École doctorale n° 386 : Sciences mathématiques de Paris Centre

Thèse de doctorat

Mathématiques : Logique et fondements de l'Informatique

Sous la direction de Christine Tasson et Lionel Vaux Auclair

présentée et soutenue publiquement par

Jules Chouquet

le 6 décembre 2019

Une géométrie du calcul

Réseaux de preuve, Appel-Par-Pousse-Valeur et Topologie du consensus

Jury

Mme. Carole Delporte,	Professeur, Université de Paris	Examinatrice
M. Thomas Ehrhard,	Directeur de recherches CNRS, Université de Paris	Examinateur
M. Damiano Mazza,	Chargé de recherches CNRS, Univirsité Paris 13	Rapporteur
M. Samuel Mimram,	Professeur, École Polytechnique	Rapporteur
M. Laurent Regnier,	Professeur, Aix-Marseille Université	Examinateur
Mme Christine Tasson,	Maître de conférences, Université de Paris	Co-directrice de thèse
M. Lorenzo Tortora de Falco,	Professeur, Università Roma Tre	Examinateur
M. Lionel Vaux Auclair,	Maître de conférences, Aix-Marseille Université	Co-directeur de thèse

Irif - Universivé de Paris
UMR CNRS 8243, Bâtiment Sophie Germain
8 place Aurélie Nemours, 75205 Paris Cedex 13

À Magali

Résumé

Cette thèse propose une étude quantitative de plusieurs modèles de calcul de l'informatique fondamentale et de la théorie de la démonstration. Deux approches sont menées : la première consiste à examiner les mécanismes d'approximation multilinéaires dans des systèmes issus du λ-calcul et de la Logique Linéaire. La seconde consiste à étudier les modèles topologiques pour les systèmes distribués et à les adapter aux algorithmes probabilistes. On étudie d'abord le développement de Taylor des réseaux de preuve de la Logique Linéaire. On introduit des méthodes de démonstration qui utilisent la géométrie de l'élimination des coupures des réseaux multiplicatifs, et qui permettent de manipuler des sommes infinies de réseaux de façon sûre et correcte, pour en extraire des propriétés sur les réductions qui sont à l'œuvre. Ensuite, nous introduisons un langage permettant de définir le développement de Taylor syntaxique pour l'Appel-Par-Pousse-Valeur (Call-By-Push-Value), en capturant certaines propriétés de la sémantique dénotationelle liées aux morphismes de coalgèbres. Puis nous nous intéressons aux systèmes distribués (à mémoire partagée, tolérants aux pannes), et au problème du Consensus. On utilise un modèle topologique qui permet d'interpréter la communication dans les complexes simpliciaux, et on l'adapte de façon à transformer les résultats d'impossibilité bien connus en résultats de borne inférieure de probabilité pour des algorithmes probabilistes.

Mots-clefs Logique Linéaire, Développement de Taylor, Lambda calcul, Call-By-Push-Value, Réseaux de preuve, Théorie de la démonstration, Systèmes distribués, Topologie, Consensus, Probabilités

Abstract

This Phd thesis presents a quantitative study of various computation models of fundamental computer science and proof theory, in two principad directions : the first consists in the examination of mecanismis of multilinear approximations in systems related to λ-calculus and Linear Logic. The second consists in a study of topological models for asynchronous distributed systems, and probabilistic algorithms. We first study Taylor expansion in Linear Logic proof nets. We introduce proof methods using the geometry of cut elimination in multiplicative nets, and which allow to work with infinite sums of nets in a safe and correct way, in order to extract properties about reduction. Then, we introduce a language allowing us to define Taylor expansion for Call-By-Push-Value, while capturing some properties of the denotational semantics, related to coalgebras morphisms. We focus then on fault tolerant-distributed systems with shared memory, and to Consensus problem. We use a topological model which allows to interpret communication with simplicial complexes, and we adapt in so as to transform the well-known impossibility results in lower bounds for probabilistic algorithms.

Keywords Linear Logic, Taylor Expansion, Lambda Calculus, Call-By-Push-Value, Proof nets, Proof theory, distributed systems, topology, Consensus, Probabilities

Remerciements

Je remercie :

D'abord et avant tout ma directrice et mon directeur de thèse : Lionel, pour avoir accepté de recevoir le stagiaire philosophe [1] que j'étais, pour me former à la pratique de la recherche scientifique avec beaucoup de pédagogie et, oserai-je le dire, de patience. Puis de m'avoir fait confiance pour renouveler cette collaboration dans un projet de plus grande envergure, et ainsi avoir pu continuer à me faire découvrir tout un tas de choses très intéressantes [2]. Christine, pour avoir pris le relais dans mon encadrement, m'avoir montré que j'étais capable de plus que je n'aurais cru à force d'exigence ; pour avoir été au moins aussi bienveillante qu'exigeante ; et bien sûr pour notre programme d'échange culturel autour des produits régionaux du terroir français. Tous les deux pour avoir toujours fait preuve de compréhension et de soutien quand j'ai pu être ralenti dans mon travail par ma santé. Sachez également que vous avez fait des jaloux parmi mes collègues doctorants, par la disponibilité et l'investissement dont vous avez tous deux fait preuve au cours de cette direction.

Alexis Saurin pour m'avoir proposé de faire cette thèse qui convenait en tous points à ce que je pouvais espérer faire. Je remercie également l'Agence Nationale de la Recherche pour avoir intégralement financé cette thèse à travers le projet RAPIDO piloté par Alexis.

Damiano Mazza et Samuel Mimram pour avoir accepté d'être les rapporteurs de cette thèse, et pour le long travail qu'ils ont fourni à cet effet.

Je suis également très heureux et très fier que Laurent Regnier et Thomas Ehrhard aient accepté d'être examinateurs. Une grande partie des recherches que j'ai menées ces dernières années prennent source dans leurs travaux.

De même, Lorenzo Tortora de Falco pour prendre part à ce jury, et d'avoir pu faire le déplacement (à Paris en décembre, soulignons-le), ainsi que Carole Delporte pour avoir également accepté d'examiner cette thèse, ainsi que pour avoir été d'une précieuse aide dans la délicate organisation de mes enseignements.

Ma famille, qui a suivi et encouragé mes études. Tout particulièrement mes parents, qui m'ont accompagné et soutenu. Lisa, Calixte et Corto, Cécile, Louna et Loréna, Thibault, Jacky et Dominique, ma grand mère, mon oncle Joël, mon autre oncle Joël, Annette,... Bien sûr Olivier, Marie-Lorraine, Nicolas, merci de votre soutien, de votre aide, de votre hospitalité, et du reste [3].

1. Je fais référence à ma formation universitaire davantage qu'à ma personnalité.
2. Dont l'échine de porc à la sauce chorizo-moules de *Lacaille*, les claviers BÉPO, Igorrr, et accessoirement pas mal de trucs de logique et de maths.
3. Dont l'organisation du pot de thèse !

