpha \Theta \cdot V)[x := U] + \left(\frac{\partial \Theta}{\partial x} \cdot (\mathrm{D}_\alpha U \cdot V)\right)[x := U].$$

Si $x \notin \mathrm{VL}(V)$ *alors*

$$\mathrm{D}_\alpha\left(\frac{\partial \Theta}{\partial x} \cdot U\right) \cdot V = \frac{\partial \mathrm{D}_\alpha \Theta \cdot V}{\partial x} \cdot U + \frac{\partial \Theta}{\partial x} \cdot (\mathrm{D}_\alpha U \cdot V).$$

Si $\alpha \neq \beta$ *et* $\alpha \notin \mathrm{NL}(U) \cup \mathrm{NL}(V)$ *alors*

$$\mathrm{D}_\beta (\Theta)_\alpha U \cdot V = (\mathrm{D}_\beta \Theta \cdot V)_\alpha U + (\mathrm{D}_\alpha \Theta \cdot (\mathrm{D}_\beta U \cdot V))_\alpha U.$$

Si $\alpha \notin \mathrm{NL}(V)$ *alors*

$$\mathrm{D}_\beta\left(\mathrm{D}_\alpha \Theta \cdot U\right) \cdot V = \mathrm{D}_\alpha\left(\mathrm{D}_\beta \Theta \cdot V\right) \cdot U + \mathrm{D}_\alpha \Theta \cdot (\mathrm{D}_\beta U \cdot V).$$

Le cas de l'application nommée dans les lemmes 6.10 à 6.13 ressemble fortement au cas de l'application dans les définitions 6.3 à 6.6. En particulier, le lemme 6.11 (resp. 6.13) fait apparaître la linéarisation présente dans la définition de la dérivée partielle (resp nommée). Ceci renforce la pertinence des notations choisies. De la même manière, le cas de la dérivée nommée dans les lemmes 6.10 à 6.13 correspond aux items impliquant la dérivée dans les définitions 6.3 à 6.6.

6.1.5 Opérations itérées

Les lemmes 6.10 à 6.13 nous permettent de définir des versions itérées des opérations.

Corollaire 6.14 *Si* $x_1, \dots, x_n$ *sont des variables distinctes deux à deux, libres dans aucun des termes* $u_1, \dots, u_n$*, alors pour tout objet* Θ*,*

$$\Theta\left[x_{\sigma(1)} := u_{\sigma(1)}\right] \dots \left[x_{\sigma(n)} := u_{\sigma(n)}\right]$$

ne dépend pas de la permutation σ *de* $\{1, \dots, n\}$*. On l'écrit*

$$\Theta\left[x_1, \dots, x_n := u_1, \dots, u_n\right].$$

Corollaire 6.15 *Si les variables* $x_1, \dots, x_n$ *ne sont libres dans aucun des termes* $u_1, \dots, u_n$*, alors pour tout objet* Θ*,*

$$\frac{\partial}{\partial x_{\sigma(n)}}\left(\dots \frac{\partial \Theta}{\partial x_{\sigma(1)}} \cdot u_{\sigma(1)} \dots\right) \cdot u_{\sigma(n)}$$

ne dépend pas de la permutation σ *de* $\{1, \dots, n\}$*. On l'écrit*

$$\frac{\partial^n \Theta}{\partial x_1 \dots \partial x_n} \cdot (u_1, \dots, u_n).$$

Si $x_1 = \dots = x_n = x$*, on abrège en* $\frac{\partial^n \Theta}{\partial x^n} \cdot (u_1, \dots, u_n)$*.*

Corollaire 6.16 *Si* $\alpha_1, \dots, \alpha_n$ *sont des noms distincts deux à deux, libres dans aucun des termes* $u_1, \dots, u_n$*, alors pour tout objet* Θ*,*

$$\left(\dots (\Theta)_{\alpha_{\sigma(1)}} u_{\sigma(1)} \dots\right)_{\alpha_{\sigma(n)}} u_{\sigma(n)}$$

ne dépend pas de la permutation σ *de* $\{1, \dots, n\}$*. On l'écrit*

$$(\Theta)_{\alpha_1, \dots, \alpha_n} (u_1, \dots, u_n).$$

Corollaire 6.17 *Si les noms* $\alpha_1, \dots, \alpha_n$ *ne sont libres dans aucun des termes* $u_1, \dots, u_n$*, alors pour tout objet* Θ*,*

$$D_{\alpha_{\sigma(n)}}\left(\dots D_{\alpha_{\sigma(1)}} \Theta \cdot u_{\sigma(1)} \dots\right) \cdot u_{\sigma(n)}$$

ne dépend pas de la permutation σ *de* $\{1, \dots, n\}$*. On l'écrit*

$$D_{\alpha_1, \dots, \alpha_n} \Theta \cdot (u_1, \dots, u_n).$$

Si $\alpha_1 = \dots = \alpha_n = \alpha$*, on abrège en* $D^n_\alpha \Theta \cdot (u_1, \dots, u_n)$*.*

6.2 Réduction

La réduction du λμ-calcul différentiel se déduit de celles du λμ-calcul [Par92] et du λ-calcul différentiel [ER03], avec la règle supplémentaire

$$D\mu\alpha\, c \cdot T \to \mu\alpha\, D_\alpha c \cdot T$$

qui correspond au seul redex nouveau. Remarquons le parallèle avec la règle du λμ-calcul

$$(\mu\alpha\, c)\, T \to \mu\alpha\, (c)_\alpha\, T$$

qui renforce la pertinence notre choix de notations.

6.2.1 β-réduction

Définition 6.18 *On appelle relation contextuelle tout couple τ de relations binaires, respectivement sur les termes et les commandes, également notées τ, et telles que :*

- *si $s\ \tau\ S'$ alors $\lambda x\, s\ \tau\ \lambda x\, S'$, $Ds \cdot t\ \tau\ DS' \cdot t$, $Dt \cdot s\ \tau\ Dt \cdot S'$, $(s)\, T\ \tau\ (S')\, T$ et $[\alpha]\, s\ \tau\ [\alpha]\, S'$;*
- *si $c\ \tau\ C'$ alors $\mu\alpha\, c\ \tau\ \mu\alpha\, C'$;*
- *si $T\ \tau\ T'$ alors $(s)\, T\ \tau\ (s)\, T'$;*
- *et si $\theta_0\ \tau\ \Theta_0'$ alors $\theta_0 + \Theta_1\ \tau\ \Theta_0' + \Theta_1$.*

On peut toujours considérer la clôture contextuelle d'une relation sur les termes ou les commandes : celle-ci est définie par induction sur les termes et commandes, par les clauses de la définition précédente. Il s'agit bien d'une induction car chacune des clauses se réfère uniquement à des sous-objets stricts. En particulier, la clôture contextuelle est une opération croissante au sens de l'inclusion des relations.

Ceci contraste avec la définition de la réduction dans [ER03, Section 2] ou [Vau07b, Section 3.2], où l'on procède par induction sur la profondeur du redex concerné éliminé. Ceci est rendu nécessaire par la présence de combinaisons linéaires plutôt que des seules sommes : nous avons étudié ce phénomène plus en détail dans [Vau07c]. Rappelons que nous avons choisi ici de ne considérer que des sommes : l'égalité des termes est donnée par les seules α-équivalence et permutativité, de sorte que θ et Θ sont toujours des sous-objets stricts de $\theta + \Theta$.

Définition 6.19 *La réduction $\to$ du λμ-calcul différentiel est la plus petite relation contextuelle telle que :*

$$(\lambda x\, s)\, T \to s\,[x := T] \tag{6.1}$$

$$D\lambda x\, s \cdot t \to \lambda x \left(\frac{\partial\, s}{\partial\, x} \cdot t \right) \tag{6.2}$$

$$(\mu\alpha\, c)\, T \to \mu\alpha\, ((c)_\alpha\, T) \tag{6.3}$$

$$D\mu\alpha\, c \cdot t \to \mu\alpha\, (D_\alpha c \cdot t) \tag{6.4}$$

avec $x \notin VL(t)$ et $\alpha \notin NL(T) \cup NL(t)$.

Définition 6.20 *Une relation binaire τ sur le monoïde $(M, +, 0)$ est dite linéaire si $0 \,\tau\, 0$ et si $m + n \,\tau\, m' + n'$ dès que $m \,\tau\, m'$ et $n \,\tau\, n'$.*

Notons que $\to$ n'est pas linéaire : $0 \not\to 0$.

Définition 6.21 *On appelle congruence tout couple τ de relations, respectivement sur les termes et les commandes, chacune étant réflexive et linéaire et également notée τ, telles que si $S \,\tau\, S'$, $T \,\tau\, T'$ et $C \,\tau\, C'$, alors*

$$\begin{array}{rcl} \lambda x\, S & \tau & \lambda x\, S' \\ D S \cdot T & \tau & D S' \cdot T' \\ (S)\, T & \tau & (S')\, T' \\ [\alpha]\, S & \tau & [\alpha]\, S' \\ \mu\alpha\, C & \tau & \mu\alpha\, C' \ . \end{array}$$

En particulier, une congruence est contextuelle, et la plus petite congruence contenant une relation contient aussi sa clôture contextuelle. Mieux :

Lemme 6.22 *La fermeture réflexive et transitive d'une relation contextuelle est une congruence.*

DÉMONSTRATION: La réflexivité est donnée. La linéarité provient de la réflexivité (pour 0) et de l'itération par transitivité de la dernière clause de contextualité. Les autres conditions s'obtiennent des clauses correspondantes de la contextualité, par transitivité pour l'application et la dérivée. □

En particulier, $\to^*$ est une congruence.

6.2.2 Réduction parallèle

On prouve la confluence par la méthode habituelle de Tait–Martin-Löf : on exhibe une extension parallèle de la réduction, dans laquelle tous les redex d'un terme peuvent être réduits simultanément. On montrera ensuite qu'elle vérifie la propriété du diamant.

Définition 6.23 *La réduction parallèle $\rightrightarrows$ est la plus petite congruence telle que :*

$$(D^n \lambda x\, s \cdot (u_1, \ldots, u_n))\, T \rightrightarrows \left(\frac{\partial^n s}{\partial x^n} \cdot (u_1, \ldots, u_n)\right)[x := T] \tag{6.5}$$

$$D^n \lambda x\, s \cdot (u_1, \ldots, u_n) \rightrightarrows \lambda x \left(\frac{\partial^n s}{\partial x^n} \cdot (u_1, \ldots, u_n)\right) \tag{6.6}$$

$$(D^n \mu\alpha\, c \cdot (u_1, \ldots, u_n))\, T \rightrightarrows \mu\alpha\, ((D^n_\alpha c \cdot (u_1, \ldots, u_n))_\alpha\, T) \tag{6.7}$$

$$D^n \mu\alpha\, c \cdot (u_1, \ldots, u_n) \rightrightarrows \mu\alpha\, (D^n_\alpha c \cdot (u_1, \ldots, u_n)) \tag{6.8}$$

avec les précautions d'usage permettant d'éviter la capture de variables ou de noms.

Il faut bien remarquer que la forme générale d'un redex pour l'application dans la réduction parallèle est

$$(D^n \lambda x\, v \cdot (w_1, \ldots, w_n))\, T \text{ ou } (D^n \mu \alpha\, c \cdot (w_1, \ldots, w_n))\, T$$

et n'est donc pas limitée aux seuls redex de $\to$, c'est-à-dire

$$(\lambda x\, v)\, T \text{ ou } (\mu \alpha\, c)\, T.$$

Cette généralisation est nécessaire pour que le lemme 6.25 soit valide : elle est donc importante pour que $\rightrightarrows$ vérifie la propriété du diamant (lemme 6.28). Ceci s'accorde avec le fait que la λ-abstraction, consommée par l'application, ne l'est pas par la dérivée : voir aussi la définition 6.26 du réduit plein.

Lemme 6.24 *On a* $\to \subset \rightrightarrows \subset \to^*$.

DÉMONSTRATION: Le fait que $S \to T$ implique $S \rightrightarrows T$ est direct d'après les définitions. De plus $S \rightrightarrows T$ implique $S \to^* T$: c'est vrai dans le cas des redex de $\rightrightarrows$ et la clôture par congruence est croissante. □

Le lemme suivant établit la propriété essentielle de la réduction parallèle.

Lemme 6.25 *Soient* Θ *et* Θ' *des objets et* S *et* $S' \in \Lambda^+$. *Si* $\Theta \rightrightarrows \Theta'$ *et* $S \rightrightarrows S'$, *alors pour tous* $x \in \mathcal{V}$ *et* $\alpha \in \mathcal{N}$ *on a :*

$$\begin{aligned}
\Theta[x := S] &\rightrightarrows \Theta'[x := S'] \ , \\
\frac{\partial \Theta}{\partial x} \cdot S &\rightrightarrows \frac{\partial \Theta'}{\partial x} \cdot S' \ , \\
(\Theta)_\alpha\, S &\rightrightarrows (\Theta')_\alpha\, S' \quad \textit{et} \\
D_\alpha \Theta \cdot S &\rightrightarrows D_\alpha \Theta' \cdot S' \ .
\end{aligned}$$

DÉMONSTRATION: Par induction sur Θ, en inspectant les cas possibles de réduction, et en utilisant le fait que $\rightrightarrows$ est une congruence. Dans les cas où on réduit un redex, on utilise les lemmes de commutation 6.10 à 6.13. □

6.2.3 Confluence

On démontre maintenant que $\rightrightarrows$ a la propriété du diamant, en exhibant un réduit commun à tous les réduits d'un terme : cette variante de la méthode de Tait–Martin-Löf est due à Takahashi [Tak95]. [1]

1. Merci à Pierre Lescanne, qui enseigne cette méthode dans ses cours à l'École Normale Supérieure de Lyon.

Définition 6.26 *On définit le réduit plein $\Theta\downarrow$ par induction sur Θ :*

$$\begin{aligned}
x\downarrow &= x \\
(\lambda x\, s)\downarrow &= \lambda x\, s\downarrow \\
(\mu\alpha\, c)\downarrow &= \mu\alpha\, c\downarrow \\
((D^n\lambda x\, s\cdot(u_1,\dots,u_n))\, T)\downarrow &= \left(\frac{\partial^n s\downarrow}{\partial x^n}\cdot(u_1\downarrow,\dots,u_n\downarrow)\right)[x := T\downarrow] \\
((D^n\mu\alpha\, c\cdot(u_1,\dots,u_n))\, T)\downarrow &= \mu\alpha\,((D^n_\alpha\, c\downarrow\cdot(u_1\downarrow,\dots,u_n\downarrow))_\alpha\, T\downarrow) \\
((D^n x\cdot(u_1,\dots,u_n))\, T)\downarrow &= (D^n x\cdot(u_1\downarrow,\dots,u_n\downarrow))\, T\downarrow \\
(D^n\lambda x\, s\cdot(u_1,\dots,u_n))\downarrow &= \lambda x\left(\frac{\partial^n s\downarrow}{\partial x^n}\cdot(u_1\downarrow,\dots,u_n\downarrow)\right) \\
(D^n\mu\alpha\, c\cdot(u_1,\dots,u_n))\downarrow &= \mu\alpha\,(D^n_\alpha\, c\downarrow\cdot(u_1\downarrow,\dots,u_n\downarrow)) \\
(D^n x\cdot(u_1,\dots,u_n))\downarrow &= D^n x\cdot(u_1\downarrow,\dots,u_n\downarrow) \\
([\alpha]\, s)\downarrow &= [\alpha]\, s\downarrow \\
0\downarrow &= 0 \\
(\theta+\Theta)\downarrow &= \theta\downarrow+\Theta\downarrow
\end{aligned}$$

Le réduit plein $\Theta\downarrow$ est obtenu à partir de Θ en réduisant simultanément tous ses redex. En particulier, c'est bien un réduit de Θ par $\rightrightarrows$:

Lemme 6.27 *Pour tout objet Θ, $\Theta \rightrightarrows \Theta\downarrow$.*

DÉMONSTRATION: C'est une conséquence directe des définitions. □

Le réduit plein est aussi un réduit commun à tous les autres réduits de Θ par $\rightrightarrows$:

Lemme 6.28 *Si $\Theta \rightrightarrows \Theta'$ alors $\Theta' \rightrightarrows \Theta\downarrow$.*

DÉMONSTRATION: Ce résultat se prouve en inspectant tous les cas possibles pour la réduction $\Theta \rightrightarrows \Theta'$, en utilisant le lemme 6.25 dans les cas correspondant à des redex : la preuve est en fait très proche de celle du lemme 6.25. □

Théorème 6.29 *La réduction $\to$ est confluente.*

DÉMONSTRATION: C'est un corollaire direct du lemme précédent, via les inclusions $\to \subset \rightrightarrows \subset \to^*$. □

6.3 Traduction dans les réseaux différentiels polarisés

6.3.1 Typage

On obtient un système de types simples pour le λμ-calcul différentiel en réunissant les règles de typage du λμ-calcul et celles du λ-calcul différentiel données dans [ER03]. Les jugements de typage sont ceux du λμ-calcul, c'est-à-dire de la forme $\Gamma \vdash s : A \mid \Delta$ ou $c : (\Gamma \vdash \Delta)$ où Γ est un environnement de

$$\frac{}{\Gamma, x : A \vdash x : A \mid \Delta} \qquad \frac{\Gamma, x : A \vdash s : B \mid \Delta}{\Gamma \vdash \lambda x\, s : A \to B \mid \Delta} \qquad \frac{c : (\Gamma \vdash \alpha : A, \Delta)}{\Gamma \vdash \mu\alpha\, c : A \mid \Delta}$$

$$\frac{\Gamma \vdash s : A \to B \mid \Delta \qquad \Gamma \vdash t : A \mid \Delta}{\Gamma \vdash (s)\, t : B \mid \Delta} \qquad \frac{\Gamma \vdash s : A \to B \mid \Delta \qquad \Gamma \vdash t : A \mid \Delta}{\Gamma \vdash Ds \cdot t : A \to B \mid \Delta}$$

$$\frac{\Gamma \vdash s : A \mid \alpha : A, \Delta}{[\alpha]\, s : (\Gamma \vdash \alpha : A, \Delta)}$$

$$\frac{}{\Gamma \vdash 0 : A \mid \Delta} \qquad \frac{\Gamma \vdash s : A \mid \Delta \qquad \Gamma \vdash S : A \mid \Delta}{\Gamma \vdash s + S : A \mid \Delta}$$

FIGURE 6.1 – Règles de typage simple pour le λμ-calcul différentiel.

variables et Δ est un environnement de noms. Les règles de typage sont données dans la figure la figure 6.1.

Dans [Vau07b], l'auteur démontre que les termes et commandes simplement typés du λμ-calcul différentiel normalisent fortement. [2]

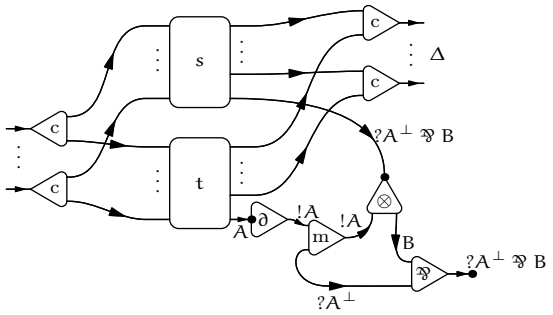
6.3.2 Traduction

La traduction s'établit selon les modalités de la définition 2.22 (page 42) augmentée de la règle suivante : à une dérivation se terminant par

$$\frac{\Gamma \vdash s : A \to B \mid \Delta \qquad \Gamma \vdash t : A \mid \Delta}{\Gamma \vdash Ds \cdot t : A \to B \mid \Delta}$$

on associe le RDP typé simple

$$[\![Ds \cdot t]\!]_\Gamma^\Delta =$$



.

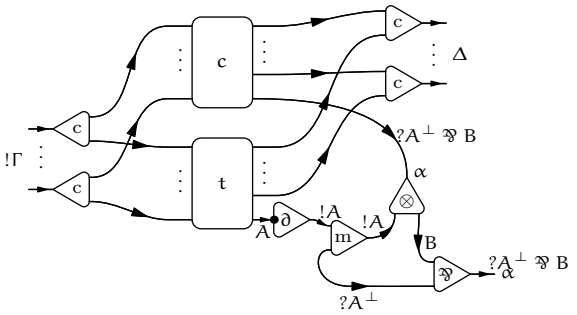
On étend cette traduction à tous les termes typés par :

$$\left[\!\!\left[\sum_{i=1}^{n} s_i \right]\!\!\right]_\Gamma^\Delta = \sum_{i=1}^{n} [\![s_i]\!]_\Gamma^\Delta .$$

2. Dans [Vau07b], on considère en fait des combinaisons linéaires formelles de termes simples, à coefficients dans un semi-anneau, et donc pas seulement des sommes : ceci impose de définir la réduction de manière légèrement différente. Il est toutefois facile d'adapter la preuve du cas où les coefficients sont des entiers naturels, au calcul présenté ici.

Définition 6.30 *Si* μ *et* ν *sont des SDP, on note la réduction modulo équivalence structurelle :* $\mu \longrightarrow_s \nu$ *ssi il existe* ν' *tel que* $\mu \longrightarrow_{[!]} \nu'$ *et* $\nu =_s \nu'$.

Lemme 6.31 *Pour toute commande simple* c *et tout terme simple* s *tels que* $c : (\Gamma \vdash \alpha : A \to B, \Delta)$ *et* $\Gamma \vdash s : A \mid \Delta$, *on a*

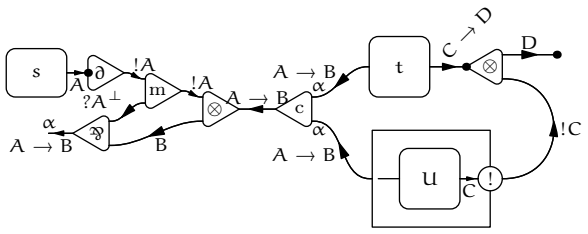


$$\longrightarrow_s^* [\![D_\alpha c \cdot s]\!]_\Gamma^{\alpha:A\to B,\Delta}.$$

Démonstration: On démontre ce résultat par récurrence sur c, en même temps qu'un résultat similaire pour les termes. Les deux seuls cas non directs sont celui de l'application et celui de la dérivée. On détaille celui de l'application : celui-ci fait intervenir intensivement l'équivalence structurelle, et permet de faire le parallèle entre la linéarisation à la volée réalisée dans $D_\alpha(t)\, u \cdot s$ et la loi de la chaîne (figure 5.1, page 78). Supposons donc qu'on type

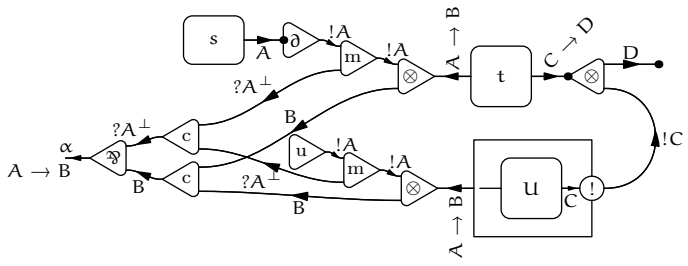
$$\frac{\Gamma \vdash t : C \to D \mid \alpha : A \to B, \Delta \qquad \Gamma \vdash u : C \mid \alpha : A \to B, \Delta}{\Gamma \vdash (t)\, u : D \mid \alpha : A \to B, \Delta}$$

et $\Gamma \vdash s : A \mid \Delta$, et considérons le RDP simple typé : [3]



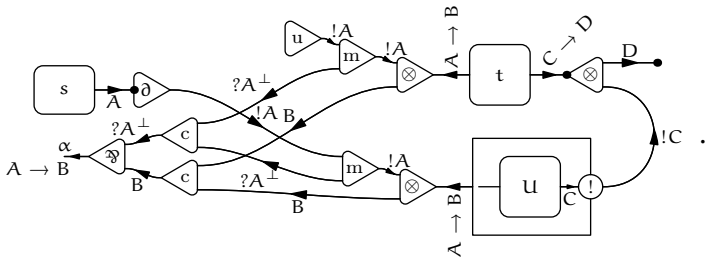
qui est une simple déformation du réseau à considérer. Par réduction simple, puis de routage on aboutit à la somme $\mu_1 + \mu_2$ des deux réseaux simples

$\mu_1 =$

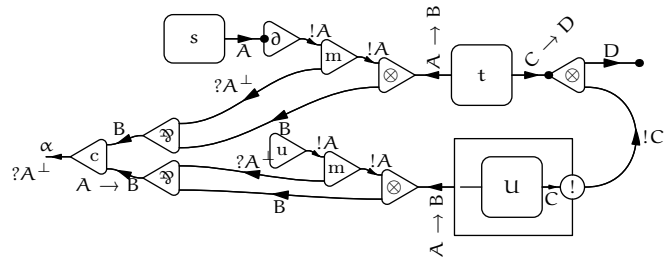


3. Afin de ne pas ployer sous leur abondance, on ne fait pas figurer les fils et les contractions correspondant aux contextes Γ et Δ.

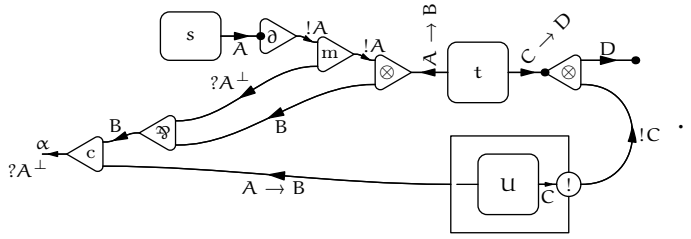
et

$\mu_2 =$ 

Par équivalence structurelle,

$\mu_1 =_s$ 

puis

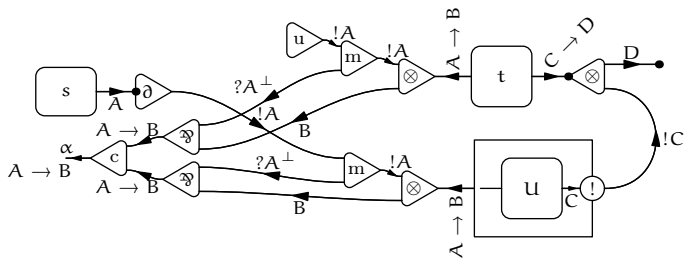
$\mu_1 =_s$ 

.

D'après l'hypothèse d'induction sur t, et par contextualité de $\longrightarrow_s$, on obtient

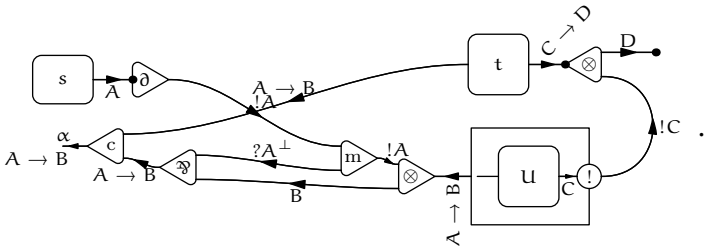
$$\mu_1 \longrightarrow_s^* [\![(D_\alpha t \cdot s)\, u]\!]_\Gamma^{\alpha:A\to B,\Delta} .$$

De même,

$\mu_2 =_s$ 

puis

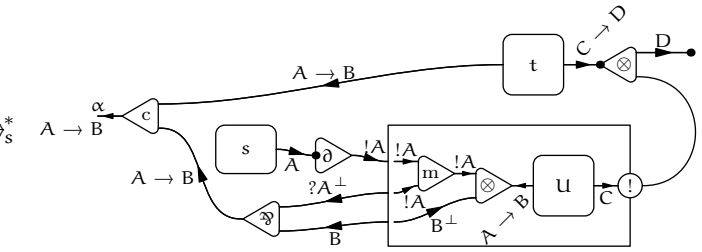
$\mu_2 =_s$



.

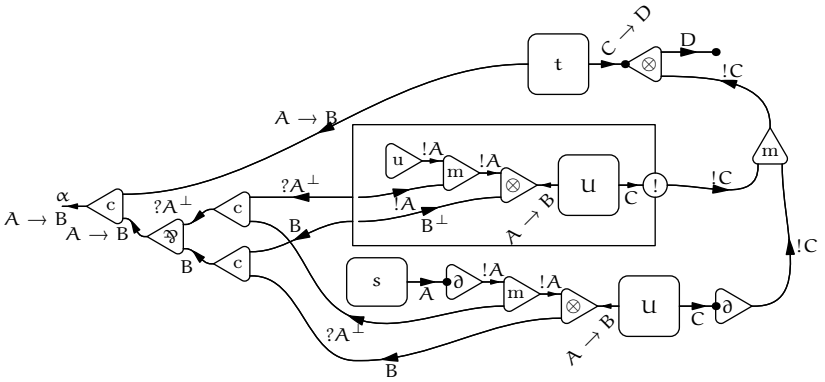
On a donc

$\mu_2 \longrightarrow_s^*$



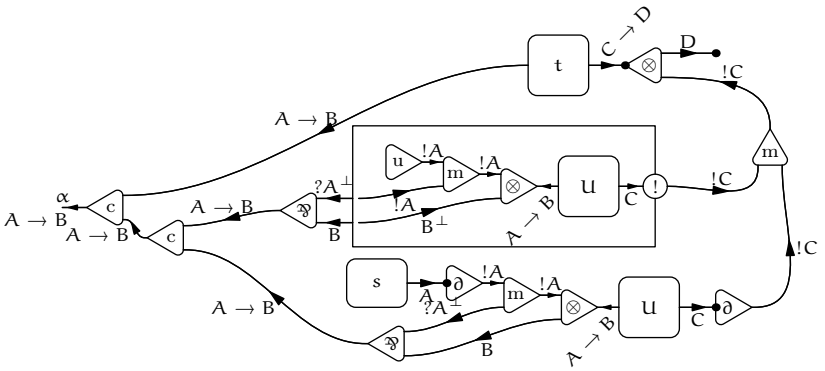
puis

$\mu_2 \longrightarrow_s^* \mu_2' =$



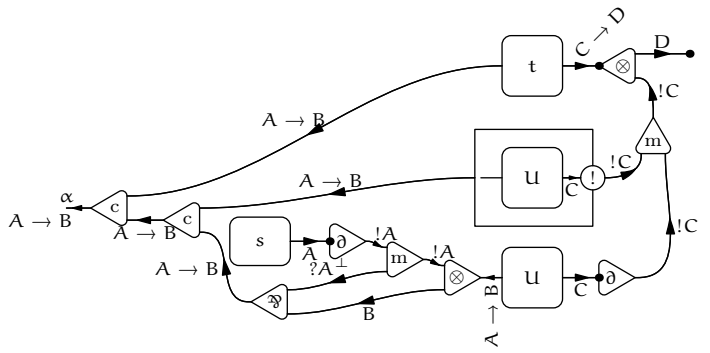
par la loi de la chaîne. De nouveau par équivalence structurelle,

$\mu_2' =_s$



et

$\mu_2' =_s$



.

D'après l'hypothèse d'induction sur U, et par contextualité de $\longrightarrow_s$, on obtient

$$\mu_2' \longrightarrow_s^* [\![(Dt \cdot (D_\alpha U \cdot s))\, U]\!]_\Gamma^{\alpha:A\to B,\Delta}.$$

□

Les lemmes 2.10, 2.23 et 2.36 s'adaptent facilement au cadre du λμ-calcul différentiel et des RDP. En particulier, la preuve de cette nouvelle version du lemme 2.36 fait également intervenir la linéarisation à la volée interprétée grâce à la loi de la chaîne.

Théorème 6.32 *La réduction des RDP typés simule la* β*-réduction du* λμ*-calcul différentiel, à équivalence structurelle près : si* $\Gamma \vdash S : A \mid \Delta$ *et* $S \to T$ *alors* $\Gamma \vdash T : A \mid \Delta$ *et* $[\![S]\!]_\Gamma^\Delta \longrightarrow_s^* [\![T]\!]_\Gamma^\Delta$.

DÉMONSTRATION: Par contextualité des deux réductions, il suffit de regarder le cas d'un redex en tête du terme. Si c'est un redex de la forme $(\lambda x\, s)\, t$ ou $D\lambda x\, s \cdot t$, on réduit la coupure $\langle \otimes , \parr \rangle$ correspondante et on applique respectivement le lemme 2.10 ou 2.36, dans leurs versions adaptées au λμ-calcul différentiel. Sinon, c'est un redex de la forme $(\mu\alpha\, c)\, t$ ou $D\mu\alpha\, c \cdot t$, et on applique directement le lemme 2.23 ou 6.31, en renommant le port interne $\bullet$ en α. □

6.3.3 Conclusion

Remarquons que l'interprétation du λμ-calcul différentiel dans les RDP ne fait aucunement usage des opérations costructurelles généralisées, c'est-à-dire des cellules m et u typées avec des types positifs non-exponentiels. C'est-à-dire que le λμ-calcul différentiel correspond plutôt à la réunion des RIP et des RID.

Considérons en effet les SI sur la signature $(\Sigma_{LL} \cup \Sigma_{Diff})^{[!]}$, avec le système de types donné par les figures 2.7 et 2.11 : on s'aperçoit que la réunion des règles de réduction des RIP et des RID avec boîtes définit un système de réduction complet. C'est en fait cette notion de réseaux qui est la cible de l'interprétation du λμ-calcul différentiel. Sous cet éclairage, on peut affirmer que les règles structurelles généralisées de LLP d'un côté, et les règles costructurelles des réseaux différentiels de l'autre côté, n'ont pas d'interaction véritable.

La réduction associée à $D\mu\alpha\, c \cdot s$ apparaît donc comme donnée gratuitement : l'interprétation de la dérivée et celle des opérateurs de contrôle semblent orthogonales, et parfaitement compatibles. Du point de vue logique, le λμ-calcul différentiel n'introduit donc rien de neuf : sa définition est simplement le témoin d'une compatibilité.

Chapitre 7

$\bar{\lambda}\mu$-calcul avec produit de convolution

Le $\bar{\lambda}\mu$-calcul de Herbelin transpose la correspondance de Curry-Howard établie par Parigot entre le $\lambda\mu$-calcul et la déduction naturelle classique, dans le cadre du calcul des séquents. En particulier, la notion d'application y est remplacée par la notion de coupure entre un terme et une pile d'arguments.

Si on inspecte la traduction du $\bar{\lambda}\mu$-calcul dans les réseaux polarisés (section 2.4.4, page 45), on remarque que l'interprétation des piles fait apparaître une conclusion positive (la négation négation linéaire d'un type simple, portée par le port distingué $\bullet$), contrairement à ce qui se passait pour le λ-calcul ou le $\lambda\mu$-calcul, où les interprétations en réseaux de tous les objets ne faisaient apparaître que des conclusions négatives (des types simples, et des exponentielles de la forme $?A^{\perp}$). Le $\bar{\lambda}\mu$-calcul est donc le candidat naturel pour tenter de faire émerger le sens calculatoire du produit de convolution sur les types positifs.

Ce chapitre reproduit en partie l'article [Vau07a], en corrigeant la preuve de confluence. On y définit une extension du $\bar{\lambda}\mu$-calcul, en introduisant un produit de convolution sur les piles. On en profite pour renforcer l'analogie avec le produit de convolution des distributions, déjà présente dans la sémantique des règles costructurelles sur les exponentielles (voir [Ehr01, Section 5.4] ou [Ehr05, Section 3]).

7.1 Syntaxe

Comme le $\bar{\lambda}\mu$-calcul usuel, le $\bar{\lambda}\mu$-calcul avec produit de convolution fait intervenir trois catégories syntaxiques : termes, piles et commandes. Le produit de convolution $*$ est une opération binaire sur les piles et 1 est la pile unité.

Comme au chapitre 6, chaque catégorie d'objets est munie d'une structure de monoïde commutatif : on note $+$ la loi correspondante et 0 l'objet nul pour chaque catégorie. Toutes les constructions sont linéaires, sauf l'empilement d'un terme sur une pile qui est seulement linéaire en la pile. En particulier, $*$ distribue sur $+$. Là encore, on réalise ces propriétés en donnant d'abord une syntaxe de

base, sur laquelle l'égalité entre objets se ramène à l'α-équivalence, puis en introduisant des notations plus générales.

7.1.1 Morphologie

Définition 7.1 *On définit les ensembles Λ (termes simples $s, t, u, \dots$), Λ^+ (termes $S, T, U, \dots$), $\mathcal{S}_0$ (piles atomiques $\sigma, \tau, \dots$), $\mathcal{S}$ (piles simples $e, f, \dots$), $\mathcal{S}^+$ (piles $E, F, \dots$), $\mathcal{C}$ (commandes simples $c, d, \dots$) et $\mathcal{C}^+$ (commandes $C, D, \dots$), par les grammaires mutuellement récursives suivantes :*

$$\begin{array}{rcl} s & ::= & x \mid \lambda x\, s \mid \mu\alpha\, c \\ \sigma & ::= & \alpha \mid S \cdot e \\ e & ::= & 1 \mid \sigma * e \\ c & ::= & \langle s \,,\, e \rangle \\ S & ::= & 0 \mid s + S \\ E & ::= & 0 \mid e + E \\ C & ::= & 0 \mid c + C. \end{array}$$

On considère les termes, piles et les commandes modulo permutativité de la somme : on identifie par exemple $s + (s' + S)$ et $s' + (s + S)$. On considère aussi les piles simples modulo permutativité du produit : par exemple, $\alpha * ((S \cdot e) * e') = (S \cdot e) * (\alpha * e')$. On appelle égalité permutative la relation d'équivalence ainsi définie sur chacun des ensembles précédents. L'égalité permutative préserve les variables libres et liées, elle est donc compatible avec l'α-conversion. La clôture transitive commune de l'α-équivalence et de l'égalité permutative est trivialement une relation d'équivalence : dans toute la suite, on considérera comme égaux deux objets identifiés par cette relation. [1]

7.1.2 Notations

On appelle objet simple tout terme simple, toute pile simple et toute commande simple, et objet tout terme, toute pile et toute commande. On utilisera la minuscule grecque θ pour dénoter un objet simple et la capitale Θ pour un objet. En général, si un objet simple θ et un objet Θ apparaissent dans le même contexte, il est sous-entendu qu'ils appartiennent au même genre syntaxique (terme, pile ou commande).