Tous les enseignants-chercheurs qui m'ont aidé et soutenu lors de mes premières années d'enseignement dans un milieu que je connaissais mal, et des sujets (je peux le dire maintenant) que je ne maîtrisais pas. Parmi eux, Giovanni Bernardi, Dominique Poulalhon, Mihaela Sighireanu, Anne Micheli, Valia Mitsou, et tous ceux que j'oublie.

Les membres du GdrI Logique Linéaire, avec qui mes intéractions furent toujours enrichissantes, et où j'ai toujours pris plaisir à échanger. Je pense à Michele Pagani, Claudia Faggian, Giulio Manzonetto, Giulio Guerrieri, Thomas Seiller, Marie Kerjean, Stefano Guerrini, Luc Pellissier, et tous ceux que j'oublie.

Ceux avec qui j'ai eu l'occasion d'avoir des conversations enrichissantes pour alimenter mes travaux de recherche, à l'IRIF et ailleurs. Je pense à Paul-André Melliès [4] , Pierre Fraigniaud, Ami Paz, Maurice Herlihy, Petr Kuznetsov.

Ceux avec qui j'aurais aimé travailler davantage, mais où le temps ou les occasions ont manqué : en particulier Alexis, Thomas Ehrhard, Lorenzo.

Ceux qui, à l'IRIF, ont été d'une aide précieuse et d'une patience admirable pour les démarches administratives que j'ai du entreprendre au laboratoire, merci à Dieneba, Étienne, Natalia, Éva.

Tous les doctorants de l'IRIF qui ont contribué à rendre sympathique mon séjour dans ce laboratoire. Théo, Nicolas, qui m'ont souvent aidé ces dernières années quand je devais enseigner des choses qui m'étaient encore très mystérieuses... Cédric, Victor, Léo, Chaitanya, Zeinab, Antoine, Rémi, Farzad, Sidi, Félix, et tous ceux que j'oublie.

Gianluca, trop fugace cobureau et copain ; et bien sûr Léonard, qui a été un camarade particulièrement agréable, fier amateur d'énigmes, de jazz manouche et de procrastination, et dont les goûts en matière de décoration sont assez proches des miens pour qu'ils hantent le bureau 3055 pour un moment... Daniel [5], infatigable poète, indispensable vecteur d'humanité et de convivialité.

Naziha pour sa bonne humeur et nos conversations du petit matin.

Les membres de l'équipe LdP de l'I2M, à Marseille, où j'ai passé la première année de ma thèse. Dimitri, Manu, Alexei, Federico, Yves, Laurent [6] Myriam, Jessica, Corinne, les ferrovipathes de la salle de pause, et tous ceux que j'oublie.

ChoCoLa, ses sympathiques organisateurs et participants.

Tous mes copains. Théo [7], Alice et Nicolas [8], Christian [9], Paul et Barbara [10], Dimitri [11], Maya, Rémi, Gautier, Laurent, Andrea [12], Rasa, Davide, merci pour votre amitié indéfectible et chère. Sarah, François, Cléo, Jean-Christophe, Anna, Tatiana, Robin, Milo, Éléonore, Justine, Clotilde, Dounia.

Tous les enseignants qui m'ont doucement amené vers cet aboutissement d'études. À Sisteron : J-M. Trouillet, C. Hilliou, S. Carrillo, J. Pouzol. À Mar-

4. Qui m'a appris qu'une qualité essentielle du chercheur réside dans le sang froid devant la difficulté, et même la perversité de constater que « ça déconne complètement » avec une certaine satisfaction.

5. Qui m'a fait beaucoup fait cogiter sur le titre de cette thèse, en me suggérant *Harry Potter et la thèse de Jules*, qui n'était peut-être après tout pas si mal.

6. Pour ton suivi assidu et attentif de mes lectures depuis des années, et tes sages recommandations autour de Vladimir Nabokov et Roger Martin du Gard.

7. Pour ton amitié et ta disponibilité (sauf pendant les soldes du *Vieux Campeur* et la saison skiable).

8. Pour votre amitié (qui survit aux soldes du *Vieux Campeur* et à la saison skiable).

9. En attendant de pouvoir servir l'*Épique tête de veau* à l'académie du Coq en pâte.

10. Pour être restés des champions.

11. Pour être mon plus vieil ami, parce que l'on aura toujours trop de choses à se dire.

12. Pour le cas où je n'imprimerais qu'une version de ma thèse.

seille : F. Schwencke, C. Bourne-Chastel, L. Salina, F. Ortiz, O.Solinas. À Paris : V. Israël-Jost, P. Wagner, S. Roux, P. Ravel, A. Tanase, F-E. Rollet, A. Tonneau, A. Naibo. Bien sûr Jean Fichot et Susana Berestovoy. Et tous ceux que j'oublie.

Sophie Partouche, pour aiguiser chez moi le goût de la musique.

L'équipe du collège Gay-Lussac de Colombes. En particulier Damien, Mickaël, Audrey, Sabina, Aurélie, Mahechi...

Tout le personnel des médiathèques de Colombes, en particulier Fred de la médiathèque Jacques Prévert.

Je tiens aussi à remercier ceux qui ont su m'accompagner d'une autre façon, parfois à leur insu, en rendant plus chaleureuses et conviviales mes errances citadines : Noëlle du *Primerose* (Sisteron), Betty de *chez Betty* (St Vincent sur Jabron), Francis et Catherine de *la Goutette*, Pierrot « claque-billets » du *Petit Nice*, Antonia de *Lacaille*, les équipes du *Marengo*, du *Little Pub*, de la *Maison Hantée*, de la *Poz* (Marseille), Rabba du *Latin St Germain*, Thierry du *Piment Café*, Clyde du *Reflet*, les équipes de l'*Art Brut*, de *Galerie 88* et du *Berthom*, Julie, son frère, et Sony de *My Tho* (Paris), mais aussi les équipes du *Local Bear* et de l'*Ethik* (Colombes).

Magali. Difficile d'écrire tout ce pour quoi j'ai à te remercier. Il faudrait une autre thèse, mais celle-là nous continuerons à l'écrire ensemble. En attendant, je te dédie celle-ci, et tout ce qui peut être dédié.

Cette thèse est aussi dédicacée à : Frédéric Courant, Jamy Gourmaud et Sabine Quindou (et Marcel), pour avoir éveillé et entretenu chez moi le plaisir de comprendre. Jean Sibelius, Philippe Etchebest, mes étudiants sympathiques, Christian Binet (à qui j'emprunte sans son accord ce format de dédicace), mes collègues sympathiques, Lucien Jacques, les tièles sétoises, Panaït Istrati, les recrutements permanents dans la recherche académique, François Rollin, Garth Ennis, Eric Powell, Antonín Dvořák, les endives (crues).