Soient $\theta_1, \dots, \theta_n$ des objets simples et Θ un objet, on abrège $\theta_1 + (\cdots + (\theta_n + \Theta) \cdots)$ en $\theta_1 + \cdots + \theta_n + \Theta$. Si θ est un objet simple, on pourra aussi écrire θ pour l'objet correspondant $\theta + 0$. Ainsi, tout objet Θ peut s'écrire sans ambiguïté $\Theta = \theta_1 + \cdots + \theta_n$ ou même $\Theta = \sum_{i=1}^{n} \theta_i$. Supposons que $\Theta = \theta_1 + \cdots + \theta_n$ et $\Theta' = \theta'_1 + \cdots + \theta'_p$: on note $\Theta + \Theta'$ l'objet $\theta_1 + \cdots + \theta_n + \theta'_1 + \cdots + \theta'_p$. On munit ainsi l'ensemble des termes (resp. des piles, des commandes) d'une

1. Comme à la section 2.4.4, on a repris les notations de Curien et Herbelin dans [CH00] pour les constructions du $\bar{\lambda}\mu$-calcul. En particulier, on note $S \cdot e$ pour la pile dont la tête est S et la queue e : il ne s'agit pas d'un produit et, surtout, ce n'est pas une opération permutative.

structure de monoïde commutatif, librement généré par les termes simples (resp. piles simples, commandes simples) pour la loi $+$ et le neutre 0.

De la même manière, si les $\sigma_1, \dots, \sigma_n$ sont des piles atomiques, et e une pile simple, on abrège $\sigma_1 * (\cdots * (\sigma_n * e) \cdots)$ en $\sigma_1 * \cdots * \sigma_n * e$. Si σ est une pile atomique, on écrit σ pour la pile simple $\sigma * 1$. Ainsi, toute pile simple e peut s'écrire sans ambiguïté $e = \sigma_1 * \cdots * \sigma_n$ ou même $e = \prod_{i=1}^{n} \sigma_i$. Supposons que $e = \sigma_1 * \cdots * \sigma_n$ et $e' = \sigma'_1 * \cdots * \sigma'_p$: on note $e * e'$ la pile simple $\sigma_1 * \cdots * \sigma_n * \sigma'_1 * \cdots * \sigma'_p$. L'ensemble des piles simples est alors le monoïde commutatif libre généré par les piles atomiques par $*$, avec l'unité 1.

On étend maintenant les constructions de la syntaxe par linéarité.

Définition 7.2 *Soient des objets* $s_1, \dots, s_n \in \Lambda$, $e_1, \dots, e_p, f_1, \dots, f_q \in \mathcal{S}$, $c_1, \dots, c_r \in \mathcal{C}$ *et* $S \in \Lambda^+$. *On pose alors*

$$\begin{aligned}
\lambda x \left(\sum_{i=1}^{n} s_i \right) &= \sum_{i=1}^{n} \lambda x\, s_i \\
\mu\alpha \left(\sum_{l=1}^{r} c_l \right) &= \sum_{l=1}^{r} \mu\alpha\, c_l \\
S \cdot \left(\sum_{j=1}^{p} e_j \right) &= \sum_{j=1}^{p} S \cdot e_j \\
\left(\sum_{j=1}^{p} e_j \right) * \left(\sum_{k=1}^{q} f_k \right) &= \sum_{j=1}^{p} \sum_{k=1}^{q} e_j * f_k \\
\left\langle \sum_{i=1}^{n} s_i \, , \sum_{j=1}^{p} e_j \right\rangle &= \sum_{i=1}^{n} \sum_{j=1}^{p} \langle s_i \, , e_j \rangle \, .
\end{aligned}$$

Notons que la position d'argument en tête d'une pile simple *n'est pas* linéaire. C'est l'analogue de la position d'argument d'une application dans le λ-calcul. Comme au chapitre précédent, les notations que nous venons d'introduire rendent la lecture d'un objet ambiguë, mais univoque : toutes les manières de lire une expression désignant un terme, une pile ou une commande mènent au même objet.

À ces notations près, l'ensemble des piles est en fait un semi-anneau : en effet, $(\mathcal{S}^+, +, 0)$ et $(\mathcal{S}^+, *, 1)$ sont des monoïdes commutatifs, le produit $*$ distribue sur $+$ et 0 est absorbant. Par ailleurs, les λ- et μ-abstractions sont linéaires, $\cdot$ est linéaire à droite, et la coupure est bilinéaire.

7.1.3 Substitutions

Comme dans le $\bar{\lambda}\mu$-calcul usuel, on introduit deux formes de substitution : la substitution d'un terme à une variable et la substitution d'une pile à un nom.

Définition 7.3 *La substitution* $\Theta[x := T]$ *est définie inductivement par :*

$$\begin{array}{rcl}
y[x := T] & = & \begin{cases} T & \textit{si } x = y \\ y & \textit{sinon} \end{cases} \\
(\lambda y\, s)[x := T] & = & \lambda y\, (s[x := T]) \\
(\mu\alpha\, c)[x := T] & = & \mu\alpha\, (c[x := T]) \\
\alpha[x := T] & = & \alpha \\
(S \cdot e)[x := T] & = & (S[x := T]) \cdot (e[x := T]) \\
1[x := T] & = & 1 \\
(\sigma * e)[x := T] & = & (\sigma[x := T]) * (e[x := T]) \\
\langle s\, , e\rangle[x := T] & = & \langle s[x := T]\, , e[x := T]\rangle \\
0[x := T] & = & 0 \\
(\theta + \Theta)[x := T] & = & \theta[x := T] + \Theta[x := T]
\end{array}$$

où $x \neq y$ *et* $y \notin \mathrm{VL}(T)$ *dans le cas de la* λ*-abstraction, et* $\alpha \notin \mathrm{NL}(T)$ *dans le cas de la* μ*-abstraction.*

Définition 7.4 *La substitution* $\Theta[\alpha := E]$ *est définie inductivement par :*

$$\begin{array}{rcl}
y[\alpha := E] & = & y \\
(\lambda y\, s)[\alpha := E] & = & \lambda y\, (s[\alpha := E]) \\
(\mu\beta\, c)[\alpha := E] & = & \mu\beta\, (c[\alpha := E]) \\
\beta[\alpha := E] & = & \begin{cases} E & \textit{si } \alpha = \beta \\ \beta & \textit{sinon} \end{cases} \\
(S \cdot e)[\alpha := E] & = & (S[\alpha := E]) \cdot (e[\alpha := E]) \\
1[\alpha := E] & = & 1 \\
(\sigma * e)[\alpha := E] & = & (\sigma[\alpha := E]) * (e[\alpha := E]) \\
\langle s\, , e\rangle[\alpha := E] & = & \langle s[\alpha := E]\, , e[\alpha := E]\rangle \\
0[\alpha := E] & = & 0 \\
(\theta + \Theta)[\alpha := E] & = & \theta[\alpha := E] + \Theta[\alpha := E]
\end{array}$$

où $y \notin \mathrm{VL}(E)$ *dans le cas de la* λ*-abstraction, et* $\alpha \neq \beta$ *et* $\beta \notin \mathrm{NL}(E)$ *dans le cas de la* μ*-abstraction.*

7.2 Traduction dans les réseaux différentiels polarisés

Plutôt que de donner immédiatement la définition de la réduction du $\bar{\lambda}\mu$-calcul avec produit de convolution, telle que dans [Vau07a], on présente la traduction de ses objets dans les RDP, afin de montrer comment la dynamique du calcul peut se déduire de celle des réseaux.

$$\frac{}{\Gamma, x : A \vdash x : A \mid \Delta} \qquad \frac{\Gamma, x : A \vdash s : B \mid \Delta}{\Gamma \vdash \lambda x\, s : A \to B \mid \Delta} \qquad \frac{c : (\Gamma \vdash \alpha : A, \Delta)}{\Gamma \vdash \mu\alpha\, c : A \mid \Delta}$$

$$\frac{}{\Gamma \mid \alpha : A \vdash \alpha : A, \Delta} \qquad \frac{\Gamma \vdash S : A \mid \Delta \qquad \Gamma \mid e : B \vdash \Delta}{\Gamma \mid S \cdot e : A \to B \vdash \Delta}$$

$$\frac{\Gamma \vdash s : A \mid \Delta \qquad \Gamma \mid e : A \vdash \Delta}{\langle s\,,\, e\rangle : (\Gamma \vdash \Delta)}$$

$$\frac{}{\Gamma \mid 1 : A \vdash \Delta} \qquad \frac{\Gamma \mid \sigma : A \vdash \Delta \qquad \Gamma \mid e : A \vdash \Delta}{\Gamma \mid \sigma * e : A \vdash \Delta}$$

$$\frac{}{\Gamma \vdash 0 : A \mid \Delta} \qquad \frac{\Gamma \vdash s : A \mid \Delta \qquad \Gamma \vdash S : A \mid \Delta}{\Gamma \vdash s + S : A \mid \Delta}$$

$$\frac{}{\Gamma \mid 0 : A \vdash \Delta} \qquad \frac{\Gamma \mid e : A \vdash \Delta \qquad \Gamma \mid E : A \vdash \Delta}{\Gamma \mid e + E : A \vdash \Delta}$$

$$\frac{}{0 : (\Gamma \vdash \Delta)} \qquad \frac{c : (\Gamma \vdash \Delta) \qquad C : (\Gamma \vdash \Delta)}{c + C : (\Gamma \vdash \Delta)}$$

FIGURE 7.1 – Règles de typage simple pour le $\bar{\lambda}\mu$-calcul avec produit de convolution.

7.2.1 Typage

On définit un système de types simples pour les objets du $\bar{\lambda}\mu$-calcul avec produit de convolution. Les jugements de typage sont ceux du $\bar{\lambda}\mu$-calcul, c'est-à-dire de la forme $\Gamma \vdash s : A \mid \Delta$, $\Gamma \mid e : A \vdash \Delta$ ou $c : (\Gamma \vdash \Delta)$ où Γ est un environnement de variables et Δ est un environnement de noms. Les règles de typage sont données dans la figure la figure 7.1.

7.2.2 Traduction

Rappelons que l'idée qui sous-tend l'introduction du $\bar{\lambda}\mu$-calcul avec produit de convolution est de fournir une interprétation dans un calcul de termes au produit de convolution sur les types positifs des RDP.

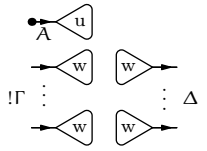
La traduction des objets simples s'établit selon les modalités de la définition 2.25 (page 45) augmentée des règles de la figure 7.2 : les constructions des piles atomiques du $\bar{\lambda}\mu$-calcul avec produit de convolution sont celles des piles du $\bar{\lambda}\mu$-calcul. On l'étend à tous les objets par :

$$\left[\!\!\left[\sum_{i=1}^{n} \theta_i \right]\!\!\right]_{\Gamma}^{\Delta} = \overline{\sum_{i=1}^{n}} [\![\theta_i]\!]_{\Gamma}^{\Delta}\,.$$

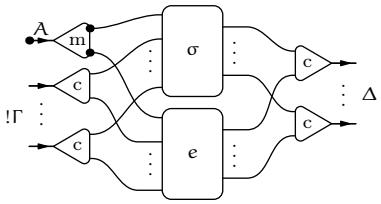
Notons que l'interprétation du $\bar{\lambda}\mu$-calcul avec produit de convolution dans les RDP n'utilise pas la cellule dérivée (∂).

Lemme 7.5 *Si* $\Gamma, x : A \vdash s : B \mid \Delta$, *et si* y *et* z *sont des variables fraîches*

$[\![1]\!]^{\Delta}_{\Gamma}$ =



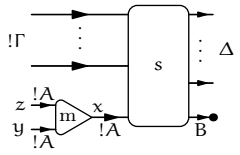
$[\![\sigma * e]\!]^{\Delta}_{\Gamma}$ =



.

FIGURE 7.2 – Traduction du produit de convolution des piles dans les RDP.

distinctes, alors le RDP simple typé



se réduit en au moins une étape en un RDP structurellement équivalent à $[\![s\,[x := y + z]]\!]^{\Delta}_{\Gamma,y:A,z:A}$.

DÉMONSTRATION: On démontre ce résultat en même temps que des résultats similaires pour les piles et commandes, par induction sur les objets : c'est évidemment la réduction de routage qui introduit les sommes. □

Les lemmes 2.10 (page 38) et 2.26 (page 47), ainsi que la deuxième partie du lemme 2.36 (page 53), s'adaptent facilement au cadre du $\bar{\lambda}\mu$-calcul avec produit de convolution et des RDP. De ces résultats, on peut déduire que :

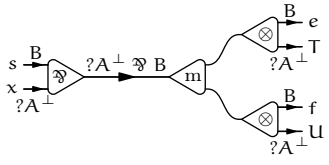
Corollaire 7.6 *Si* $\Gamma, x : A \vdash s : B \mid \Delta$, $\Gamma \vdash T : A \mid \Delta$, $\Gamma \vdash U : A \mid \Delta$, $\Gamma \mid e : B \vdash \Delta$ *et* $\Gamma \mid f : B \vdash \Delta$, *alors*

$$[\![\langle \lambda x\, s\,,\, (T \cdot e) * (U \cdot f)\rangle]\!]^{\Delta}_{\Gamma}$$

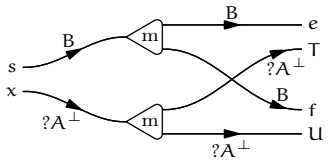
se réduit en au moins une étape en un RDP structurellement équivalent à

$$[\![\langle s\,[x := T + U]\,,\, e * f\rangle]\!]^{\Delta}_{\Gamma}\,.$$

DÉMONSTRATION: On applique le lemme précédent, composé avec le lemme 2.10, après avoir réduit



en



dans $[\![\langle \lambda x\, s\, ,\, (T \cdot e) * (U \cdot f)\rangle]\!]^{\Delta}_{\Gamma}$ (on a nommé les ports libres afin de guider l'intuition du lecteur). □

C'est-à-dire que la pile $(T \cdot e) * (U \cdot f)$ se comporte comme $(T + U) \cdot (e * f)$.

7.3 Réduction

D'après la section précédente, une définition naturelle de la réduction du $\bar{\lambda}\mu$-calcul avec produit de convolution pourrait être :

$$\langle \lambda x\, s\, ,\, (T \cdot e) * (U \cdot f)\rangle \to \langle s\,[x := T + U]\, ,\, e * f\rangle$$

ou de manière plus générale

$$\left\langle \lambda x\, s\, ,\, \prod_{i=1}^{n} (T_i \cdot e_i) \right\rangle \to \left\langle s \left[x := \sum_{i=1}^{n} T_i \right] ,\, \prod_{i=1}^{n} e_i \right\rangle$$

avec comme cas particulier

$$\langle \lambda x\, s\, ,\, 1\rangle \to \langle s\,[x := 0]\, ,\, 1\rangle\,.$$

En ce sens, le produit de convolution sur les piles apparaît comme une entrelacement de sommes sur les arguments. Cette notion de réduction a toutefois le défaut d'imposer une contrainte de synchronisation : on ne réduit que si toutes les piles d'un produit ont un argument en tête. On va en donner une version plus fine, qui ne présente pas cette restriction, qu'on analysera ensuite en termes d'extensionnalité.

7.3.1 Définition

Définition 7.7 *On appelle relation contextuelle tout triplet τ de relations binaires, respectivement sur les termes, piles et commandes, également notées τ, et telles que :*

- *si $S\,\tau\,S'$ alors $\lambda x\, S\,\tau\,\lambda x\, S'$, $S \cdot E\,\tau\,S' \cdot E$ et $\langle S\, ,\, E\rangle\,\tau\,(S')\,E$;*
- *si $C\,\tau\,C'$ alors $\mu\alpha\, C\,\tau\,\mu\alpha\, C'$;*
- *si $E\,\tau\,E'$ alors $\langle S\, ,\, E\rangle\,\tau\,\langle S\, ,\, E'\rangle$, $S \cdot E\,\tau\,S \cdot E'$ et $E * F\,\tau\,E' * F$;*
- *et si $\theta_0\,\tau\,\Theta_0'$ alors $\theta_0 + \Theta_1\,\tau\,\Theta_0' + \Theta_1$.*

Définition 7.8 *La réduction $\to$ du $\bar{\lambda}\mu$-calcul avec produit de convolution est la plus petite relation contextuelle telle que :*

$$\langle \mu\alpha\, c\, ,\, e\rangle \to c\,[\alpha := e] \tag{7.1}$$

$$\langle \lambda x\, s\, ,\, (T \cdot e) * f\rangle \to \langle \lambda y\, \mu\beta\, \langle s\,[x := y + T]\, ,\, e * \beta\rangle\, ,\, f\rangle \tag{7.2}$$

$$\langle \lambda x\, s\, ,\, 1\rangle \to \langle s\,[x := 0]\, ,\, 1\rangle \tag{7.3}$$

avec y une variable fraîche et β un nom frais dans le cas (7.2).

Notons que le cas (7.2) peut aussi s'écrire

$$\langle \lambda x\, s\,,\, (T \cdot e) * f \rangle \to \langle \lambda x\, \mu\beta\, \langle s\,[x := x + T]\,,\, e * \beta \rangle\,,\, f \rangle$$

si $x \notin \mathrm{VL}(T)$. Par ailleurs, on a la suite de réductions :

$$\begin{aligned}
\langle \lambda x\, s\,,\, (T \cdot e) * (U \cdot f) \rangle &\to \langle \lambda x\, \mu\beta\, \langle s\,[x := x + T]\,,\, e * \beta \rangle\,,\, U \cdot f \rangle \\
&\to \langle \lambda x\, \mu\gamma\, \langle \mu\beta\, \langle s\,[x := x + T + U]\,,\, e * \beta \rangle\,,\, f * \gamma \rangle\,,\, 1 \rangle \\
&\to \langle \mu\gamma\, \langle \mu\beta\, \langle s\,[x := T + U]\,,\, e * \beta \rangle\,,\, f * \gamma \rangle\,,\, 1 \rangle \\
&\to \langle \mu\beta\, \langle s\,[x := T + U]\,,\, e * \beta \rangle\,,\, f \rangle \\
&\to \langle s\,[x := T + U]\,,\, e * f \rangle
\end{aligned}$$

qui décompose l'étape de réduction « grossière » envisagée plus haut.

On peut tout de suite montrer que la réduction qu'on vient de définir correspond bien à celle des réseaux.

Théorème 7.9 *La réduction des RDP simule celle du $\bar{\lambda}\mu$-calcul avec produit de convolution, au sens du théorème 6.32 (page 103).*

DÉMONSTRATION: Par contextualité, il suffit de regarder le cas d'un redex. Pour $\langle \mu\alpha\, c\,,\, e \rangle$, on applique directement le lemme 2.26, en renommant le port interne $\bullet$ en α. Pour $\langle \lambda x\, s\,,\, 1 \rangle$, on réduit la coupure $\langle ⅋\,,\, u \rangle$ puis on applique la deuxième partie du lemme 2.36. Enfin pour $\langle \lambda x\, s\,,\, (T \cdot e) * f \rangle$, on réduit la coupure $\langle ⅋\,,\, m \rangle$, puis $\langle ⅋\,,\, \otimes \rangle$, et on applique successivement le lemme 7.5 avec des variables fraîches y et z, puis le lemme 2.10 pour produire le RDP associé à

$$\langle \lambda y\, \mu\alpha\, \langle s\,[x := y + z]\,[z := T]\,,\, e * \alpha \rangle\,,\, f \rangle \quad .$$

□

Définition 7.10 *On appelle congruence tout triplet τ de relations, respectivement sur les termes, piles et commandes, chacune étant réflexive et linéaire et également notée τ, telles que si $S\ \tau\ S'$, $E\ \tau\ E'$, $F\ \tau\ F'$, et $C\ \tau\ C'$, alors*

$$\begin{aligned}
\lambda x\, S &\ \tau\ \lambda x\, S' \\
\mu\alpha\, C &\ \tau\ \mu\alpha\, C' \\
S \cdot E &\ \tau\ S' \cdot E' \\
E * F &\ \tau\ E' * F' \\
\langle S\,,\, E \rangle &\ \tau\ \langle S'\,,\, E' \rangle\,.
\end{aligned}$$

Comme au chapitre précédent, on a le résultat suivant :

Lemme 7.11 *La fermeture réflexive et transitive d'une relation contextuelle est une congruence.*

On note $\to^*$ la fermeture réflexive et transitive de $\to$, qui est donc une congruence. On a déjà établi dans le détail que

$$\left\langle \lambda x\, s\, ,\, \prod_{i=1}^{n} (T_i \cdot e_i) \right\rangle \to^* \left\langle s \left[x := \sum_{i=1}^{n} T_i \right] ,\, \prod_{i=1}^{n} e_i \right\rangle$$

avec $n = 2$. Le cas général se démontre sans difficulté. En particulier, la réduction du $\bar{\lambda}\mu$-calcul avec produit de convolution simule celle du $\bar{\lambda}\mu$-calcul :

$$\langle \lambda x\, s\, ,\, S \cdot e \rangle \to^* \langle s\,[x := S]\, ,\, e \rangle\, .$$

7.3.2 Notions d'η-expansion

Rappelons que l'η-*expansion* en λ-calcul est donnée par : $s \leftarrow_\eta \lambda x\,(s)\,x$ où $x \notin \mathrm{VL}(s)$. Elle correspond à l'introduction de l'extensionnalité dans l'égalité induite par la réduction. De même en $\bar{\lambda}\mu$-calcul, on peut poser

$$s \leftarrow_\eta \lambda x\, \mu\alpha\, \langle s\, ,\, x \cdot \alpha \rangle$$

où $x \notin \mathrm{VL}(s)$ et $\alpha \notin \mathrm{NL}(s)$. En effet, pour toute pile e et tous termes s et t, si $x \notin \mathrm{VL}(s)$ et $\alpha \notin \mathrm{NL}(s)$, on a $\langle \lambda x\, \mu\alpha\, \langle s\, ,\, x \cdot \alpha \rangle\, ,\, t \cdot e \rangle \to^* \langle s\, ,\, t \cdot e \rangle$.

Dans le cadre du $\bar{\lambda}\mu$-calcul avec produit de convolution, on peut également définir une notion d'extensionnalité partielle.

Définition 7.12 *On note $\leftarrow'_\eta$ la plus petite relation contextuelle telle que :*

$$\langle s\, ,\, e * e' \rangle \leftarrow'_\eta \langle \lambda x\, \mu\alpha\, \langle s\, ,\, e * (x \cdot \alpha) \rangle\, ,\, e' \rangle \quad .$$

où x est une variable fraîche et α un nom frais.

Il s'agit bien d'une notion d'extensionnalité : pour toutes piles E, F et tous termes S et T, si x et α sont frais, on a

$$\langle \lambda x\, \mu\alpha\, \langle S\, ,\, E * (x \cdot \alpha) \rangle\, ,\, T \cdot F \rangle \to^* \langle S\, ,\, E * (T \cdot F) \rangle = \langle S\, ,\, E * F \rangle \quad .$$

C'est-à-dire qu'on considère extensionnellement un seul facteur du produit de piles. Cette extensionnalité partielle permet d'offrir un autre point de vue sur le cas (7.2) de la définition de $\to$. En effet,

$$\langle \lambda x\, s\, ,\, (T \cdot e) * f \rangle \leftarrow'_\eta \langle \lambda y\, \mu\alpha\, \langle \lambda x\, s\, ,\, (T \cdot e) * (y \cdot \alpha) \rangle\, ,\, f \rangle$$

qui se réduit en

$$\langle \lambda y\, \mu\alpha\, \langle s\,[x := y + T]\, ,\, e * \alpha) \rangle\, ,\, f \rangle$$

par la réduction déduite du corollaire 7.6. On peut donc voir la version « fine » de la réduction comme un cas particulier de la version « grossière », modulo un peu d'extensionnalité.

7.3.3 À propos de convolution

Le produit de convolution des distributions peut être formulé de la manière suivante (voir [Sch66, Formule (VI, 2 ; 6)]) :

Définition 7.13 *Si* e *et* f *sont des distributions dont l'une au moins est à support compact, alors il existe une distribution* $e*f$ *uniquement déterminée telle que, pour toute fonction de test* φ*,*

$$\langle \lambda z\, \varphi(z)\, ,\, e * f \rangle = \langle \lambda y\, \langle \lambda x\, \varphi(x+y)\, ,\, e \rangle\, ,\, f \rangle \quad .$$

Ici, on reprend des notations proches du λ-calcul pour expliciter les changements de variables, et on écrit $\langle \varphi\, ,\, e \rangle$ pour $e(\varphi)$. Ceci va nous permettre d'expliciter une analogie formelle avec le $\bar{\lambda}\mu$-calcul avec produit de convolution.

En effet, remarquons que les deux commandes simples

$$\langle \lambda z\, s\, ,\, (S \cdot e) * (T \cdot f) \rangle$$

et

$$\langle \lambda y\, \mu\beta \langle \lambda x\, \mu\alpha \langle s\, [z := x+y]\, ,\, \alpha * \beta \rangle\, ,\, S \cdot e \rangle\, ,\, T \cdot f \rangle$$

se réduisent toutes les deux en $\langle s\, [z := S+T]\, ,\, e * f \rangle$. Cette identité est très semblable à celle de la définition 7.13, mais elle est plus lourde pour deux raisons.

D'abord, dans le formalisme des distributions et des fonctions de test, φ est à valeurs scalaires, et $\langle \varphi\, ,\, e \rangle$ est un scalaire. Dans le $\bar{\lambda}\mu$-calcul comme dans le λ-calcul, les termes représentent toujours des objets fonctionnels. D'où la présence des μ-abstractions et de la coupure la plus interne : celles-ci prennent en compte le fait que $s\, [z := x+y]$ n'est pas un scalaire, et peut encore attendre des arguments.

Ensuite, les fonctions sont en général considérée extensionnellement : en λ-calcul, ceci revient à considérer les termes modulo η-expansion. Ainsi, dans la formulation qu'on a donnée, on a fait figurer des arguments en tête de pile. On peut observer que l'expansion $\leftarrow'_\eta$ introduit l'extensionnalité suffisante pour se passer d'arguments de tête. On a en effet :

$$\begin{array}{rcl} \langle \lambda z\, s\, ,\, e * f \rangle & \leftarrow'_\eta & \langle \lambda y\, \mu\beta\, \langle \lambda z\, s\, ,\, e * (y \cdot \beta) \rangle\, ,\, f \rangle \\ & \to & \langle \lambda y\, \mu\beta \langle \lambda x\, \mu\alpha \langle s\, [z := x+y]\, ,\, \alpha * \beta \rangle\, ,\, e \rangle\, ,\, f \rangle \end{array}$$

ce qui rend l'analogie avec les distributions assez frappante.

7.3.4 Confluence

On démontre à nouveau la confluence de ce calcul en définissant une version parallèle $\rightrightarrows$ de la réduction $\to$ qui possède la propriété du diamant. Cette section corrige [Vau07a], où la réduction parallèle, contrairement à ce qui était annoncé, ne vérifiait pas cette propriété.

Définition 7.14 *La réduction parallèle* $\rightrightarrows$ *du* $\bar{\lambda}\mu$*-calcul avec produit de convolution est la plus petite congruence telle que, si* $s \rightrightarrows S'$, $c \rightrightarrows C'$, $e \rightrightarrows E'$, *et pour tout* $i = 0, \ldots, n$, $S_i \rightrightarrows S'_i$ *et* $e_i \rightrightarrows E'_i$, *alors :*

$$\langle \mu\alpha\, c\, ,\, e\rangle \;\rightrightarrows\; C'[\alpha := E'] \tag{7.4}$$

$$\left\langle \lambda x\, s\, ,\, e * \prod_{i=0}^{n} (S_i \cdot e_i) \right\rangle \;\rightrightarrows\; \left\langle \lambda x\, \mu\alpha \left\langle S' \left[x := x + \sum_{i=0}^{n} S'_i \right] ,\, \alpha * \prod_{i=0}^{n} E'_i \right\rangle ,\, E' \right\rangle \tag{7.5}$$

$$\left\langle \lambda x\, \mu\alpha\, c\, ,\, e * \prod_{i=1}^{n} (S_i \cdot e_i) \right\rangle \;\rightrightarrows\; \left\langle \lambda x\, \mu\alpha \left(C' \left[x := x + \sum_{i=1}^{n} S'_i \right] \left[\alpha := \alpha * \prod_{i=1}^{n} E'_i \right] \right) ,\, E' \right\rangle \tag{7.6}$$

$$\left\langle \lambda x\, s\, ,\, \prod_{i=1}^{n} (S_i \cdot e_i) \right\rangle \;\rightrightarrows\; \left\langle S' \left[x := \sum_{i=1}^{n} S'_i \right] ,\, \prod_{i=1}^{n} E'_i \right\rangle \tag{7.7}$$

$$\left\langle \lambda x\, \mu\alpha\, c\, ,\, \prod_{i=1}^{n} (S_i \cdot e_i) \right\rangle \;\rightrightarrows\; C' \left[x := \sum_{i=1}^{n} S'_i \right] \left[\alpha := \prod_{i=1}^{n} E'_i \right] . \tag{7.8}$$

où α *est un nom frais dans (7.5), et avec les précautions d'usage pour éviter la capture de variables ou de noms.*

Les cas (7.6) et (7.8) sont essentiels : ils permettent de rattraper en un pas de réduction parallèle la création d'une μ-abstraction dans le cas (7.5). Sans cette précaution, le lemme 7.18 ne saurait être valide (c'est ce qui avait échappé à [Vau07a]). Insistons sur le fait que l'index démarre à 0 dans le cas (7.5) : ceci pour assurer l'inclusion $\rightrightarrows \subset \to^*$. Le lemme suivant établit la propriété attendue de $\rightrightarrows$.

Lemme 7.15 *Si* Θ *et* Θ' *sont des objets,* S *et* $S' \in \Lambda^+$, *et* E *et* $E' \in \mathcal{S}^+$, *tels que* $\Theta \rightrightarrows \Theta'$, $S \rightrightarrows S'$ *et* $E \rightrightarrows E'$, *alors pour toute variable* x *et tout nom* α, *on a :*

$$\Theta[x := S] \;\rightrightarrows\; \Theta'[x := S']$$
$$\Theta[\alpha := E] \;\rightrightarrows\; \Theta'[\alpha := E'] \ .$$

DÉMONSTRATION: Par induction sur Θ, en inspectant les cas possibles de réduction. □

Comme au chapitre précédent, on démontre que $\rightrightarrows$ a la propriété du diamant en exhibant un réduit commun à tous les réduits d'un terme. La définition de ce réduit plein étant assez complexe, on explicite d'abord une notion de *plus grand pas de réduction parallèle* (sans se préoccuper du passage au contexte).

Définition 7.16 *Soient* $s \in \Lambda$ *et* $e \in \mathcal{S}$. *On écrit* $e = (\prod_{j=1}^{p} \alpha_j) * \prod_{i=1}^{n} (S_i \cdot e_i)$.

S'il existe $c \in \mathcal{C}$ *tel que* $s = \mu\alpha\, c$, *alors on pose :*

$$\langle \lambda x\, s\, ,\, e\rangle_0 = \begin{cases} c\left[x := \sum_{i=1}^{n} S_i\right]\left[\alpha := \prod_{i=1}^{n} e_i\right] & \text{si } p = 0\,; \\ \left\langle \lambda x\, \mu\alpha \left(c\left[x := x + \sum_{i=1}^{n} S_i\right]\left[\alpha := \alpha * \prod_{i=1}^{n} e_i\right]\right), \prod_{j=1}^{p} \alpha_j \right\rangle & \text{sinon.} \end{cases}$$

Sinon, on pose :

$$\langle \lambda x\, s\, ,\, e\rangle_0 = \begin{cases} \left\langle s\left[x := \sum_{i=1}^{n} S_i\right], \prod_{i=1}^{n} e_i \right\rangle & \text{si } p = 0\,; \\ \left\langle \lambda x\, \mu\alpha \left\langle s\left[x := x + \sum_{i=1}^{n} S_i\right], \alpha * \prod_{i=1}^{n} e_i \right\rangle, \prod_{j=1}^{p} \alpha_j \right\rangle & \\ \qquad \text{où } \alpha \text{ est un nom frais, si } p \neq 0 \text{ et } n \neq 0\,; & \\ \langle \lambda x\, s\, ,\, e\rangle & \text{si } p \neq 0 \text{ et } n = 0. \end{cases}$$

Cette définition s'étend naturellement par linéarité.

On a clairement $\langle \lambda x\, S\, ,\, E\rangle \rightrightarrows \langle \lambda x\, S\, ,\, E\rangle_0$.

Définition 7.17 *On définit le réduit plein* $\Theta\downarrow$ *par induction sur* Θ *:*

$$\begin{aligned} x\downarrow &= x \\ (\lambda x\, s)\big\downarrow &= \lambda x\, s\downarrow \\ (\mu\alpha\, c)\big\downarrow &= \mu\alpha\, c\downarrow \\ \alpha\downarrow &= \alpha \\ (S \cdot e)\big\downarrow &= S\downarrow \cdot e\downarrow \\ \langle x\, ,\, e\rangle\big\downarrow &= \langle x\, ,\, e\downarrow\rangle \\ \langle \lambda x\, s\, ,\, e\rangle\big\downarrow &= \begin{cases} \langle \lambda x\, \mu\alpha\, c\downarrow\, ,\, e\downarrow\rangle_0 & \text{si } s = \mu\alpha\, c \\ \langle \lambda x\, s\downarrow\, ,\, e\downarrow\rangle_0 & \text{sinon} \end{cases} \\ \langle \mu\alpha\, c\, ,\, e\rangle\big\downarrow &= c\downarrow[\alpha := e\downarrow] \\ 1\downarrow &= 1 \\ (\sigma * e)\big\downarrow &= \sigma\downarrow * e\downarrow \\ 0\downarrow &= 0 \\ (\theta + \Theta)\big\downarrow &= \theta\downarrow + \Theta\downarrow \end{aligned}$$

avec les précautions d'usage pour éviter la capture de variable ou de nom.

Le réduit plein $\Theta\downarrow$ est obtenu à partir de Θ en réduisant simultanément tous ses redex. En particulier, c'est bien un réduit de Θ.

Lemme 7.18 *Pour tout objet* Θ, $\Theta \rightrightarrows \Theta\downarrow$.

DÉMONSTRATION: C'est une conséquence directe des définitions. □

Le réduit plein est aussi un réduit commun à tous les autres réduits de Θ.

Lemme 7.19 *Si* $\Theta \rightrightarrows \Theta'$ *alors* $\Theta' \rightrightarrows \Theta{\downarrow}$.

DÉMONSTRATION: Ce résultat se prouve en inspectant tous les cas possibles pour la réduction $\Theta \rightrightarrows \Theta'$, en utilisant le lemme 7.15 dans les cas correspondant à des redex. Les règles (7.6) et (7.8) sont cruciales pour ce lemme : elles sont nécessaires pour prendre en compte la μ-abstraction introduite dans Θ' par (7.5), lorsque celle-ci n'apparaît pas dans $\Theta{\downarrow}$. □

Théorème 7.20 *La réduction* $\rightarrow$ *est confluente.*

DÉMONSTRATION: C'est un corollaire direct du lemme précédent, via les inclusions $\rightarrow \subset \rightrightarrows \subset \rightarrow^*$. □

7.4 Un calcul désynchronisé

On remarque que $S \cdot e$ a le même comportement calculatoire que $(S \cdot 1) * (0 \cdot e)$. Ceci suggère d'introduire les notations suivantes :

Définition 7.21 *Pour tout terme* S *et tout pile* E*, on pose* $S^! = S \cdot 1$ *et* $\uparrow E = 0 \cdot E$.

On peut alors restreindre les constructions de piles atomiques à partir des noms, aux deux formes précédentes. Ceci n'induit pas de perte d'expressivité, vu la remarque précédente. Ainsi, les piles deviennent des sortes de soupes, dans lesquelles nagent des noms, des arguments et des queues de piles : on s'est affranchi du lien de causalité entre la consommation d'un argument et l'accès à la suite de la pile.