Cette thèse n'est pas dédicacée à : mes étudiants antipathiques, l'autotune, la précarité de l'emploi dans la recherche académique, mes collègues antipathiques (et ceux qui laissent leur vaisselle sale dans l'évier), Kev Adams, la faim dans le monde, Kim Kardashian, le réchauffement climatique, Michel Sardou, les côtes de blette, Booba, les barbiers qui rasent tous — et exclusivement — les gens qui ne se rasent pas eux-même, les faux ongles, les endives (cuites).

D'une façon générale, cette thèse est dédicacée aux curieux de tous genres, scientifiques ou non, avec qui mes rapports furent aussi divers qu'enrichissants.

Il y a trois stades de connaissance : ce qu'on croit, ce qu'on sait et ce qu'on peut prouver.
Tu crois que tu tiens tes pouvoirs d'une araignée radioactive et tu penses que tu sais que c'est vrai, mais peux-tu le prouver ?

Ezekiel Sims [a]

---

a. Dans *Spider Man : Le secret de Peter Parker* , J. M. Straczynski, Panini comics, 2001

Avant-propos

Ce mémoire de thèse est organisé en quatre chapitres indépendants. Le premier chapitre, introductif, a vocation à être général, et à faciliter la compréhension globale des domaines abordés. Pour autant, les trois chapitres suivants peuvent chacun être abordés séparément. De plus, chacun d'entre eux commence par une introduction propre visant à faciliter sa compréhension et sa lecture s'il est pris isolément.

L'introduction ne contient pas de matériel formellement indispensable à la lecture des chapitres suivants, à l'exception de la section 1.4 qui décrit les conventions de notations utilisées qui sont communes à l'ensemble du mémoire.

Le deuxième chapitre a fait l'objet d'une publication en collaboration avec Lionel Vaux Auclair [CV18], mais présente néanmoins des développements et des résultats supplémentaires.

Le troisième chapitre est issu d'une collaboration avec Christine Tasson, et a fait l'objet de deux publications (dont l'une avec cette dernière, à paraître) [Cho19, CT20].

Le quatrième chapitre est également l'objet de recherches avec Christine Tasson.

Les notations sont légion, le nombre de règles de réductions distinctes, par exemple, est très important. Pour cette raison, nous proposons en fin d'ouvrage une nomenclature à laquelle on peut se rapporter pour retrouver la définition d'une notation particulière et accompagner le parcours de passages techniques.

Table des matières

Chapitre 1

Introduction

C'est pas parce que c'est compliqué que c'est meilleur.

Philippe Etchebest [a]

a. *Cauchemar en cuisine – comment l'éviter*, M6 éditions, 2016

Cette thèse contient un nombre important de calculs et de dessins, ce qui, à défaut de mieux, lui donne un titre. Cette introduction a vocation à donner une idée de ce dont il s'agit un peu plus précisément. On commence par décrire quelques idées fondamentales en des termes les plus généraux et les moins techniques possibles, qui permettent de comprendre comment apparaissent les questions auxquelles on essaie de répondre. On donne ensuite un panorama chronologique des domaines de nos investigations, pour situer historiquement le contexte scientifique de nos études. Puis, avant de rentrer dans le vif du sujet, nous décrivons plus précisément le contenu du mémoire et le détail de nos contributions.

1.1 Remarques générales

1.1.1 Logique : formules et démonstrations

Un point de départ essentiel de la logique est l'étude du raisonnement et de l'argumentation. Pour pouvoir étudier les mécanismes du raisonnement en général, sans se contenter d'exemples, de cas particuliers, il nous faut nous *abstraire* du langage de tous les jours. La première étape de cette démarche consiste à utiliser un autre langage pour construire cette abstraction. Ainsi, on choisit dans

un premier temps des noms de *variables*, par exemple parmi les lettres latines majuscules $A, B, \ldots$ ou les lettres grecques minuscules $\varphi, \psi, \theta, \ldots$ La deuxième étape consiste à choisir un ensemble de symboles que l'on appellera *connecteurs*, et qui permettront d'assembler les variables entre elles. Les connecteurs logiques de base sont bien connus, il s'agit des termes « non », « et », « ou », « implique ».

À l'aide de ce langage abstrait, il devient possible de parler de *formules*, qui sont des suites de variables et de connecteurs, auxquelles on peut donner un sens. Pour illustrer nos propos dans ce chapitre introductif, nous allons considérer un langage simple constitué de variables $A, B, C, \ldots$ et d'un seul symbole : « implique », qui s'écrira « $\to$ » (et qui peut aussi se lire « si ... alors... »). Les formules peuvent alors prendre par exemple la forme : $A \to B$, ou $(A \to B) \to C$, ou encore $A \to (B \to C)$. Le parenthésage peut changer le sens d'une formule, comme en mathématiques élémentaires.

Disposer ainsi d'une syntaxe qui soit abstraite du langage naturel permet de montrer des propriétés qui sont assez générales pour s'appliquer à une infinité de cas. Par exemple, certaines formules, que l'on appelle *tautologies*, sont toujours vraies, comme la formule $A \to A$. Il est alors inutile de vérifier que la phrase *Si le cheval d'Alexandre s'appelle Bucéphale, alors le cheval d'Alexandre s'appelle Bucéphale* est vraie. Il suffit de s'assurer qu'elle est une occurrence de la tautologie $A \to A$. La généralité du langage permet donc de montrer qu'une infinité d'énoncés est correcte, en travaillant sur une seule formule.

La dernière étape, toujours pour étudier les structures des raisonnements et de l'argumentation, n'est pas la moindre : il reste à savoir que faire de ces formules. Comment montrer qu'une formule est vraie, ou qu'une formule est conséquence d'une autre ? On pourrait dire que la logique consiste essentiellement à répondre à cette question.

Présentons deux approches distinctes, quoique complémentaires, qui s'occupent de ce genre de problèmes, et qui constituent deux domaines d'étude à part entière.

La première est la *théorie des modèles*. Elle cherche à donner un sens, une interprétation aux formules, pour parler de vérité. En effet, il est impossible (notamment depuis les travaux du logicien polonais Alfred Tarski dans les années 1930) de parler de la vérité d'une formule si ce n'est pas par rapport à une interprétation (que l'on appelle aussi *modèle*). L'idée à l'œuvre en théorie des modèles est de donner, donc, une interprétation pour les formules, et de déterminer en fonction de cette interprétation si la formule est vraie ou non. Par exemple, dans un modèle où la variable A est interprétée par l'énoncé *Bucéphale est un cheval*, et la variable B est interprétée par l'énoncé *Bucéphale est un équidé*, la formule $A \to B$ est vraie. Si l'on interprétait A de la même façon, mais B par l'énoncé *Bucéphale a une robe noire*, alors la formule $A \to B$ serait fausse (sauf dans un monde où tous les chevaux auraient une robe noire, naturellement).

La deuxième approche est la *théorie de la démonstration*, c'est la discipline dans laquelle se situe une partie importante de ce mémoire de thèse. Il n'est plus question de donner un sens aux formules, mais de donner des outils permettant d'en construire des démonstrations. Comme il existe plusieurs façons de construire une interprétation des formules, il existe plusieurs façons de construire un système de démonstration. On ne parlera pas alors de formule *vraie* dans un

modèle, mais de formule *démontrable* dans un système de preuve [1].

Les systèmes de preuve sont donc constitués d'un ensemble de règles. Une règle, le plus souvent, est présentée de la manière suivante :

$$\frac{\text{Hypothèse 1} \quad \dots \quad \text{Hypothèse n}}{\text{Conclusion}} \text{ Nom de la règle utilisée}$$

La barre horizontale représente donc la déduction d'une formule à partir d'autres formules (le plus souvent : une, deux ou trois). L'exemple le plus canonique de règle de déduction (on parle aussi de règle de *dérivation*) est la suivante, que l'on appelle encore pour des raisons historiques *Modus ponens* :

$$\frac{A \to B \qquad A}{B} \textit{ Modus ponens}$$

Cette règle exprime le fait que si l'on dispose d'une démonstration de $A \to B$, et d'une démonstration de A, alors on peut considérer que l'on a une démonstration de B. Dans l'exemple ci-dessus, les formules $A \to B$ et A peuvent elles-même être issus d'autres démonstration de la sorte, il n'y a pas de limite à la taille que peuvent prendre les preuves. On parle souvent d'*arbre de preuve*, ou d'arbre de dérivation pour désigner la structure de ces objets ; les feuilles étant les hypothèses les plus hautes, et la racine étant la conclusion. Les règles de déduction constituent des nœuds entre les différentes branches.

Les formules qui constituent les feuilles des arbres de dérivation ne sont en fait pas toutes des hypothèses. Certaines ont un statut particulier, et sont considérées comme systématiquement démontrables, sans hypothèses. Ce sont les briques de base des preuves formelles : les *axiomes*.

La forme des axiomes varie selon les systèmes de preuve. L'une des plus répandues est la suivante :

$$\frac{}{A \to A} \text{ Axiome}$$

C'est-à-dire que l'on considère que pour toute formule A, la formule $A \to A$ (dont on a dit qu'elle était une tautologie) n'a pas besoin d'hypothèses pour être démontrée.

Nous allons maintenant décrire un aspect central de la théorie de la démonstration, qui sera à l'étude dans la suite de nos travaux.

1.1.2 Dynamique des preuves

Un aspect important des démonstrations formelles est que la même formule peut être prouvée de plusieurs façons différentes. C'est-à-dire que l'on a plusieurs arbres de dérivation qui aboutissent à la même conclusion en partant des mêmes hypothèses. Ces derniers peuvent en revanche utiliser les axiomes de façons distinctes, car on rappelle que les axiomes ne sont pas des hypothèses à part entière, mais des règles de déduction qui ne nécessitent pas d'hypothèse.

Considérons par exemple les deux preuves suivantes, qui démontrent la formule B sous hypothèses $A \to B$ et A (en abrégeant *Modus ponens* par *Mod. pon.* et Axiome par Ax) :

1. On emploiera indifféremment les termes *preuve* et *démonstration* tout au long de ce mémoire

$$\dfrac{\dfrac{}{B \to B}\ \text{Ax} \qquad \dfrac{A \to B \qquad A}{B}\ \textit{Mod. pon.}}{B}\ \textit{Mod. Pon.} \qquad\qquad \dfrac{A \to B \qquad A}{B}\ \textit{Mod. pon.}$$

On constate en effet que les deux démonstrations ont les mêmes formules non démontrées à leurs feuilles, ce sont les hypothèses, et qu'elles diffèrent en ce que celle de gauche utilise un axiome et une règle *Modus Ponens* supplémentaire.

On observe également que la preuve de gauche est inutilement compliquée. En effet, utiliser l'axiome et le *Modus ponens* est ici superflu, car la formule que l'on souhaitait démontrer, B, était déjà démontrée plus haut. Ce morceau de preuve redondant, qui fait faire un « détour » à la démonstration, est un exemple très simple de ce que l'on appelle une *coupure*.

Le résultat qui pose les fondations de la démonstration est un théorème dû à Gerhard Gentzen [Gen55], qui établit que toute démonstration peut être réécrite en supprimant ces détours. Il s'agit du *Théorème d'élimination des coupures* sur lequel nous aurons l'occasion de revenir à de multiples reprises.

L'idée essentielle qui est à retenir ici est que les preuves formelles sont considérées comme des objets mathématiques, et que l'élimination d'une coupure (la suppression d'un de ces détours) donne une opération qui permet de passer d'une démonstration à l'autre. Rappelons qu'il s'agit toujours de preuves d'une même formule sous les mêmes hypothèses, c'est la structure interne de la démonstration qui est modifiée lors de cette opération.

1.1.3 Lambda-calcul

Un autre langage formel qui nous occupera plus loin est le λ-calcul. On peut le présenter sous deux aspects différents. Du point de vue des mathématiques, il s'agit d'une théorie générale des fonctions. Du point de vue informatique, il s'agit d'un ancêtre des langages de programmation.

Le λ-calcul est un langage de termes, que l'on peut voir comme des fonctions : le terme qui s'écrit $\lambda x M$ correspond à la fonction qui à n'importe quel objet x associe l'objet M. Par exemple, on peut écrire $\lambda x(x^2)$, qui représente la fonction « carré ». Et le terme qui s'écrit MN correspond à une fonction associée à M, qui s'applique à un argument représenté par N. Par exemple, le terme $(\lambda x(x^2))(5)$ représente la fonction « carré » appliquée au nombre 5.

On a donc une syntaxe de termes. Cette syntaxe est accompagnée d'une règle de calcul qui permet de passer d'un terme à un autre. Cette règle s'appelle la beta-réduction, et s'écrit $\to_\beta$. Elle correspond à l'application d'une fonction à son argument. Dans l'exemple donné ci-dessus, on peut écrire

$$(\lambda x(x^2))(5) \to_\beta 5^2 = 25$$

L'argument 5 a pris la place de la variable x, et le calcul a alors pu donner un résultat.

Sans rentrer encore dans les détails, retenons pour l'instant que l'on dispose d'un langage, et que les objets écrits dans ce langage peuvent être transformés à l'aide d'une opération élémentaire, la beta-réduction. On a vu avec l'exemple plus haut que le calcul de fonctions mathématiques pouvait s'exprimer dans le λ-calcul. Du point de vue informatique maintenant, on peut considérer les termes comme des programmes, et la β-réduction comme l'exécution de ces programmes.