Dans ce calcul, la règle de réduction :

$$\langle \lambda x\, s\, ,\, (T \cdot e) * f \rangle \rightarrow \langle \lambda x\, \mu\alpha\, \langle s\, [x := T]\, ,\, e * \alpha \rangle\, ,\, f \rangle$$

se décompose en les deux règles élémentaires :

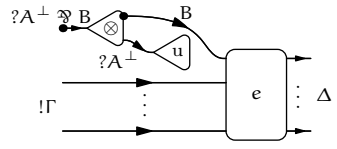
$$\begin{array}{rcl} \langle \lambda x\, s\, ,\, T^! * f \rangle & \rightarrow & \langle \lambda x\, s\, [x := x + T]\, ,\, f \rangle \\ \langle \lambda x\, s\, ,\, \uparrow e * f \rangle & \rightarrow & \langle \lambda x\, \mu\alpha\, \langle s\, ,\, e * \alpha \rangle\, ,\, f \rangle\, . \end{array}$$

Il est ainsi légitime de considérer un $\bar{\lambda}\mu$-calcul avec produit de convolution dans lequel $S^!$ et $\uparrow e$ ne sont plus des abréviations, mais bien des constructions de piles qui remplacent $S \cdot e$ dans la syntaxe. À cette syntaxe alternative, on associe un système de types avec les règles additionnelles

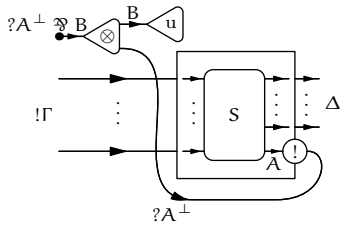
$$\frac{\Gamma \mid e : B \vdash \Delta}{\Gamma \mid \uparrow e : A \rightarrow B \vdash \Delta} \quad \text{et} \quad \frac{\Gamma \vdash S : A \mid \Delta}{\Gamma \mid S^! : A \rightarrow B \vdash \Delta}\, .$$

La traduction dans les RDP s'obtient alors comme auparavant, en remplaçant la clause concernant $S \cdot e$ par les deux suivantes :

$[\![\uparrow e]\!]^{\Delta}_{\Gamma}$ =

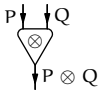


$[\![S^!]\!]^{\Delta}_{\Gamma}$ =

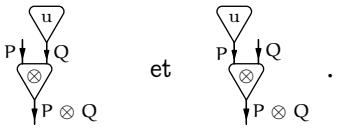


.

On vérifie sans difficulté que la réduction des réseaux simule encore celle du calcul, au sens du théorème 6.32 (page 103). La décomposition de $S \cdot e$ en $S^!$ et $\uparrow e$ revient à celle de

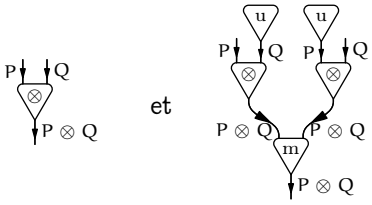


en

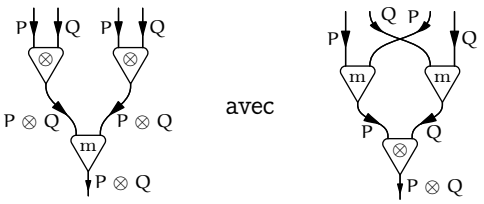


.

Plus précisément, l'égalité $S \cdot e = S^! * \uparrow e$ se lit dans les réseaux comme le fait que

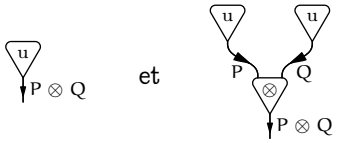


ont le même comportement dans la réduction. Modulo l'équivalence structurelle, on revient à l'identification de



(voir la section 3.4.2, page 65). De même, on peut remarquer que $\uparrow 1$ et 1 ont le

même comportement dans la réduction, ce qui renvoie à l'identification de



dans la sémantique et l'équivalence structurelle.

Au chapitre suivant, on met à profit cette décomposition pour introduire un $\bar{\lambda}\mu$-calcul différentiel, dans lequel les arguments linéaires (correspondant à la dérivée du $\lambda\mu$-calcul différentiel) ont un statut similaire à celui des arguments intuitionnistes (ceux de la forme $S^!$).

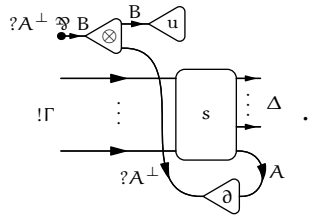
Chapitre 8

$\bar{\lambda}\mu$-calcul différentiel

Dans ce chapitre, on définit un $\bar{\lambda}\mu$-calcul différentiel avec produit de convolution qui met en jeu toute l'expressivité des réseaux différentiels polarisés. Aux constructions de piles atomiques $S^!$ et $\uparrow e$, on ajoute la formation de $[s]$, où s est un terme simple, qui constitue une version différentielle de $s^!$. On type

$$\frac{\Gamma \vdash s : A \mid \Delta}{\Gamma \mid [s] : A \to B \vdash \Delta}$$

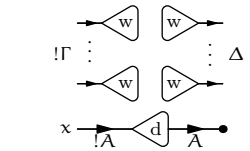
qu'on interprète par le RDP simple



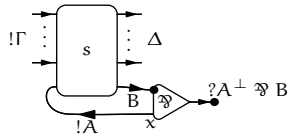
L'enjeu est de définir une extension de la réduction du $\bar{\lambda}\mu$-calcul avec produit de convolution qui corresponde à cette traduction. La présence de la dérivée interdit la duplication sans contrainte des piles au cours de la réduction : en particulier, la règle $\langle \mu\alpha\, c \,,\, e\rangle \to c\,[\alpha := e]$ ne saurait rester valide. On définit plutôt une opération $\langle c \,,\, e\rangle_\alpha$ telle que $\langle \mu\alpha\, c \,,\, e\rangle \to \langle c \,,\, e\rangle_\alpha$, au plus proche de la réduction des RDP : $\langle c \,,\, e\rangle_\alpha$ correspond au réduit du réseau obtenu en coupant le port $\bullet$ de la traduction de e sur le port α de la traduction de c.

Les règles de réduction obtenues font de $[s]$ un opérateur de dérivation dans la direction de s. Moralement, de même qu'on peut traduire $(s)\,T$ par $\mu\alpha\,\langle s \,,\, T \cdot \alpha\rangle$ ou $\mu\alpha\,\langle s \,,\, T^! * \uparrow\alpha\rangle$, on peut traduire $Ds \cdot t$ par $\mu\alpha\,\langle s \,,\, [T] * \alpha\rangle$ (noter l'absence du décalage $\uparrow$ dans ce dernier cas). Dans l'analogie avec les distributions, $s^!$ correspond à la masse de Dirac en s, et $[s]$ à sa dérivée. Le décalage $\uparrow e$ est l'artefact de la fonctionnalité des termes.

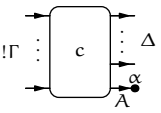
$$\frac{}{\Gamma, x : A \vdash x : A \mid \Delta}$$



$$\frac{\Gamma, x : A \vdash s : B \mid \Delta}{\Gamma \vdash \lambda x\, s : A \to B \mid \Delta}$$



$$\frac{c : (\Gamma \vdash \alpha : A, \Delta)}{\Gamma \vdash \mu\alpha\, c : A \mid \Delta}$$



$$\frac{}{\Gamma \vdash 0 : A \mid \Delta} \qquad 0$$

$$\frac{\Gamma \vdash s : A \mid \Delta \qquad \Gamma \vdash S : A \mid \Delta}{\Gamma \vdash s + S : A \mid \Delta} \qquad [\![s]\!]^{\Delta}_{\Gamma} + [\![S]\!]^{\Delta}_{\Gamma}$$

FIGURE 8.1 – Termes du $\bar{\lambda}\mu$-calcul différentiel : typage et traduction.

8.1 Objets

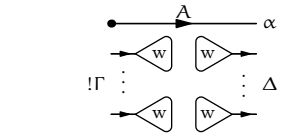
Définition 8.1 *On définit les ensembles Λ (termes simples $s, t, u, \dots$), Λ^+ (termes $S, T, U, \dots$), $\mathcal{S}_0$ (piles atomiques $\sigma, \tau, \dots$), $\mathcal{S}$ (piles simples $e, f, \dots$), $\mathcal{S}^+$ (piles $E, F, \dots$), $\mathcal{C}$ (commandes simples $c, d, \dots$) et $\mathcal{C}^+$ (commandes C, $D, \dots$), par les grammaires mutuellement récursives suivantes :*

$$\begin{array}{rcl} s & ::= & x \mid \lambda x\, s \mid \mu\alpha\, c \\ \sigma & ::= & \alpha \mid S^! \mid [s] \mid \uparrow e \\ e & ::= & 1 \mid \sigma * e \\ c & ::= & \langle s\, ,\, e \rangle \\ S & ::= & 0 \mid s + S \\ E & ::= & 0 \mid e + E \\ C & ::= & 0 \mid c + C \end{array}$$

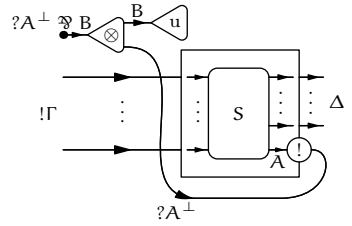
On donne immédiatement un système de types simples pour ces objets, ainsi que leur traduction dans les réseaux. Les jugements de typage sont de la même forme que ceux du $\bar{\lambda}\mu$-calcul avec produit de convolution, et la traduction s'établit de manière similaire : les règles de typage et de traduction sont rassemblées en figures 8.1 à 8.3. Les seules nouveautés par rapport au $\bar{\lambda}\mu$-calcul avec produit de convolution sont la désynchronisation annoncée à la dernière section du chapitre précédent, et l'introduction de $[s]$, qui fait intervenir la dérivée ∂.

Sans surprise, les RDP purement typés permettent d'interpréter tous les objets du calcul pur, par les mêmes règles de traduction. Remarquons que nous

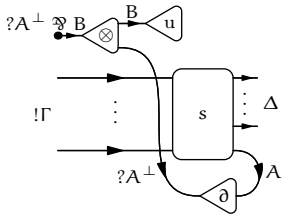
$$\frac{}{\Gamma \mid \alpha : A \vdash \alpha : A, \Delta}$$



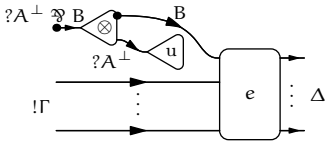
$$\frac{\Gamma \vdash S : A \mid \Delta}{\Gamma \mid S^{!} : A \to B \vdash \Delta}$$



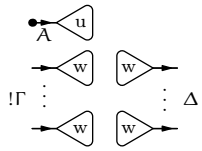
$$\frac{\Gamma \vdash s : A \mid \Delta}{\Gamma \mid [s] : A \to B \vdash \Delta}$$



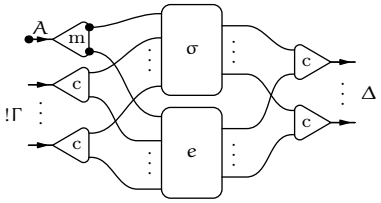
$$\frac{\Gamma \mid e : B \vdash \Delta}{\Gamma \mid \uparrow e : A \to B \vdash \Delta}$$



$$\frac{}{\Gamma \mid 1 : A \vdash \Delta}$$



$$\frac{\Gamma \mid \sigma : A \vdash \Delta \qquad \Gamma \mid e : A \vdash \Delta}{\Gamma \mid \sigma * e : A \vdash \Delta}$$



$$\frac{}{\Gamma \mid 0 : A \vdash \Delta}$$

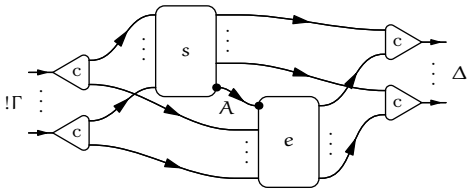
$$0$$

$$\frac{\Gamma \mid e : A \vdash \Delta \qquad \Gamma \mid E : A \vdash \Delta}{\Gamma \mid e + E : A \vdash \Delta}$$

$$[\![e]\!]^{\Delta}_{\Gamma} + [\![E]\!]^{\Delta}_{\Gamma}$$

FIGURE 8.2 – Piles du $\bar{\lambda}\mu$-calcul différentiel : typage et traduction.

$$\frac{\Gamma \vdash s : A \mid \Delta \qquad \Gamma \mid e : A \vdash \Delta}{\langle s\,,\, e\rangle : (\Gamma \vdash \Delta)}$$



$$\frac{}{0 : (\Gamma \vdash \Delta)}$$

$$0$$

$$\frac{c : (\Gamma \vdash \Delta) \qquad C : (\Gamma \vdash \Delta)}{c + C : (\Gamma \vdash \Delta)}$$

$$[\![c]\!]^{\Delta}_{\Gamma} + [\![C]\!]^{\Delta}_{\Gamma}$$

FIGURE 8.3 – Commandes du $\bar{\lambda}\mu$-calcul différentiel : typage et traduction.

avons utilisé toutes les cellules des SDP. On peut même, de manière assez informelle, associer à chaque construction des SDP séquentielles (figure 5.3, page 80) une construction de la syntaxe qui en fait un usage caractéristique, comme résumé ci-dessous :

axiome	$\rightsquigarrow$	nom α ;
coupure	$\rightsquigarrow$	commande $\langle s\,,\, e\rangle$;
par $\parr$	$\rightsquigarrow$	abstraction $\lambda x\, s$;
tenseur $\otimes$	$\rightsquigarrow$	décalage $\uparrow e$;
déréliction d	$\rightsquigarrow$	variable x ;
dérivée ∂	$\rightsquigarrow$	dérivée $[s]$;
boîte	$\rightsquigarrow$	boîte $S^!$;
produit m	$\rightsquigarrow$	produit $e * f$;
unité u	$\rightsquigarrow$	unité 1 ;
affaiblissement w et contraction c	$\rightsquigarrow$	μ-abstraction ;

quoique de manière détournée dans ce dernier cas (par équivalence structurelle).

À nouveau, on considère les termes, piles et commandes modulo α-équivalence et permutativité de la somme et du produit. On introduit également les notations permettant d'écrire $\lambda x\, S$, $\mu\alpha\, C$, $[S]$, $E * F$, $\uparrow E$ et $\langle S\,,\, E\rangle$, pour tout terme S, toute commande C, et toutes piles E et F, par linéarité de ces constructions ($S^!$ est la seule construction non-linéaire). On munit encore l'ensemble des termes (resp. piles, commandes) d'une structure de monoïde pour la somme et 0. L'ensemble des piles est de plus un semi-anneau avec la multiplication $*$ et l'unité 1.

8.2 Opérations

Comme en $\lambda\mu$-calcul différentiel, on introduit quatre opérations syntaxiques sur les objets : la substitution d'un terme à une variable et la substitution d'une

pile à un nom d'une part, et la dérivée partielle et la dérivée nommée d'autre part. On y ajoutera une cinquième opération, la coupure nommée $\langle c , e \rangle_\alpha$, qui sera le réduit typique de $\langle \mu\alpha\, c , e \rangle$.

Les deux substitutions sont définies de la manière évidente.

Définition 8.2 *La substitution* $\Theta\,[x := T]$ *est comme en définition 7.3 pour les cas pertinents, plus les clauses :*

$$\begin{array}{rcl} (\uparrow e)\,[x := T] & = & \uparrow (e\,[x := T]) \quad , \\ S^!\,[x := T] & = & (s\,[x := T])^! \quad \textit{et} \\ [s]\,[x := T] & = & [s\,[x := T]] \quad . \end{array}$$

La substitution $\Theta\,[\alpha := E]$ *est comme en définition 7.4 pour les cas pertinents, plus les clauses :*

$$\begin{array}{rcl} (\uparrow e)\,[\alpha := E] & = & \uparrow (e\,[\alpha := E]) \quad , \\ S^!\,[\alpha := E] & = & (S\,[\alpha := E])^! \quad \textit{et} \\ [s]\,[\alpha := E] & = & [s\,[\alpha := E]] \quad . \end{array}$$

Il est facile de voir que l'analogue du lemme 2.10 (page 38) reste valide : sa démonstration n'implique que les règles de réduction de la boîte face aux cellules structurelles c et w et la déréliction d. Par contre, on ne peut reproduire le lemme 2.26 (page 47) : la dérivée ∂ qui intervient dans la traduction de $[s]$ n'est pas dupliquée mais routée par les cellules structurelles. L'objectif de cette section est d'aboutir à la définition de la coupure nommée $\langle \Theta , E \rangle_\alpha$, qui remplacera la substitution $\Theta\,[\alpha := e]$ dans ce résultat.

On définit la dérivée partielle en gardant en tête qu'on souhaite établir l'analogue du lemme 2.36 (page 53).

Définition 8.3 *La dérivée partielle* $\frac{\partial\,\Theta}{\partial\,x}\cdot T$ *est définie par :*

$$\begin{aligned}
\frac{\partial\,y}{\partial\,x}\cdot T &= \delta_{x,y}T\\
\frac{\partial\,\lambda y\,s}{\partial\,x}\cdot T &= \lambda y\left(\frac{\partial\,s}{\partial\,x}\cdot T\right)\\
\frac{\partial\,\mu\alpha\,c}{\partial\,x}\cdot T &= \mu\alpha\left(\frac{\partial\,c}{\partial\,x}\cdot T\right)\\
\frac{\partial\,\alpha}{\partial\,x}\cdot T &= 0\\
\frac{\partial\,[s]}{\partial\,x}\cdot T &= \left[\frac{\partial\,s}{\partial\,x}\cdot T\right]\\
\frac{\partial\,S^{!}}{\partial\,x}\cdot T &= S^{!}*\left[\frac{\partial\,S}{\partial\,x}\cdot T\right]\\
\frac{\partial\uparrow e}{\partial\,x}\cdot T &= \uparrow\left(\frac{\partial\,e}{\partial\,x}\cdot T\right)\\
\frac{\partial\,1}{\partial\,x}\cdot T &= 0\\
\frac{\partial(\sigma * e)}{\partial\,x}\cdot T &= \left(\frac{\partial\,\sigma}{\partial\,x}\cdot T\right)*e+\sigma*\left(\frac{\partial\,e}{\partial\,x}\cdot T\right)\\
\frac{\partial\,\langle s\,,e\rangle}{\partial\,x}\cdot T &= \left\langle\frac{\partial\,s}{\partial\,x}\cdot T,e\right\rangle+\left\langle s\,,\frac{\partial\,e}{\partial\,x}\cdot T\right\rangle\\
\frac{\partial\,0}{\partial\,x}\cdot T &= 0\\
\frac{\partial}{\partial\,x}\left(\theta+\Theta\right)\cdot T &= \frac{\partial\,\theta}{\partial\,x}\cdot T+\frac{\partial\,\Theta}{\partial\,x}\cdot T
\end{aligned}$$

où $x\neq y$ *et* $y\notin \mathrm{VL}(T)$ *dans le cas de la* λ*-abstraction, et* $\alpha\notin \mathrm{NL}(T)$ *dans le cas de la* μ*-abstraction.*

Dans le cas de $S^{!}$, on voit apparaître une linéarisation semblable à celle du cas de l'application en $\lambda\mu$-calcul différentiel (définition 6.4, page 6.4). Celle-ci est bien entendu le reflet dans le calcul de la loi de la chaîne (figure 5.1, page 78). De même, l'apparition de sommes (pour le produit $\sigma * e$ et la commande $\langle s\,,e\rangle$) ou l'annihilation en 0 (dans le cas de la variable, du nom et de la pile unité 1), caractérisent la réduction de routage (figure 2.14, page 51).

De la même manière que pour la dérivée partielle, on définit la dérivée nommée en vue d'établir l'analogue du lemme 6.31 (page 100).