Dans la section précédente, nous avions également présenté une syntaxe avec des objets (les preuves formelles) auxquels on pouvait appliquer une opération (l'élimination des coupures) pour obtenir d'autres du même genre. Il existe une correspondance entre le λ-calcul et les systèmes de preuve, qui associe les preuves aux λ-termes, et l'élimination des coupures à la beta-reduction. Nous aurons l'occasion de revenir sur cette correspondance, appelée *correspondance de Curry-Howard*. Contentons-nous pour l'instant de souligner que la dynamique des preuves et les opérations du λ-calcul sont intrinsèquement liées, et que cette correspondance amène un entrelacement entre la théorie de la démonstration et l'informatique fondamentale qui sera au cœur de nos réflexions et de nos travaux.

1.1.4 Logique Linéaire

La correspondance de Curry-Howard, nous l'avons vu, dresse une analogie entre la transformation des preuves et l'exécution des programmes. Un aspect important de ladite exécution est la façon dont sont utilisés les arguments d'un terme donné. Par exemple, le terme vu plus haut $\lambda x(x^2)$ n'utilise qu'une fois son argument, il prend un nombre et le met au carré. En revanche, un autre terme : $\lambda x(x + \frac{1}{x} \times 2x - \frac{x}{2})$, produisant un calcul plus complexe, peut avoir besoin de *dupliquer* son argument. En effet, si on l'applique comme plus haut au nombre 5, la beta-réduction prendra la forme suivante :

$$(\lambda x(x + \frac{1}{x} \times 2x - \frac{x}{2}))5 \rightarrow_\beta 5 + \frac{1}{5} \times 2 \times 5 - \frac{5}{2}$$

L'argument a du être dupliqué pour que le calcul se poursuive.

De manière opposée, un terme peut *effacer* son argument. Si l'on prend le terme qui à tout argument associe une constante, par exemple $\lambda x42$, alors pour tout terme M mis en argument, la beta réduction donnera : $(\lambda x42)M \rightarrow_\beta 42$. L'argument n'est pas utilisé, on dit qu'il est effacé par la réduction.

Le principe de la Logique Linéaire consiste à prendre en compte ces notions de duplication et d'effacement du λ-calcul. Or, nous l'avons dit, le λ-calcul peut s'associer à la logique et à la théorie de la démonstration. L'idée générale de la Logique Linéaire est justement de construire un langage logique et une théorie de la démonstration qui permette de donner l'équivalent logique des mécanismes de duplication et d'effacement du λ-calcul.

De nouveaux connecteurs sont introduits. En particulier, la *flèche linéaire* $\multimap$ remplace la flèche habituelle $\rightarrow$, mais est plus précise : la dynamique des preuves associée aux formules $A \multimap B$ ne permet pas en général la duplication ni l'effacement des arguments. Pour qu'une ressource soit duplicable ou effaçable, un autre connecteur est introduit : l'exponentielle « ! ». Dans l'exécution du calcul, un argument sera duplicable ou effaçable s'il correpond du point de vue logique à une formule de la forme $!A$.

Un autre aspect de la Logique Linéaire sera longuement étudié dans ce mémoire de thèse. La théorie de la démonstration adaptée à cette nouvelle logique permet d'écrire les preuves sous formes graphiques. Cette syntaxe s'appelle *réseaux de preuve* et sera au cœur du chapitre 2, en introduction duquel on en trouvera une présentation générale et des exemples.

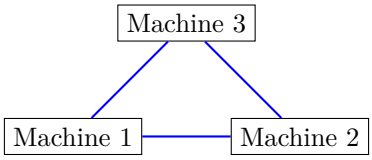
1.1.5 Algorithmes distribués

Les liens entre logique et informatique permettent d'étudier des propriétés de l'exécution de programmes à travers la théorie de la démonstration. Mais, au delà du λ-calcul, il existe d'autres modèles informatiques plus précis, et très différents, pour lesquels il est possible de fournir un modèle abstrait.

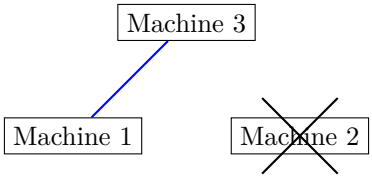
Dans le dernier chapitre de ce mémoire, nous nous intéresserons aux systèmes distribués. Dans ces systèmes, plusieurs ordinateurs, ou plusieurs processeurs font des calculs en même temps, et communiquent entre eux pour s'échanger les résultats de leurs calculs. Ce modèle de calcul est intéressant à plusieurs égards. En particulier, c'est un modèle concret, et à l'œuvre dans un large champ de développement de l'informatique contemporaine (multiprocesseurs, blockchain).

Ce qui nous intéressera plus particulièrement dans ce modèle est qu'il peut être étudié à travers des outils mathématiques généraux. De la même façon que l'on aura étudié les mécanismes de consommation de ressources en programmation à l'aide de la théorie de la démonstration (Logique Linéaire), nous étudierons ici les mécanismes de communications entre processeurs à l'aide d'outils provenant non plus de la logique, mais de la topologie.

L'idée de cette approche est de représenter les différents états du système distribué par des objets très simples. Par exemple, un système avec trois machines communiquant entre elles sera représenté de la façon suivante :



L'une des propriétés primordiales de cette représentation est qu'elle continue à fonctionner si l'une des machines tombe en panne (ce qui arrive en pratique). Si la machine 2 tombe en panne par exemple, on peut représenter la situation comme suit :



C'est-à-dire que le lien qui existait déjà entre les machines 1 et 3 continue à représenter une communication valide, bien que l'une des machines ne puisse plus envoyer ni recevoir d'informations.

Ces objets graphiques qui représentent la communication entre différentes entités sont des *complexes simpliciaux*. Leur étude permet de démontrer que certains algorithmes peuvent, ou non, être exécutés par des systèmes distribués.

1.2 Remarques historiques orientées

Nous proposons dans cette section un survol historique de l'apparition des concepts et notions qui sont rattachés à nos objets d'étude. Le développement du mémoire de thèse consistant en des contributions techniques et précises, nous

avons à cœur de remettre en contexte le cadre scientifique dans lequel se placent ces contributions. Par souci d'introduire le lecteur au sujet de nos recherches, nous donnons ce panorama chronologique pour situer l'espace de nos travaux dans le plus large espace de la logique et de l'informatique théorique à travers leurs histoires.

Le lecteur souhaitant en savoir davantage pourra se rapporter à tout bon ouvrage sur l'histoire de la logique, comme il en existe quelques uns.

1.2.1 La logique

1.2.1.1 L'antiquité grecque

Sans rentrer dans un examen archéologique des origines de la logique qui n'aurait pas sa place ici, signalons toutefois que l'on situe ses premières manifestations chez les penseurs de l'Antiquité grecque (en occident, du moins. Les corpus de philosophie chinoise et indienne, ou de l'époque babylonienne présentent d'autres sources possibles de cette apparition). La première utilisation du terme $\lambda o \gamma \iota \kappa \eta^{'}$ (*logikè*) comme d'un objet d'étude est attribuée à Xénocrate, mais on attribue davantage à Aristote et aux stoïciens les premières études de la logique à part entière. Soulignons d'ailleurs son importance en rappelant la célèbre métaphore stoïcienne de l'œuf, représentant une partition de la philosophie (c'est-à-dire pour les stoïciens, de la science dans un sens très général) : la physique constitue le jaune, l'éthique constitue le blanc, et la logique la coquille.

La logique concerne alors l'étude du raisonnement, et vise à donner les conditions sous lesquelles un argument est ou non valable. Si l'on est assez loin de la logique mathématique contemporaine, arrêtons-nous sur le fait que la syllogistique aristotélicienne propose néanmoins une étude générale de ce qui sera appelé plus tard *conséquence logique*. En effet, si Aristote ne dispose pas d'un langage propre dédié à la logique, la validité des syllogismes est déjà pensée à un niveau général et abstrait du langage. Prenons par exemple le syllogisme suivant :

$$\frac{\begin{array}{c}\text{Tous les } A \text{ sont } B \\ \text{Tous les } B \text{ sont } C\end{array}}{\text{Donc Tous les } A \text{ sont } C}$$

Nommé *syllogisme Barbara* par les logiciens médiévaux, cette inférence est valide quelle que soit l'interprétation donnée aux noms A, B, et C. C'est en cela que l'on dispose déjà d'une étude de l'argumentation qui soit *abstraite*.

1.2.1.2 Les médiévaux

Les corpus aristotéliciens ont, à la suite de leur géniteur, eu un parcours particulièrement riche, et pérenne. Le corpus dédié à la logique en particulier, l'*Organon*, a été décortiqué, digéré, et transmis à travers diverses époques et civilisations. D'abord repris par les néoplatoniciens comme Porphyre ou Plotin, ces textes ont transité entre l'Europe occidentale et le monde arabe, en passant par des savants tels qu'Avicenne, Averroës ou Al-Farabi. Puis à travers notamment l'école de Tolède, ils ont été réimportés en Europe, et ont repris de l'importance dans les études philosophiques et théologiques du haut Moyen-Âge.

Les penseurs de cette époque qui étudient la logique sont alors souvent philosophes et théologiens. On peut citer Thomas d'Aquin, Abélard, Buridan, ou

encore Guillaume d'Ockham. La logique est alors toujours intrinsèquement liée au langage naturel, car elle étudie la validité des argumentations. C'est toujours une forme de syllogistique qui est à l'œuvre, mais dans un contexte où les énoncés dont on examine la validité sont le plus souvent chargé de contenus théologiques, et il s'agit de déterminer si une proposition est ou non conséquence logique d'autres propositions provenant par exemple de textes sacrés. Une grande variété de syllogismes est cataloguée en *sophismata*, et leur analyse logique consiste alors à compenser l'ambiguïté du langage naturel (ou du latin, le plus souvent) par une étude rigoureuse.

La logique ne dispose pas d'un langage à part qui lui permettrait de s'extraire des difficultés de la langue du discours. On considère souvent qu'il faut attendre le XIX^{e} siècle pour voir apparaître une mathématisation de la logique.

1.2.1.3 La *vraie* logique et la crise des fondements

L'histoire de la logique est parfois décrite comme commençant au XIX^{e} siècle, à partir du moment où il existe un langage, une écriture, propres à la logique. Les époques évoquées ci-dessus en seraient alors une préhistoire.

La paternité de la logique moderne est communément attribuée à Gottlöb Frege. Son *Idéographie* [Fre99] donne une façon de représenter la structure logique du raisonnement de façon formelle. À la différence des approches citées jusqu'à maintenant, un glissement s'est produit en ce qui concerne le contenu des énoncés étudiés. En effet, les propositions que Frege cherchait à formaliser dans une logique générale sont des formules mathématiques, et le langage développé a pour objectif de permettre une formalisation de l'arithmétique.

Nous sommes dans un contexte où les développements des mathématiques, en particulier avec l'introduction des géométries non euclidiennes, semblent appeler à l'installation d'un cadre unifié, à l'explicitation de bases solides. Cette démarche a donné lieu à plusieurs tentatives, dont les *Fondements de l'arithmétique* [Fre80] de Frege, basée sur la théorie des ensembles de Cantor.

Nous arrivons alors à la fameuse *crise des fondements*, à l'orée du XX^{e} siècle. Au moment de publier le deuxième volume de ses *Fondements de l'arithmétique*, Frege reçoit une lettre de Bertrand Russell indiquant une contradiction dans son système. Le volume est publié accompagné d'un appendice où l'auteur admet qu'il vient de recevoir une note de Russell menant à l'effondrement de son travail. Le paradoxe est en effet issu d'une des lois fondamentales de l'ouvrage, la fameuse loi V, qui postule l'existence, pour tout prédicat P, de son extension $\{x \mid P(x)\}$, c'est-à-dire l'ensemble de tous les éléments ayant la propriété P. Le paradoxe de Russell s'exprime alors simplement : en prenant comme prédicat $P(x) = x \notin x$, l'existence de l'ensemble $\{x \mid P(x)\}$ est nécessairement contradictoire (il s'appartient à lui-même si et seulement s'il ne s'appartient pas à lui-même).

Russell et Whitehead proposent un nouveau système de fondements des mathématiques devant dépasser les contradictions de la sorte. Ils publient les *Principia Mathematica* [WR27]. Mais à nouveau, ces fondements logiques des mathématiques vont être victimes de l'Histoire. Les théorèmes d'incomplétude de Gödel, dans les années 1930, vont établir que dans les systèmes formels assez puissants pour exprimer les propositions de l'arithmétique, il existera toujours des formules qui seront vraies si et seulement si elles ne sont pas démontrables. Ces résultats, fondamentaux s'il en est, ont montré qu'il fallait repenser toute

démarche visant à donner un système logique général pour les fondements des mathématiques.

Au cœur de ces limitations, se place la notion de démonstrabilité des propositions. Nous avons jusqu'ici traité de système général permettant d'exprimer des propositions à l'aide des outils fournis par la logique. Mais un tel système doit permettre également d'écrire les démonstrations des formules exprimées dans un cadre tout aussi général. C'est le rôle de la théorie de la démonstration dont nous disons quelques mots ci-dessous.

1.2.1.4 La théorie de la démonstration

La syllogistique aristotélicienne constituait déjà, dans ce que nous avons appelé *préhistoire de la logique*, une théorie de la démonstration. On pouvait dire qu'une proposition était valide si elle pouvait être obtenue depuis une des différentes formes établies de syllogisme (comme *Barbara*, présenté plus haut) à partir d'autres propositions valides.