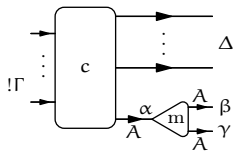
Définition 8.4 *La dérivée nommée* $D_\alpha\Theta \cdot T$ *est définie par :*

$$\begin{aligned}
D_\alpha x \cdot T &= 0 \\
D_\alpha \lambda x\, s \cdot T &= \lambda x\, (D_\alpha s \cdot T) \\
D_\alpha \mu\beta\, c \cdot T &= \mu\beta\, (D_\alpha c \cdot T) \\
D_\alpha \beta \cdot T &= \delta_{\alpha,\beta}\, (\alpha * [T]) \\
D_\alpha\, [s] \cdot T &= [D_\alpha s \cdot T] \\
D_\alpha S^! \cdot T &= S^! * [D_\alpha S \cdot T] \\
D_\alpha \uparrow e \cdot T &= \uparrow (D_\alpha e \cdot T) \\
D_\alpha 1 \cdot T &= 0 \\
D_\alpha (\sigma * e) \cdot T &= (D_\alpha \sigma \cdot T) * e + \sigma * (D_\alpha e \cdot T) \\
D_\alpha\, \langle s\, ,\, e \rangle \cdot T &= \langle D_\alpha s \cdot T, e \rangle + \langle s\, ,\, D_\alpha e \cdot T \rangle \\
D_\alpha 0 \cdot T &= 0 \\
D_\alpha (\theta + \Theta) \cdot T &= D_\alpha \theta \cdot T + D_\alpha \Theta \cdot T
\end{aligned}$$

où $\alpha \neq \beta$ *et* $\beta \notin NL(T)$ *dans le cas de la* μ*-abstraction, et* $x \notin VL(T)$ *dans le cas de la* λ*-abstraction.*

Là encore, la linéarisation à la volée dans le cas de $S^!$ caractérise la loi de la chaîne, et on voit l'effet de la réduction de routage dans l'apparition de sommes.

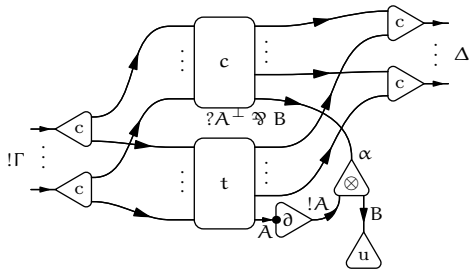
Lemme 8.5 *Si* $c : (\Gamma \vdash \alpha : A, \Delta)$*, et si* β *et* γ *sont des noms frais distincts, alors le RDP simple typé*



se réduit en au moins une étape en un RDP simple structurellement équivalent à $[\![c\,[\alpha := \beta * \gamma]]\!]_\Gamma^{\Delta, \beta:A, \gamma:A}$*.*

DÉMONSTRATION: On démontre ce résultat en même temps que des résultats similaires pour les piles et commandes, par induction sur les objets. Ici, seule la réduction structurelle (figure 2.12, page 48) intervient. □

Lemme 8.6 *Si* $c : (\Gamma \vdash \alpha : A \to B, \Delta)$*, et* $\Gamma \vdash t : A \mid \Delta$*, le RDP simple typé*



se réduit en un RDP structurellement équivalent à

$$[\![(D_\alpha c \cdot t)\,[\alpha := 1]]\!]^\Delta_\Gamma .$$

DÉMONSTRATION: On prouve ce résultat, en même temps que des résultats similaires pour les termes et piles, par induction sur les objets. Modulo équivalence structurelle, c'est l'analogue du lemme 6.31, où on efface la sortie α par une cellule unité u. □

On arrive enfin à la coupure nommée.

Définition 8.7 *On définit la coupure nommée* $\langle \Theta , e \rangle_\alpha$ *par :*

$$\langle \Theta , \beta \rangle_\alpha = \Theta\,[\alpha := \beta] \tag{8.1}$$
$$\langle \Theta , [s] \rangle_\alpha = (D_\alpha \Theta \cdot s)\,[\alpha := 1] \tag{8.2}$$
$$\left\langle \Theta , S^! \right\rangle_\alpha = \Theta\left[\alpha := S^!\right] \tag{8.3}$$
$$\langle \Theta , \uparrow e \rangle_\alpha = \langle \Theta\,[\alpha := \uparrow\alpha] , e \rangle_\alpha \tag{8.4}$$
$$\langle \Theta , 1 \rangle_\alpha = \Theta\,[\alpha := 1] \tag{8.5}$$
$$\langle \Theta , (\sigma * e) \rangle_\alpha = \left\langle \langle \Theta\,[\alpha := \beta * \alpha] , \sigma \rangle_\beta , e \right\rangle_\alpha \tag{8.6}$$

en supposant dans le dernier cas que β *est un nom frais. On étend cette définition par linéarité :* $\langle \Theta , 0 \rangle_\alpha = 0$ *et* $\langle \Theta , e + E \rangle_\alpha = \langle \Theta , e \rangle_\alpha + \langle \Theta , E \rangle_\alpha$.

Notons que si e ne contient pas de dérivée $[s]$, alors on retrouve $\langle \Theta , e \rangle_\alpha = \Theta\,[\alpha := e]$.

Lemme 8.8 *Si* $c : (\Gamma \vdash \alpha : A, \Delta)$ *et* $\Gamma \mid e : A \vdash \Delta$*, alors*

$$[\![\langle \mu\alpha\, c , e \rangle]\!]^\Delta_\Gamma$$

se réduit en un RDP structurellement équivalent à

$$[\![\langle c , e \rangle_\alpha]\!]^\Delta_\Gamma .$$

DÉMONSTRATION: La preuve est évidemment par induction sur e, avec :

- un renommage de port libre pour le cas (8.1) ;
- le lemme 8.5 dans le cas (8.6), et sa variante zéro-aire évidente pour (8.5) ;
- le lemme 8.6 pour le cas (8.2) ;
- une décomposition du lemme 2.26 (page 47) pour les cas (8.3) et (8.4).

□

8.3 Réduction

Définition 8.9 *On appelle relation contextuelle tout triplet* τ *de relations binaires, respectivement sur les termes, piles et commandes, également notées* τ*, vérifiant les clauses de la définition 7.7 plus :*

- *si* $S\ \tau\ S'$ *alors* $[S]\ \tau\ [S']$ *et* $S^!\ \tau\ S'^!$ *;*

— *si* $E \tau E'$ *alors* $\uparrow E \tau \uparrow E'$.

On appelle congruence toute relation contextuelle vérifiant en plus les clauses pertinentes de la définition 7.10.

La réduction du $\bar{\lambda}\mu$-calcul différentiel s'obtient à partir de celle du $\bar{\lambda}\mu$-calcul avec produit de convolution (dans sa version désynchronisée) en remplaçant, comme annoncé, la réduction

$$\langle \mu\alpha\, c\, ,\, e\rangle \to c\,[\alpha := e]$$

par

$$\langle \mu\alpha\, c\, ,\, e\rangle \to \langle c\, ,\, e\rangle_\alpha$$

et en ajoutant la règle de réduction

$$\langle \lambda x\, s\, ,\, [t] * f\rangle \to \left\langle \lambda x \left(\frac{\partial\, s}{\partial\, x} \cdot t\right) ,\, f \right\rangle$$

qui est une version $\bar{\lambda}\mu$-calculesque de celle du λ-calcul différentiel

$$D\lambda x\, s \cdot t \to \lambda x \left(\frac{\partial\, s}{\partial\, x} \cdot t\right).$$

Plus formellement :

Définition 8.10 *La réduction $\to$ est la plus petite relation contextuelle vérifiant :*

$$\begin{aligned}
\langle \mu\alpha\, c\, ,\, e\rangle &\to \langle c\, ,\, e\rangle_\alpha && (8.7)\\
\langle \lambda x\, s\, ,\, 1\rangle &\to \langle s\,[x := 0]\, ,\, 1\rangle && (8.8)\\
\left\langle \lambda x\, s\, ,\, S^! * f\right\rangle &\to \langle \lambda x\, s\,[x := S + x]\, ,\, f\rangle && (8.9)\\
\langle \lambda x\, s\, ,\, [t] * f\rangle &\to \left\langle \lambda x \left(\frac{\partial\, s}{\partial\, x} \cdot t\right) ,\, f \right\rangle && (8.10)\\
\langle \lambda x\, s\, ,\, \uparrow e * f\rangle &\to \langle \lambda x\, \mu\alpha\, \langle s\, ,\, e * \alpha\rangle\, ,\, f\rangle && (8.11)
\end{aligned}$$

où α est un nom frais dans le dernier cas, et avec les précautions évidentes pour éviter la capture de variables.

On note $\to^*$ la fermeture réflexive et transitive de $\to$, qui est une congruence (car $\to$ est contextuelle).

Théorème 8.11 *La réduction des RDP simule celle du $\bar{\lambda}\mu$-calcul différentiel, au sens du théorème 6.32 (page 103).*

DÉMONSTRATION: Ça ne devrait faire aucun doute : les définitions ont été choisies en ce sens. □

On démontre la confluence de $\to$ comme précédemment, par la méthode de Tait–Martin-Löf.

Définition 8.12 *La réduction parallèle $\rightrightarrows$ du $\bar{\lambda}\mu$-calcul différentiel est la plus petite congruence telle que les assertions suivantes soient vérifiées.*

1. *Si $c \rightrightarrows C'$ et $e \rightrightarrows E'$ alors*

$$\langle \mu\alpha\, c \,,\, e\rangle \rightrightarrows \langle C' \,,\, E'\rangle_\alpha \,. \tag{8.12}$$

2. *Supposons $s \rightrightarrows S'$, $e \rightrightarrows E'$, pour tout $i \in \{1,\dots,m\}$, $s_i \rightrightarrows S'_i$, pour tout $j \in \{1,\dots,n\}$, $T_j \rightrightarrows T'_j$, et pour tout $k \in \{0,\dots,p\}$, $e_k \rightrightarrows E'_k$. Supposons aussi que x n'est libre dans aucun des objets s_i, T_j ou e_k. Alors*

$$\left\langle \lambda x\, s \,,\, \prod_{i=1}^{m} [s_i] * \prod_{j=1}^{n} T_j^! * \prod_{k=1}^{p} \uparrow e_k \right\rangle \rightrightarrows \left\langle \left(\frac{\partial^m S'}{\partial x^m} \cdot (S'_1,\dots,S'_m) \right) \left[x := \sum_{j=1}^{n} T'_j \right] \,,\, \prod_{k=1}^{p} E'_k \right\rangle . \tag{8.13}$$

Si de plus α est un nom frais, alors

$$\left\langle \lambda x\, s \,,\, e * \prod_{i=1}^{m} [s_i] * \prod_{j=1}^{n} T_j^! * \prod_{k=0}^{p} \uparrow e_k \right\rangle \rightrightarrows \left\langle \lambda x\, \mu\alpha \left\langle \left(\frac{\partial^m S'}{\partial x^m} \cdot (S'_1,\dots,S'_m) \right) \left[x := x + \sum_{j=1}^{n} T'_j \right] \,,\, \alpha * \prod_{k=0}^{p} E'_k \right\rangle \,,\, E' \right\rangle . \tag{8.14}$$

Enfin, on a aussi

$$\left\langle \lambda x\, s \,,\, e * \prod_{i=1}^{m} [s_i] * \prod_{j=1}^{n} T_j^! \right\rangle \rightrightarrows \left\langle \lambda x \left(\left(\frac{\partial^m S'}{\partial x^m} \cdot (S'_1,\dots,S'_m) \right) \left[x := x + \sum_{j=1}^{n} T'_j \right] \right) \,,\, E' \right\rangle . \tag{8.15}$$

3. *Supposons $c \rightrightarrows C'$, $e \rightrightarrows E'$, pour tout $i \in \{1,\dots,m\}$, $s_i \rightrightarrows S'_i$, pour tout $j \in \{1,\dots,n\}$, $T_j \rightrightarrows T'_j$, et pour tout $k \in \{1,\dots,p\}$, $e_k \rightrightarrows E'_k$. Supposons aussi que x et α ne sont libres dans aucun des objets s_i, T_j ou e_k. Alors*

$$\left\langle \lambda x\, \mu\alpha\, c \,,\, \prod_{i=1}^{m} [s_i] * \prod_{j=1}^{n} T_j^! * \prod_{k=1}^{p} \uparrow e_k \right\rangle \rightrightarrows \left\langle \left(\frac{\partial^m C'}{\partial x^m} \cdot (S'_1,\dots,S'_m) \right) \left[x := \sum_{j=1}^{n} T'_j \right] \,,\, \prod_{k=1}^{p} E'_k \right\rangle_\alpha \,, \tag{8.16}$$

et

$$\left\langle \lambda x\, \mu\alpha\, c\ ,\ e * \prod_{i=1}^{m} [s_i] * \prod_{j=1}^{n} T_j^! * \prod_{k=1}^{p} \uparrow e_k \right\rangle \rightrightarrows \tag{8.17}$$
$$\left\langle \lambda x\, \mu\alpha \left\langle \left(\frac{\partial^m C'}{\partial x^m} \cdot (S'_1, \ldots, S'_m) \right) \left[x := x + \sum_{j=1}^{n} T'_j \right] , \alpha * \prod_{k=0}^{p} E'_k \right\rangle_{\alpha} , E' \right\rangle .$$

La complexité de cette dernière définition est à mettre en rapport avec le nombre de cas de réductions pour $\rightarrow$ (il y en cinq), mais aussi et surtout, comme au chapitre précédent, avec le fait que le cas (8.11) introduit des μ-abstraction qu'il faut pouvoir rattraper par la suite.

On montre que $\rightrightarrows$ a la propriété du diamant en exhibant un réduit commun à tous les $\rightrightarrows$-réduits d'un objet. On commence par définir un *plus grand pas de réduction parallèle*.

Définition 8.13 *Soient $s \in \Lambda$ et $e \in \mathcal{S}$. On écrit*

$$e = \prod_{i=1}^{m} [s_i] * \prod_{j=1}^{n} T_j^! * \prod_{k=1}^{p} \uparrow e_k * \prod_{l=1}^{q} \alpha_l .$$

S'il existe $c \in \mathcal{C}$ tel que $s = \mu\alpha\, c$, alors on pose :

$$\langle \lambda x\, s\ ,\ e \rangle_0 = \begin{cases} \left\langle \left(\dfrac{\partial^m c}{\partial x^m} \cdot (s_1, \ldots, s_m) \right) \left[x := \sum_{j=1}^{n} S_j \right] , \prod_{k=1}^{p} e_k \right\rangle_{\alpha} & \text{si } q = 0\ ; \\ \left\langle \lambda x\, \mu\alpha \left\langle \left(\dfrac{\partial^m c}{\partial x^m} \cdot (s_1, \ldots, s_m) \right) \left[x := x + \sum_{i=1}^{n} S_i \right] , \alpha * \prod_{k=1}^{p} e_k \right\rangle_{\alpha} , \prod_{l=q}^{p} \alpha_l \right\rangle & \text{sinon.} \end{cases}$$

Sinon, on pose :

$$\langle \lambda x\, s\ ,\ e \rangle_0 = \begin{cases} \left\langle \left(\dfrac{\partial^m s}{\partial x^m} \cdot (s_1, \ldots, s_m) \right) \left[x := \sum_{j=1}^{n} T_j \right] , \prod_{k=1}^{p} e_k \right\rangle & \text{si } q = 0\ ; \\ \left\langle \lambda x \left(\left(\dfrac{\partial^m s}{\partial x^m} \cdot (s_1, \ldots, s_m) \right) \left[x := x + \sum_{j=1}^{n} T_j \right] \right) , \prod_{l=1}^{q} \alpha_l \right\rangle & \text{si } q \neq 0 \text{ et } p = 0\ ; \\ \left\langle \lambda x\, \mu\alpha \left\langle \left(\dfrac{\partial^m s}{\partial x^m} \cdot (s_1, \ldots, s_m) \right) \left[x := x + \sum_{j=1}^{n} T_j \right] , \alpha * \prod_{k=1}^{p} e_k \right\rangle , \prod_{l=1}^{q} \alpha_l \right\rangle & \text{sinon.} \end{cases}$$

Cette définition s'étend naturellement par linéarité.

On a clairement $\langle \lambda x\, S\, ,\, E\rangle \rightrightarrows \langle \lambda x\, S\, ,\, E\rangle_0$. De plus, si S et E sont en forme normale, alors $\langle \lambda x\, S\, ,\, E\rangle_0$ est en forme normale.

Définition 8.14 *On définit le réduit plein $\Theta\downarrow$ par induction sur Θ :*

$$\begin{array}{rcl}
x\downarrow & = & x \\
(\lambda x\, s)\downarrow & = & \lambda x\, s\downarrow \\
(\mu\alpha\, c)\downarrow & = & \mu\alpha\, c\downarrow \\
\alpha\downarrow & = & \alpha \\
S^!\downarrow & = & (S\downarrow)^! \\
[s]\downarrow & = & [s\downarrow] \\
(\uparrow e)\downarrow & = & \uparrow(e\downarrow) \\
\langle x\, ,\, e\rangle\downarrow & = & \langle x\, ,\, e\downarrow\rangle \\
\langle \lambda x\, s\, ,\, e\rangle\downarrow & = & \begin{cases} \langle \lambda x\, \mu\alpha\, c\downarrow\, ,\, e\downarrow\rangle_0 & \textit{si } s = \mu\alpha\, c \\ \langle \lambda x\, s\downarrow\, ,\, e\downarrow\rangle_0 & \textit{sinon} \end{cases} \\
\langle \mu\alpha\, c\, ,\, e\rangle\downarrow & = & \langle c\downarrow\, ,\, e\downarrow\rangle_\alpha \\
1\downarrow & = & 1 \\
(\sigma * e)\downarrow & = & \sigma\downarrow * e\downarrow \\
0\downarrow & = & 0 \\
(\theta + \Theta)\downarrow & = & \theta\downarrow + \Theta\downarrow
\end{array}$$

avec les précautions d'usage pour éviter la capture de variable ou de nom.

Lemme 8.15 *Pour tout objet Θ, $\Theta \rightrightarrows \Theta\downarrow$.*

DÉMONSTRATION: C'est une conséquence directe des définitions. □

Lemme 8.16 *Si $\Theta \rightrightarrows \Theta'$ alors $\Theta' \rightrightarrows \Theta\downarrow$.*

DÉMONSTRATION: Ce résultat se prouve en inspectant tous les cas possibles pour la réduction $\Theta \rightrightarrows \Theta'$. □

Théorème 8.17 *La réduction $\to$ est confluente.*

DÉMONSTRATION: C'est un corollaire direct du lemme précédent, via les inclusions $\to \subset \rightrightarrows \subset \to^*$. □

8.4 Normalisation forte

On démontre la forte normalisabilité des objets typés par une méthode de réductibilité.

Emmanuel Polonowski a démontré dans sa thèse [Pol04] la normalisation forte d'une version avec substitutions explicites du $\bar{\lambda}\mu\tilde{\mu}$-calcul de Curien et Herbelin [CH00]. La première étape de sa preuve consiste à démontrer la normalisation forte du système avec substitutions implicites : on adapte à notre

calcul ses définitions des ensembles de candidats. Comme notre système n'a pas la symétrie termes-piles du $\bar{\lambda}\mu\tilde{\mu}$-calcul, on n'a toutefois pas besoin d'introduire l'ensemble de candidats associé à chaque type par un point fixe.