Dans un contexte où les propositions sont des formules écrites dans un langage formel, les premiers systèmes de démonstration sont attribués à David Hilbert. Par exemple, pour un langage simple, doté de variables propositionnelles $A, B, \ldots$, et de l'unique connecteur $\to$, le *système à la Hilbert* associé consiste en deux schémas d'axiomes, et d'une règle d'inférence. Les deux schémas d'axiomes sont les suivants :

— (A1) : $\varphi \to (\psi \to \varphi)$

— (A2) : $(\varphi \to \psi \to \theta) \to (\varphi \to \psi) \to (\varphi \to \theta)$

L'unique règle d'inférence, celle qui permet de passer d'une formule à une autre, est le *modus ponens*, que nous avons déjà rencontré :

$$\frac{\varphi \to \psi \qquad \varphi}{\psi}$$

À l'instar des syllogismes aristotéliciens, les symboles $\varphi, \psi, \theta, \ldots$ peuvent être remplacés par n'importe quelle formule du langage. C'est pour cette raison que l'on parle de *schéma d'axiomes*.

Ce cadre n'est pas des plus commodes pour l'écriture des démonstrations. La preuve de la formule $A \to A$ par exemple, n'est pas du tout immédiate. Pour autant, en plus de constituer le premier système de démonstration, cette construction aura un autre intérêt historique sur lequel nous reviendrons plus loin.

Il faut attendre 1934 pour voir apparaître un système de démonstration plus pratique, et plus naturel. Cette année-là, Gerhard Gentzen introduit dans un article, *Recherches sur la déduction logique* [Gen55] la *Déduction Naturelle*. Cette dernière est appelée ainsi en référence à l'aspect intuitif de la façon dont les preuves peuvent s'y écrire.

Ce système peut se présenter sous forme de *séquents* (ou *séquences* dans certaines traductions). Un séquent, pour la déduction naturelle, est une suite de formules, par exemple $(A_1, \ldots, A_n)$ souvent abrégée en Γ, les hypothèses, suivie d'une conclusion A. Un séquent de la sorte s'écrit $A_1, \ldots, A_n \vdash A$, et se lit : « la formule A est démontrée sous les hypothèses $A_1, \ldots, A_n$ ». Le symbole $\vdash$ est donc une relation binaire représentant la démonstrabilité. Les axiomes sont

plus simples, ils sont de la forme $\overline{A \vdash A}$, et la règle permettant de construire une implication s'écrit comme suit :

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}$$

Quant au *modus ponens* des systèmes à la Hilbert s'écrit comme suit :

$$\frac{\Gamma_1 \vdash A \to B \qquad \Gamma_2 \vdash A}{\Gamma_1, \Gamma_2 \vdash B}$$

Le premier mérite de Gentzen est donc d'avoir donné une manière formelle mais compréhensible d'écrire les preuves de la logique (du premier ordre).

Le second, qui n'est pas le moindre, est l'introduction d'une notion fondamentale, celle de *dynamique* des démonstrations. Gentzen introduit un algorithme permettant de passer d'une preuve à une autre : l'élimination des coupures. Nous en avons déjà parlé mais rappelons simplement que cette procédure permet de passer d'une preuve qui fait des *détours* (les fameuses coupures), qui utilise des lemmes, à une preuve directe sans résultat intermédiaire (on parle de preuve sans coupure ou de preuve analytique).

Le théorème principal de Gentzen, le théorème d'élimination des coupures (*Hauptsatz* dans le texte, pour *Théorème fondamental*), établit que toute démonstration formelle peut être réécrite en une preuve analytique. La procédure de transformation des preuves décrit alors un algorithme pour parvenir à ce résultat.

On peut désormais considérer les preuves comme des objets mathématiques, avec des définitions précises, des règles de construction, un langage, et maintenant munies d'un ensemble d'opération : les différentes règles d'élimination des coupures. Les démonstrations formelles constituent donc un espace structuré. La correspondance de Curry-Howard va permettre d'établir un lien entre ces dernières, et une notion plus générale de calcul, ou d'algorithme, comme nous allons le voir.

Par souci de cohérence chronologique, il nous faut évoquer ici le λ-calcul, bien qu'il en sera davantage question plus loin. Contentons-nous pour l'instant de l'information suivante : cependant que Gentzen inventait la théorie de la démonstration moderne, c'est-à-dire dans les années 1930, le mathématicien américain Alonzo Church inventait, lui, ce que les logiciens considèrent souvent comme le premier langage de programmation, à savoir le λ-calcul, qui tiendra aussi une place cruciale dans la correspondance de Curry-Howard.

1.2.1.5 La correspondance de Curry-Howard

Dans les années 1950, Haskell Brooks Curry publie avec Robert Feys un manuel de logique combinatoire [CF58]. La logique combinatoire a été introduite pour montrer, entre autres motivations, la possibilité de développer un système similaire au λ-calcul, mais sans variables, à la suite de travaux de Schönfinkel. Le langage est composé de *combinateurs*, et muni de règles de calcul associées à ces combinateurs. Par exemple, le combinateur $\mathbf{K}$ est muni de la règle suivante : pour tous termes t_1, t_2, $\mathbf{K}t_1t_2 = t_1$. Le combinateur $\mathbf{S}$, fonctionne comme suit : $\mathbf{S}t_1t_2t_3 = (t_1t_2)(t_1t_3)$.

La logique combinatoire n'aura pas pour elle-même une postérité retentissante. Pour autant, une page du manuel de Curry et Feys rentrera dans l'histoire

comme la première manifestation d'un lien entre la logique et le calcul automatisé, pour ne pas parler encore d'informatique.

Dans cette fameuse page de 1948, attribuée à Curry et qui donnera la première partie du nom de la fameuse correspondance, l'auteur explique, à titre de remarque, que les combinateurs fonctionnent comme les preuves des systèmes à la Hilbert. On se rappelle des fameux systèmes, avec les deux axiomes et le *modus ponens* comme unique règle. Curry, donc, montre que l'on peut associer au combinateur $\mathbf{K}$ l'axiome $A_1 \ : \varphi \to (\psi \to \varphi)$, et au combinateur $\mathbf{S}$ l'axiome $A_2 \ : (\varphi \to \psi \to \theta) \to (\varphi \to \psi) \to (\varphi \to \theta)$.

Cette association ne porte pas encore le nom de typage, mais est introduite comme une *analogie frappante entre la théorie de la fonctionnalité et la théorie de l'implication dans l'algèbre propositionnelle*. Il est remarqué que, en suivant l'association d'un combinateur à une formule implicative comme ci-dessus, l'application d'un combinateur à un autre (qui représente en premier lieu l'application d'une fonction à un argument) correspond à la règle du *modus ponens*. En suivant cette analogie, il est établi une correspondance entre les associations de combinateurs et les théorèmes de la logique propositionnelle implicative.