Définition 8.18 *. Si $\mathcal{X}$ est un ensemble d'objets (de termes, de piles ou de commandes), alors on note $\mathcal{X}^\Sigma$ l'ensemble des sommes d'objets de $\mathcal{X}$:*

$$\mathcal{X}^\Sigma = \left\{ \sum_{i=1}^{n} \Theta_i ;\ \forall i,\ \Theta_i \in \mathcal{X} \right\}.$$

Si $\mathcal{E} \subset \mathcal{S}^+$ est un ensemble de piles, alors on note $\mathcal{E}^\Pi$ l'ensemble des produits de piles de $\mathcal{E}$:

$$\mathcal{E}^\Pi = \left\{ \prod_{i=1}^{n} E_i ;\ \forall i,\ E_i \in \mathcal{E} \right\}.$$

En particulier $\mathcal{S} = \mathcal{S}_0^\Pi$ et par exemple $\Lambda^+ = \Lambda^\Sigma$.

Définition 8.19 *On appelle paquet un couple*

$$\phi = ([s_1, \dots, s_n], S) \in \mathcal{M}_{fin}(\Lambda) \times \Lambda^+$$

et, si $\mathcal{T}$ est un ensemble de termes simples, on note $\mathsf{paq}(\mathcal{T}) = \mathcal{M}_{fin}(\mathcal{T}) \times \mathcal{T}^\Sigma$ *l'ensemble des paquets dont les termes sont dans $\mathcal{T}$. Si $\phi = ([s_1, \dots, s_n], S)$ et Θ est un objet, on pose*

$$\Theta[\phi/x] = \left(\frac{\partial^n \Theta}{\partial x^n} \cdot (s'_1, \dots, s'_n) \right) [x := S]\,[x_0 := x]$$

où x_0 est une variable fraîche et $s'_i = s_i\,[x := x_0]$.

Ici on prend la précaution de renommer temporairement la variable impliquée dans la substitution, afin d'éviter les collisions. C'est un peu maladroit, mais la solution consistant à définir $\Theta[\phi/x]$ directement est également lourde.

Définition 8.20 *On note $\perp\!\!\!\perp$ l'ensemble des piles fortement normalisables. Si $\mathcal{T}$ est un ensemble de termes simples et $\mathcal{E}$ un ensemble de piles atomiques, on note*

$$\mathcal{T} \bullet \mathcal{E} = \mathcal{N} \cup \left\{ S^! ;\ S \in \mathcal{T}^\Sigma \right\} \cup \{[s] ;\ s \in \mathcal{T}\} \cup \left\{ \uparrow e ;\ e \in \mathcal{E}^\Pi \right\}$$

et

$$\begin{aligned} \mathcal{T} \Rightarrow \mathcal{E} \quad = \quad \mathcal{V} \quad & \cup \quad \left\{ \mu\alpha\, c ;\ \forall e \in (\mathcal{T} \bullet \mathcal{E})^\Pi,\ \langle c\,,\, e \rangle_\alpha \in \perp\!\!\!\perp \right\} \\ & \cup \quad \left\{ \lambda x\, s ;\ \forall \phi \in \mathsf{paq}(\mathcal{T}), \forall e \in \mathcal{E}^\Pi, \langle s\,[\phi/x]\,,\, e \rangle \in \perp\!\!\!\perp \right\}. \end{aligned}$$

Définition 8.21 *On définit les ensembles de candidats de réductibilité par induction sur les types simples :*

$$\begin{aligned} [\![X \vdash]\!] &= \mathcal{N} \\ [\![\vdash X]\!] &= \mathcal{V} \cup \{\mu\alpha\, c ;\ c \in \perp\!\!\!\perp\} \\ [\![A \to B \vdash]\!] &= [\![\vdash A]\!] \bullet [\![B \vdash]\!] \\ [\![\vdash A \to B]\!] &= [\![\vdash A]\!] \Rightarrow [\![B \vdash]\!] \end{aligned}$$

Lemme 8.22 *Tout terme simple* $s \in [\![\vdash A]\!]$ *(resp. toute pile atomique* $\sigma \in [\![A \vdash]\!]$*) est fortement normalisable.*

DÉMONSTRATION: Par induction sur A. Si A est atomique, c'est direct par définition. Pour $\sigma \in [\![A \to B \vdash]\!]$ l'hypothèse d'induction s'applique directement. Soit $s \in [\![\vdash A \to B]\!]$:

— si $s = x \in \mathcal{V}$, c'est direct ;
— si $s = \mu\alpha\, c$, comme $\alpha \in [\![A \to B \vdash]\!]$, on a par définition de $[\![\vdash A \to B]\!]$ $c = \langle c\,,\, \alpha\rangle_\alpha \in \perp\!\!\!\perp$;
— enfin si $s = \lambda x\, t$, comme $\alpha \in [\![B \vdash]\!]$ et $x \in [\![\vdash A]\!]$, on a $\langle t\,[x := x]\,,\, \alpha\rangle \in \perp\!\!\!\perp$ et donc t est fortement normalisable.

□

Lemme 8.23 *Si* $s \in [\![\vdash A]\!]$ *et* $s \to S'$ *alors* $S' \in [\![\vdash A]\!]^\Sigma$*. De même, si* $\sigma \in [\![A \vdash]\!]$ *et* $\sigma \to E'$ *alors* $E' \in [\![A \vdash]\!]^\Sigma$*.*

DÉMONSTRATION: Encore par induction sur A, en inspectant les cas de réduction. □

Lemme 8.24 *Si* $s \in [\![\vdash A]\!]$ *et* $e \in [\![A \vdash]\!]^\Pi$ *alors* $\langle s\,,\, e\rangle \in \perp\!\!\!\perp$*.*

DÉMONSTRATION: Par le lemme 8.22, on raisonne par induction sur la somme des longueurs maximales des réductions respectivement depuis s et e. On inspecte les cas de réduction possibles à partir de $\langle s\,,\, e\rangle$: si c'est une réduction dans s ou dans e, on conclut par l'hypothèse d'induction ; dans le cas des redex, on applique la définition de $[\![\vdash A]\!]$. □

Lemme 8.25 *Si* $s\,[\phi/x] \in [\![\vdash B]\!]^\Sigma$ *pour tout paquet* $\phi \in \mathsf{paq}\,([\![\vdash A]\!])$*, alors* $\lambda x\, s \in [\![\vdash A \to B]\!]$*.*

DÉMONSTRATION: Par les lemmes précédents. □

Théorème 8.26 *Soient des paquets*

$$\phi_1 \in \mathsf{paq}\,([\![\vdash A_1]\!]),\ \ldots,\ \phi_n \in \mathsf{paq}\,([\![\vdash A_n]\!]),$$

et des piles

$$e_1 \in [\![B_1 \vdash]\!]^\Pi,\ \ldots,\ e_m \in [\![B_m \vdash]\!]^\Pi.$$

Supposons qu'aucune des variables $x_1, \ldots, x_n$ *ni aucun des noms* $\alpha_1, \ldots, \alpha_m$*, n'est libre dans ces objets. Alors*

— *si*

$$x_1 : A_1, \ldots, x_n : A_n \vdash S : A \mid \alpha_1 : B_1, \ldots, \alpha_m : B_m$$

est dérivable alors

$$\big\langle \ldots \langle S\,[\phi_1, \ldots, \phi_n/x_1, \ldots, x_n]\,,\, e_1\rangle_{\alpha_1} \ldots,\, e_m\big\rangle_{\alpha_m} \in [\![\vdash A]\!]^\Sigma ;$$

— *si*

$$x_1 : A_1, \ldots, x_n : A_n \mid E : A \vdash \alpha_1 : B_1, \ldots, \alpha_n : B_n$$

est dérivable alors

$$\left\langle \ldots \langle E\,[\phi_1, \ldots, \phi_n / x_1, \ldots, x_n]\,, e_1 \rangle_{\alpha_1} \ldots, e_m \right\rangle_{\alpha_m} \in [\![A \vdash]\!]^{\Pi^\Sigma};$$

— *si*

$$C : (x_1 : A_1, \ldots, x_n : A_n \vdash \alpha_1 : B_1, \ldots, \alpha_n : B_n)$$

est dérivable alors

$$\left\langle \ldots \langle C\,[\phi_1, \ldots, \phi_n / x_1, \ldots, x_n]\,, e_1 \rangle_{\alpha_1} \ldots, e_m \right\rangle_{\alpha_m} \in \perp\!\!\!\perp .$$

DÉMONSTRATION: On prouve les trois items de ce résultat par mutuelle induction sur les dérivations de typage (c'est-à-dire sur les objets) :

(i) Le cas de la variable découle des hypothèses.

(ii) Le cas essentiel est celui de l'abstraction $\lambda x\, s$, qui se démontre au moyen du lemme précédent, par l'hypothèse d'induction appliquée à s.

(iii) Le cas de la μ-abstraction découle de la définition des $\mu\alpha c \in [\![\vdash A]\!]$, par l'hypothèse d'induction appliquée à c.

(iv) Le cas des piles se déduit directement des hypothèses d'induction correspondantes, vu la définition de $[\![A \vdash]\!]$.

(v) Le cas de la commande $\langle s\,, e \rangle$ s'obtient par le lemme 8.24, appliqué aux hypothèses d'induction pour s et e.

(vi) Les trois cas 0 et les trois passages à la somme sont directs.

Notons que la présence de l'hypothèse sur la liberté des variables et des noms tient au fait qu'on n'a pas défini de substitution/coupure nommée simultanée. Il est facile de contourner cette contrainte au cours de la preuve, en renommant lorsque c'est nécessaire : par exemple $S\,[x := y]\,[\alpha := \beta] \in [\![\vdash A]\!]^\Sigma$ si et seulement si S aussi (voir également la définition 8.19). □

Un corollaire direct est la normalisation forte du $\bar{\lambda}\mu$-calcul avec produit de convolution du chapitre précédent.

À suivre. . .

En guise de conclusion, nous souhaitons aborder ici quelques axes de recherche et questions non résolues qui semblent se dégager de notre travail.

$\bar{\lambda}\mu$-calcul avec ressources

Une propriété remarquable de la définition du $\bar{\lambda}\mu$-calcul différentiel est qu'il suffit :

— de supprimer la formation des arguments linéaires $[s]$ pour obtenir la variante du $\bar{\lambda}\mu$-calcul avec produit de convolution suggérée en section 7.4 ;

— de supprimer la formation de boîtes $S^!$ pour obtenir un $\bar{\lambda}\mu$-calcul avec ressources, qui s'interprète tout naturellement dans les réseaux symétriques purs, et qui en exploite toutes les règles de réduction.

On peut naturellement coder le λ-calcul avec ressources dans ce $\bar{\lambda}\mu$-calcul avec ressources en traduisant $\langle s\rangle\,(t_1 * \cdots * t_n)$ par $\mu\alpha\,\langle s\,,\,[t_1] * \cdots * [t_n] * \uparrow\alpha\rangle$: modulo l'équivalence structurelle des RDP, ceci préserve l'interprétation du λ-calcul avec ressources dans les RID. Ceci suggère d'étudier l'expansion de la promotion définie inductivement sur les objets du $\bar{\lambda}\mu$-calcul avec produit de convolution, et à valeur dans les combinaisons linéaires formelles d'objets du $\bar{\lambda}\mu$-calcul avec ressources, avec le cas clef

$$\left(S^!\right)^* = \sum_{n=0}^{\infty} \frac{1}{n!}\,[S^*]^n$$

qui reproduit dans la syntaxe la formule de Taylor–Ehrhard étudiée dans [dC07]. Il sera en particulier intéressant de voir en quel sens une telle expansion est valide par rapport à la réduction, dans la veine de [ER06b] et [ER06a].

Dualité et appel par valeur

Notre étude est entièrement conduite sous l'éclairage de la traduction de la logique classique dans LLP par la décomposition de Girard : $A \to B = !A \multimap B$. C'est la traduction négative, qui correspond à LKT, au $\lambda\mu$-calcul et au $\bar{\lambda}\mu\tilde{\mu}_T$-calcul, c'est-à-dire à des systèmes implémentant une réduction en appel par nom (voir la troisième partie de la thèse de Laurent [Lau02]). Il existe une traduction

positive des connecteurs de la logique classique dans LLP qui correspond, elle, aux systèmes *par valeur* : LKQ, le λμ-calcul en appel par valeur de [OS97] et le $\bar{\lambda}\mu\tilde{\mu}_{Q}$-calcul. Il nous manque donc la moitié du tableau.

Dans le même ordre d'idée, on peut s'interroger (la question nous a été posée par Laurent) sur le statut du produit de convolution à travers une traduction par continuation (CPS-traduction). Le candidat naturel pour être la cible d'une telle traduction est le λ-calcul augmenté des sommes de termes et du 0 : rappelons que le produit des piles se comporte comme un entrelacement de sommes d'arguments. Nous n'avons toutefois pas (encore) su apporter de réponse satisfaisante à cette question dans le cadre d'une CPS-traduction par nom. Peut-être une éventuelle interprétation du produit de convolution dans un cadre par valeur apportera-t-elle des intuitions utiles.

Sémantique catégorique

Nous avons fourni une sémantique dénotationnelle commune aux réseaux polarisés et différentiels. La définition des RDP en étant tirée, il est évident qu'ils l'admettent également comme modèle. Ceci répond vraisemblablement à un schéma plus général : tout modèle commun aux réseaux différentiels et aux réseaux polarisés est un bon candidat pour fournir une sémantique des RDP. Il serait intéressant de voir sous quelles conditions de compatibilité cette émergence a automatiquement lieu.

Plus généralement, l'établissement d'une sémantique catégorique pour les réseaux différentiels polarisés s'impose comme une étape importante d'une meilleure compréhension de ce système. Il serait facile de donner une axiomatique générale des modèles des RDP : partant d'une catégorie $*$-autonome, il suffit de traduire les règles de réduction en diagrammes qui doivent commuter. Ceci ne nous apporterait cependant aucune information nouvelle. L'objectif d'une sémantique catégorique est plutôt de synthétiser d'une manière éclairante des conditions raisonnablement minimales pour l'émergence d'un modèle.

En ce sens, il paraît raisonnable de tenter d'adapter le travail de Laurent et Regnier dans [LR03] : celui-ci caractérise l'extension par polarisation dans un modèle de Lafont de la logique linéaire, comme une construction de co-Kleisli restreinte aux $\mathfrak{P}$-monoïdes. On peut s'attendre à ce qu'une construction similaire, impliquant cette fois des bigèbres, produise des modèles des RDP sans dérivée. On pourra alors s'interroger sur le statut d'un éventuel opérateur de dérivation dans ces modèles.

Concurrence et mobilité

Nous avons souligné en introduction les liens qu'entretiennent les réseaux différentiels, et de manière moins directe le λ-calcul avec ressources, avec des notions de concurrence. Il est clair que le $\bar{\lambda}\mu$-calcul différentiel reproduit au moins les aspects concurrents du λ-calcul avec ressources. Mais sa plus grande

proximité avec la structure des réseaux nous laisse espérer des résultats plus profonds.

En particulier, le résultat d'Ehrhard et Laurent [EL07] implique que les réseaux différentiels permettent d'exprimer non seulement des aspects concurrents, mais également de la *mobilité*. En effet, les processus π-calcul partagent des ressources sous la forme de noms de canaux, dont la portée n'est pas nécessairement globale, ce qui introduit une notion de localité : la communication d'un nom peut modifier sa portée, ce qui revient à considérer que les processus peuvent changer de localité. La correction de la traduction d'Ehrhard et Laurent prend d'ailleurs la forme d'une bissimulation : on garde la trace des noms transmis par les réductions de communication (redex $\langle \partial \, , \, d \rangle$).

Il n'est pas évident qu'un tel résultat puisse être reproduit dans le cadre du $\bar{\lambda}\mu$-calcul différentiel, en particulier parce que les réseaux interprétant les processus dans [EL07] ne sont pas séquentiels. Mais l'intérêt de la démarche d'Ehrhard et Laurent ne réside pas dans l'établissement d'une $(n+1)$-ième traduction du π-calcul dans un formalisme arbitraire. C'est plutôt une illustration de l'expressivité des réseaux différentiels, lesquels fournissent un modèle du calcul concurrent intéressant par ailleurs. Leur ancrage fort dans une théorie logique et mathématique bien établie suggère qu'ils peuvent apporter des bases solides à la théorie de la concurrence.

Dans cette optique, on cherchera plutôt à découvrir et préciser les aspects concurrents et mobiles du $\bar{\lambda}\mu$-calcul différentiel, sans nécessairement vouloir traduire fidèlement le π-calcul.

Annexes

Annexe A

Notations

Abréviations

Sigles

CCC : catégorie cartésienne fermée
EER : étape élémentaire de réduction
RDP : réseau différentiel polarisé
RID : réseau d'interaction différentiel
RIDP : réseau d'interaction différentiel pur
RIDS : réseau d'interaction différentiel simple
RIP : réseau d'interaction polarisé
RIPP : réseau d'interaction polarisé pur
RME : réseau multiplicatif-exponentiel
RMEP : réseau multiplicatif-exponentiel pur
RS : réseau symétrique
RSS : réseau symétrique simple
RDP : structure différentielle polarisée
SI : structure d'interaction
SIC : structure d'interaction concrète
SID : structure d'interaction différentielle
SIDS : structure d'interaction différentielle simple
SII : SI avec interface
SIIP : SI avec interface partielle
SIP : structure d'interaction polarisée
SIPP : structure d'interaction pure
SIS : structure d'interaction simple
SISI : SIS avec interface

SISIP : SIS avec interface partielle

SIST : SIS typée

SIT : SI typée

SME : structure multiplicative-exponentielle

SMEP : structure multiplicative-exponentielle pure

SS : structure symétrique

SSS : structure symétrique simple

Systèmes logiques

LC : calcul des séquents classique déterministe de Girard [Gir91]

LJ : calcul des séquents intuitionniste

LK : calcul des séquents classique (élimination des coupures non déterministe)

LKQ, LKT : calculs des séquents classiques déterministes de Joinet et Schellinx [Joi93, Sch94]

LL : logique linéaire [Gir87]

LLP : logique linéaire polarisée [Lau02]

MELL : fragment multiplicatif-exponentiel de la logique linéaire

Notations mathématiques courantes

$\mathbb{N}$ ou ω : ensemble des entiers naturels

$\delta_{x,y}$: symbole de Kronecker : $\delta_{x,y} = 1$ si $x = y$, 0 sinon

$\mathcal{M}_{\text{fin}}(X)$: ensemble des multiensembles sur X

$[x_1, \ldots, x_n]$: multiensemble formé des éléments x_1 à x_n, en prenant en compte les multiplicités

$[]$: multiensemble vide

Structures d'interaction

Signatures

$\mathsf{a}(\alpha)$: arité du symbole α

$\bar{\mathsf{a}}(\alpha)$: coarité du symbole α

$\mathsf{d}(\alpha)$: degré du symbole α

$\vDash_\alpha$: relation de typage associée au symbole α

$\Sigma^{[!]}$: signature des SI avec boîtes sur Σ

$\Sigma^{!}$: signature des SI avec boîtes simples sur Σ

Σ_{LL} : signature des unicellules multiplicatives-exponentielles

$\Sigma^{!}_{LL}$: signature des SME et des SIP

Σ_{Diff} : signature des SID et des SS

$\Sigma^{[!]}_{Diff}$: signature des SDP

Données des structures d'interaction

α_c : symbole de la cellule c

c^i : port actif numéro i de la cellule c

$c^{[i]}$: port passif numéro i de la cellule c

$\overline{p}_\mu$: coport du port p dans SIC μ

$\mathsf{Ce}(\mu)$: ensemble des cellules de μ

$\mathsf{Fi}(\mu)$: ensemble des fils de μ

$\mathsf{Po}(\mu)$: ensemble des ports de μ

$\mathsf{PI}(\mu)$: ensemble des ports internes de μ

$\mathsf{PL}(\mu)$: ensemble des ports libres de μ

SI : ensemble des SI

SI_Σ : ensemble des SI sur la signature Σ

SII : ensemble des SII

SIS : ensemble des SIS

SIS_Σ : ensemble des SIS sur la signature Σ

Opérations

μ^I : μ muni de l'interface I

$\langle \mu^I , \nu^J \rangle$: connexion de μ et ν à travers les interfaces I et J

ε : SIS vide

$\mu \parallel \nu$: juxtaposition des SIS μ et ν

0 : SI nulle

$\overline{\sum}_{i=1}^{n}$: somme des SIS μ_1 à μ_n

$\mu^I \vdash A_1, \ldots, A_n$: relation de typage associée à la SII μ^I : $(A_1, \ldots, A_n)$ est un typage possible des ports de I dans μ

Réduction

$\langle \alpha^i , \beta^j \rangle$: paire active entre le numéro de port actif i du symbole α et le numéro de port actif j du symbole β

PA : ensemble des paires actives

$\langle c^i , d^j \rangle$: coupure entre le port i de la cellule c et le port j de la cellule d

$c^i \bowtie d^j$: réduit de la paire active (ou du redex) $\langle c^i , d^j \rangle$

$c^i \overset{\bullet}{\bowtie} d^j$: réduit, sauf 0, de $\langle c^i , d^j \rangle$

$\longrightarrow$: relation de réduction des SI

$\xrightarrow{r}$: EER pour le redex r

$\longrightarrow_{[!]}$: réduction avec boîtes

$\multimap$: réduction sauf à 0

$\longrightarrow^n$: itérée n-ième de $\longrightarrow$

$\longrightarrow^?$: clôture réflexive de $\longrightarrow$

$\longrightarrow^+$: clôture transitive de $\longrightarrow$

$\longrightarrow^*$: clôture réflexive et transitive de $\longrightarrow$

Notions liées aux signatures de la logique linéaire

$=_s$: équivalence structurelle

$\xrightarrow{comm}$: réduction de communication

$\xrightarrow{eff}$: réduction d'effacement

$\xrightarrow{route}$: réduction de routage

$\xrightarrow{simple}$: réduction simple

$\xrightarrow{struct}$: réduction structurelle

Sémantique relationnelle

Rel : catégorie des ensembles et relations

MRel : catégorie cartésienne déduite de Rel par co-Kleisli

$\mathcal{D}$: objet réflexif de MRel

$a :: \alpha$: empilement du multiensemble $a \in \mathcal{M}_{fin}(\mathcal{D})$ sur $\alpha \in \mathcal{D}$

ι : unité de $\mathcal{D}$

$\alpha \star \beta$: produit de $\alpha, \beta \in \mathcal{D}$

Calculs de termes

Données des calculs de termes

0 : objet (terme, pile ou commande) nul

1 : pile unité en $\bar{\lambda}\mu$-calcul avec produit de convolution ou $\bar{\lambda}\mu$-calcul différentiel, horde unité en λ-calcul avec ressources

$\mathcal{N}$: ensemble des noms

$NL(\Theta)$: ensemble des noms libres de l'objet Θ

$\mathcal{V}$: ensemble des variables

$\mathrm{VL}(\Theta)$: ensemble des variables libres de l'objet Θ

Λ_0 : ensemble des pré-termes

Λ : ensemble des termes simples

Λ^+ : ensemble des termes

$\mathcal{C}$: ensemble des commandes simples

$\mathcal{C}^+$: ensemble des commandes

$\mathcal{S}_0$: ensemble des piles atomiques

$\mathcal{S}$: ensemble des piles simples

$\mathcal{S}^+$: ensemble des piles

Opérations

$\Theta[x := S]$: substitution du terme S à la variable x dans l'objet Θ

$(\Theta)_\alpha S$: application nommée de l'objet Θ au terme S à travers α

$\Theta[\alpha := E]$: substitution de la pile E au nom α dans l'objet Θ

$\frac{\partial \Theta}{\partial x} \cdot S$: dérivée partielle en x de l'objet Θ dans la direction du terme S

$D_\alpha \Theta \cdot S$: dérivée nommée de l'objet Θ dans la direction de S à travers α

$\langle \Theta , E \rangle_\alpha$: coupure nommée de l'objet Θ contre la pile E à travers α

Réduction

$\to$: β-réduction

$\to^*$: fermeture réflexive et transitive de $\to$

$\rightrightarrows$: réduction en parallèle

$\Theta\downarrow$: réduit plein de l'objet Θ

$\leftarrow_\eta$: η-expansion

$\leftarrow'_\eta$: η-expansion partielle en $\bar{\lambda}\mu$-calcul avec produit de convolution

$\langle \lambda x\, s , e \rangle_0$: plus grand pas de réduction parallèle pour un redex $\langle \lambda x\, s , e \rangle$

Annexe B

Index

Annexe C

Bibliographie

[BCL99] Gérard Boudol, Pierre-Louis Curien, and Carolina Lavatelli. A semantics for lambda calculi with resources. *Mathematical Structures in Computer Science*, 9(4) :437–482, 1999.

[BCS06] R. F. Blute, J. R. B. Cockett, and R. A. G. Seely. Differential categories. *Mathematical. Structures in Comp. Sci.*, 16(6) :1049–1083, 2006.

[BEM07] Antonio Bucciarelli, Thomas Ehrhard, and Giulio Manzonetto. Not enough points is enough. In *Computer Science Logic*, volume 4646 of *Lecture Notes in Computer Science*, pages 298–312. Springer Berlin, 2007.

[Bie93] G. Bierman. *On Intuitionistic Linear Logic*. PhD thesis, University of Cambridge, Computer Laboratory, 1993.

[BL95] Gérard Boudol and Cosimo Laneve. Termination, deadlock and divergence in the lambda-calculus with multiplicities. *Electr. Notes Theor. Comput. Sci.*, 1, 1995.

[BL00] Gérard Boudol and Cosimo Laneve. λ-calculus, multiplicities and the π-calculus. In Gordon Plotkin, Colin Stirling, and Mads Tofte, editors, *Proof, Language and Interaction : Essays in Honour of Robin Milner*, pages 659–689. MIT Press, 2000.

[Bou93] Gérard Boudol. The lambda-calculus with multiplicities (abstract). In *CONCUR '93 : Proceedings of the 4th International Conference on Concurrency Theory*, pages 1–6, London, UK, 1993. Springer-Verlag.

[CG99] Roberto Di Cosmo and Stefano Guerrini. Strong normalization of proof nets modulo structural congruences. In Paliath Narendran and Michaël Rusinowitch, editors, *RTA*, volume 1631 of *Lecture Notes in Computer Science*, pages 75–89. Springer, 1999.

[CH00] Pierre-Louis Curien and Hugo Herbelin. The duality of computation. *ACM SIGPLAN Notices*, 35(9) :233–243, 2000.

[CKP03] Roberto Di Cosmo, Delia Kesner, and Emmanuel Polonovski. Proof nets and explicit substitutions. *Mathematical Structures in Computer Science*, 13(3) :409–450, 2003.

[Dan90] Vincent Danos. *Une application de la logique linéaire à l'étude des processus de normalisation (principalement de λ-calcul)*. Ph.D. Thesis, Université Denis Diderot, Paris 7, 1990.

[dC07] Daniel de Carvalho. *Sémantiques de la logique linéaire et temps de calcul*. Thèse de doctorat, Université Aix-Marseille II, 2007.

[DR95] Vincent Danos and Laurent Regnier. Proof-nets and Hilbert space. In J.-Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*, pages 307–328. Cambridge University Press, 1995.

[Ehr01] Thomas Ehrhard. On Köthe sequence spaces and linear logic. *Mathematical Structures in Computer Science*, 12 :579–623, 2001.

[Ehr05] Thomas Ehrhard. Finiteness spaces. *Mathematical. Structures in Comp. Sci.*, 15(4) :615–646, 2005.

[EL07] Thomas Ehrhard and Olivier Laurent. Interpreting a finitary pi-calculus in differential interaction nets. In Luis Caires and Vasco T. Vasconcelos, editors, *Concurrency Theory (CONCUR '07)*, volume 4703 of *Lecture Notes in Computer Science*, pages 333–348. Springer, September 2007.

[ER03] Thomas Ehrhard and Laurent Regnier. The differential lambda-calculus. *Theoretical Computer Science*, 309 :1–41, 2003.

[ER05] Thomas Ehrhard and Laurent Regnier. Differential interaction nets. *Electr. Notes Theor. Comput. Sci.*, 123 :35–74, 2005.

[ER06a] Thomas Ehrhard and Laurent Regnier. Böhm trees, krivine's machine and the taylor expansion of lambda-terms. In Arnold Beckmann, Ulrich Berger, Benedikt Löwe, and John V. Tucker, editors, *CiE*, volume 3988 of *Lecture Notes in Computer Science*, pages 186–197. Springer, 2006.

[ER06b] Thomas Ehrhard and Laurent Regnier. Uniformity and the taylor expansion of ordinary lambda-terms. Archives ouvertes `oai:hal.archives-ouvertes.fr:hal-00150275_v1`, à paraître dans Theoretical Computer Science, 2006.

[Fio07] Marcelo P. Fiore. Differential structure in models of multiplicative biadditive intuitionistic linear logic. In Rocca [Roc07], pages 163–177.

[Gir87] Jean-Yves Girard. Linear logic. *Theor. Comput. Sci.*, 50 :1–102, 1987.

[Gir91] Jean-Yves Girard. A new constructive logic : Classical logic. *Mathematical Structures in Computer Science*, 1(3) :255–296, 1991.

[Gir96] Jean-Yves Girard. Coherent banach spaces : a continuous denotational semantics. *Electr. Notes Theor. Comput. Sci.*, 3, 1996.

[GM01] Stefano Guerrini and Andrea Masini. Parsing MELL proof nets. *Theoretical Computer Science*, 254(1–2) :317–335, 2001.

[Her95] Hugo Herbelin. *Séquents qu'on calcule*. Thèse d'université, Université Paris 7, 1995.

[JdNM07] Paulin Jacobé de Naurois and Virgile Mogbil. Correctness of multiplicative (and exponential) proof structures in nl-complete. In Jacques Duparc and Thomas A. Henziger, editors, *Computer Science Logic*, volume 4646 of *Lecture Notes in Computer Science*, pages 435–450. Springer-Verlag Heidelberg, 2007. 21st International Workshop, CSL 2007, 16th Annual Conference of the EACSL, Lausanne, Switzerland, September 11-15.

[Joi93] J.-B. Joinet. *Etude de la normalisation du calcul des séquents classique à travers la logique linéaire*. PhD thesis, Université Paris VII, Janvier 1993.

[Kri90] Jean-Louis Krivine. *Lambda-calcul, types et modèles*. Masson, Paris, 1990.

[Laf88] Yves Lafont. *Logiques, catégories et machines*. PhD thesis, Université Paris 7, 1988.

[Laf95] Yves Lafont. From proof nets to interaction nets. In J.-Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*, pages 225–247. Cambridge University Press, 1995.

[Lau02] Olivier Laurent. *Etude de la polarisation en logique*. Thèse de doctorat, Université Aix-Marseille II, March 2002.

[LR03] Olivier Laurent and Laurent Regnier. About translations of classical logic into polarized linear logic. In *Proceedings of the eighteenth annual IEEE symposium on Logic In Computer Science*, pages 11–20. IEEE Computer Society Press, June 2003.

[LS88] J. Lambek and P. J. Scott. *Introduction to higher order categorical logic*. Cambridge University Press, New York, NY, USA, 1988.

[Maz06] Damiano Mazza. *Interaction Nets : Semantics and Concurrent Extensions*. Ph.D. Thesis, Université de la Méditerranée/Università degli Studi Roma Tre, 2006.

[Mel03] Paul-André Melliès. Categorical models of linear logic revisited. Technical report, 2003. Archives ouvertes `oai:hal.archives-ouvertes.fr:hal-00154229_v1`, à paraître dans Theoretical Computer Science.

[MPW92] Robin Milner, Joachim Parrow, and David Walker. A calculus of mobile processes, i. *Inf. Comput.*, 100(1) :1–40, 1992.

[OS97] C.-H. Luke Ong and Charles A. Stewart. A curry-howard foundation for functional computation with control. In *POPL*, pages 215–227, 1997.

[Par91] Michel Parigot. Free deduction : An analysis of "computations" in classical logic. In *RCLP*, pages 361–380, 1991.

[Par92] Michel Parigot. λμ-calculus : An algorithmic interpretation of classical natural deduction. In *LPAR '92 : Proceedings of the International Conference on Logic Programming and Automated Reasoning*, pages 190–201, London, UK, 1992. Springer-Verlag.

[PdF08] M. Pagani and L. Tortora de Falco. Strong normalization property for second order linear logic. To appear in Theoretical Computer Science, 2008.

[Pol04] Emmanuel Polonowski. *Substitutions explicites, logique et normalisation.* Thèse d'université, Université Paris 7, 2004.

[Reg92] Laurent Regnier. *Lambda-calcul et réseaux.* PhD thesis, Université Paris 7, 1992.

[Roc07] Simona Ronchi Della Rocca, editor. *Typed Lambda Calculi and Applications, 8th International Conference, TLCA 2007, Paris, France, June 26-28, 2007, Proceedings*, volume 4583 of *Lecture Notes in Computer Science*. Springer, 2007.

[Sch66] Laurent Schwartz. *Théorie des distributions.* Hermann, 1966.

[Sch94] H. Schellinx. *The Noble Art of Linear Decorating.* ILLC Dissertation Series, 1994-1. PhD Thesis, University of Amsterdam, 1994.

[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, 2001.

[Tak95] Masako Takahashi. Parallel reductions in lambda-calculus. *Inf. Comput.*, 118(1) :120–127, 1995.

[Vau07a] Lionel Vaux. Convolution $\bar{\lambda}\mu$-calculus. In Rocca [Roc07], pages 381–395.

[Vau07b] Lionel Vaux. The differential λμ-calculus. *Theor. Comput. Sci.*, 379(1-2) :166–209, 2007.

[Vau07c] Lionel Vaux. On linear combinations of λ-terms. In Franz Baader, editor, *RTA*, volume 4533 of *Lecture Notes in Computer Science*, pages 374–388. Springer, 2007.

Résumé : Cette thèse de théorie de la démonstration étudie les interactions entre le λ-calcul différentiel d'Ehrhard et Regnier d'un côté, et certaines émanations calculatoires de la logique classique (le λμ-calcul de Parigot et le $\bar{\lambda}\mu$-calcul de Herbelin). L'étude est initiée et guidée par la décomposition de ces calculs dans des extensions de la logique linéaire de Girard. Dans une première partie, on définit un cadre commun pour ces extensions, dans le formalisme des réseaux d'interaction de Lafont, et on y rappelle des résultats de la littérature ou du folklore. On donne en particulier la traduction du λμ-calcul et du $\bar{\lambda}\mu$-calcul dans les réseaux polarisés de Laurent et celle du fragment finitaire du λ-calcul différentiel dans les réseaux différentiels d'Ehrhard et Regnier. Dans la deuxième partie, on introduit les réseaux différentiels polarisés (RDP), comme l'extension par une polarisation à la Laurent des réseaux différentiels. La pertinence des règles de réduction nouvelles est soulignée par l'étude d'un modèle dénotationnel commun aux réseaux différentiels et aux réseaux polarisés. Enfin, on présente trois calculs de termes, chacun pouvant être considéré comme une lecture en arrière de tout ou partie des interactions définies par les RDP : un λμ-calcul différentiel, qui correspond à la réunion des réseaux différentiels et des réseaux polarisés ; un $\bar{\lambda}\mu$-calcul avec produit de convolution sur les piles, qui fait intervenir la structure de bigèbre des types polarisés introduite dans les RDP, mais pas la dérivée ; enfin, un $\bar{\lambda}\mu$-calcul différentiel qui développe toute l'expressivité des RDP.

Mots-clés : logique linéaire — logique linéaire différentielle — logique classique — logique linéaire polarisée — λ-calcul différentiel — λμ-calcul — $\bar{\lambda}\mu$-calcul — correspondance de Curry–Howard

---

Title : Differential λ-calculus and classical logic : their computational interactions

Abstract : We study the possible interactions between Ehrhard–Regnier's differential λ-calculus and pure calculi associated with classical logic : Parigot's λμ-calculus and Herbelin's $\bar{\lambda}\mu$-calculus. The impetus to this study is given by the respective decompositions of these calculi in particular extensions of Girard's linear logic. We first provide a unified framework for these extensions, as a variant of Lafont's interaction nets. We recall well known definitions and results, either from litterature or from folklore, rephrased in this setting : in particular, we explicit translations of λμ-calculus and $\bar{\lambda}\mu$-calculus into Laurent's polarized proof nets and a translation of the finitary fragment of differential λ-calculus into Ehrhard–Regnier's differential interaction nets. In a second part, we introduce polarized differential nets (PDN) : these are obtained as an extension of differential nets by a polarization scheme à la Laurent. The new reduction rules are justified by the properties of a denotational model of both polarized nets and differential nets. Last we proceed to the introduction of three pure term calculi, each corresponding to a readback from the dynamics of PDN, through a Curry–Howard–Girard decomposition : a differential λμ-calculus, which accounts for the union of polarized nets and interaction nets ; an extension of $\bar{\lambda}\mu$-calculus involving a convolution product on stacks, which provides a computational meaning to the structure of bialgebra induced on polarized types by the reduction of PDN ; last, a differential $\bar{\lambda}\mu$-calculus encompassing all of the dynamics of PDN.

Keywords : linear logic — differential linear logic — classical logic — polarized linear logic — differential λ-calculus — λμ-calculus — $\bar{\lambda}\mu$-calculus — Curry–Howard correspondance

---

Discipline : Mathématiques

Laboratoire : Institut de Mathématiques de Luminy (UMR 6206) — Campus de Luminy, Case 907 — 13288 MARSEILLE Cedex 9