L'exemple le plus simple est sans doute la preuve de l'identité : Pour prouver la formule $A \to A$ dans un système à la Hilbert, il faut appliquer deux fois le *modus ponens*, d'abord de l'axiome A_2 à l'axiome A_1, puis de la formule obtenue à l'axiome A_1 à nouveau. Ce qui correspond en logique combinatoire à l'association $\mathbf{SKK}$. On vérifie que ce combinateur correspond bien à l'identité, en s'assurant que pour tout terme t, $(\mathbf{SKK})t = (\mathbf{K}t)(\mathbf{K}t) = \mathbf{K}(t\mathbf{K}t) = t$.

La deuxième découverte de la correspondance, fatalement due à un certain William Alvin Howard, a lieu dans les années 1970. Ce dernier établit à nouveau un lien entre un système de calcul et un système logique, mais à un plus haut niveau. Le système de calcul est le λ-calcul de Church, et le système logique est la déduction naturelle de Gentzen. À l'instar de Curry, Howard semble ne pas accorder une grande importance à ce résultat, et ne le publiera que très tardivement (il s'agira d'abord de notes photocopiées qui circuleront parmi plusieurs chercheurs).

Quelque chose de plus précis émerge de cette nouvelle analogie : la correspondance entre la dynamique des preuves et la dynamique du calcul. Le λ-calcul de Church a pour caractéristique de pouvoir exprimer toutes les fonctions calculables (c'est la fameuse *thèse de Church*, qui est généralement admise) dans un langage très simple, et muni d'une unique règle : la β-réduction.

Nous avons dit que Gentzen avait introduit une notion de dynamique des preuves en définissant la procédure d'élimination des coupures. C'est cette opération qui correspond *via* Curry-Howard à la β-réduction. Plus précisément, si un λ-terme M se β-réduit en un terme N, alors la preuve en déduction naturelle qui correspond à M se transforme en la preuve associée à N par une étape d'élimination des coupures.

Le λ-calcul étant une forme primaire de langage de programmation, et la β-réduction un prototype d'exécution informatique, cela explique pourquoi la correspondance de Curry-Howard est également désignée par l'appellation *correspondance preuves-programmes*.

Le pan de la logique attaché à la théorie de la démonstration devient alors intrinsèquement lié à l'évolution des concepts d'informatique fondamentale. Et l'étude des systèmes de preuves s'accompagne souvent de l'étude des programmes que l'on peut y associer. En particulier, établir des propriétés sur l'élimination

des coupures d'un système de démonstrations implique immédiatement des propriétés analogues au niveau des termes du langage de programmation qui y est associé.

Pour cette raison, on parle parallèlement de la dynamique des preuves et de la sémantique opérationnelle des programmes. L'étude des algorithmes et du calcul automatisé se mêle dès les années 1980 aux problématiques des théoriciens de la démonstration.

1.2.1.6 Consommation de ressources et Logique Linéaire

Dans les années 1980, Jean-Yves Girard étudie les propriétés sémantiques du λ-calcul parallèlement aux propriétés des systèmes de preuve associés. Une observation naît de cette étude : la logique que l'on attache au λ-calcul *via* la correspondance de Curry-Howard (à savoir, la Logique Intuitionniste) n'est pas assez fine pour rendre compte précisément de la dynamique des programmes. La dynamique en question ici est la suivante : l'application d'un terme M de type $A \to B$ à un terme N de type A ne rend pas compte du fait que M puisse avoir besoin de dupliquer ou d'effacer N pour produire un terme de type B.

L'idée qui donnera lieu à l'invention de la Logique Linéaire est alors que la logique intuitionniste ne dispose pas d'un langage d'une granularité assez fine pour rendre compte de la duplication ou de l'effacement des arguments lors du calcul.

En développant un nouveau langage, essentiellement propositionnel, Girard apporte une nouvelle dimension à Curry-Howard. Une nouvelle logique apparaît, accompagnée de systèmes de démonstrations, mais qui ne permet ni d'exprimer ni de démontrer plus de choses qu'avant, du point de vue des formules. En revanche, cette logique, quand considérée comme un pendant du λ-calcul, permet d'isoler et d'étudier la consommation de ressources dans l'exécution du calcul d'une façon nouvelle.

L'élimination des coupures dans ce cadre peut désormais être étudiée pour examiner la complexité des programmes, en termes de consommation de ressources, de façon bien plus précise que ce que ne le permettait la logique intuitionniste.

Girard invente dans le même temps une syntaxe pour les démonstrations de la Logique Linéaire permettant de les présenter sous forme graphique, les *réseaux de preuve*. Ces réseaux, qui sont l'objet d'une grande partie de ce mémoire de thèse (chapitre 2), permettent d'étudier les phénomènes de consommation de ressources à travers une dynamique qui est de l'ordre de la réécriture locale de graphes. Nous reviendrons sur ces propriétés, ainsi que sur leurs vices et vertus, en introduction du chapitre qui leur est dédié.

La Logique Linéaire a eu un succès assez remarquable, et elle a donne aujourd'hui lieu à diverses directions de recherche, notamment concernant la complexité algorithmique, et la sémantique quantitative des programmes.

1.2.2 Le calcul

Nous avons décrit comment la logique avait dans ses développements mené à des considérations proches de l'informatique et du calcul automatisé. Nous revenons sur certaines de ces notions avant de conclure cette rétrospective.

1.2.2.1 La thèse de Church-Turing

Revenons un peu en arrière. À la suite de la crise des fondements, et des théorèmes d'incomplétude de Gödel, il est clair que l'on ne trouvera pas de système général dans lequel on pourra tout calculer, pour le dire grossièrement. Une question qui se pose alors est « que peut-on calculer, où, et comment ? ». Une réponse à cette question apparaît dans la thèse de Church-Turing. Church, en introduisant le λ-calcul, caractérise ce que l'on considère comme les *fonctions calculables*. Parallèlement, les machines de Turing, du savant éponyme, caractérisent une notion de calculabilité plus proche de ce que l'on appellera *algorithmes*.

Les deux modèles (λ-calcul et machines de Turing) caractérisent pourtant deux notions de calculabilité rigoureusement équivalentes : toute fonction calculable représentée par un λ-terme peut être calculée par une machine de Turing, et toute procédure exécutée par une machine de Turing peut être encodée dans un λ-terme.

À travers cette thèse, les pères de l'informatique que sont Church et Turing donnent à la notion de calcul un contenu algorithmique : trouver le résultat d'un problème mathématique n'est plus ici la priorité, il s'agit de trouver une façon d'automatiser les résolutions, en décrivant des procédures dans un langage adapté. Les langages de programmation sont souvent vus comme les héritiers de ces constructions, bien que les machines de Turing et le λ-calcul soient des objets mathématiques abstraits ne disposant pas au moment de leur conception de matériel permettant de rendre concret leur aspect *automatique*.

1.2.2.2 Les algorithmes

Les machines de Turing, en apportant un cadre général à la calculabilité, permettent de caractériser les exécutions de suites d'instructions, les fameux *algorithmes*. Le support a changé depuis l'apparition des ordinateurs, mais le principe reste le même : un objet (désormais physique) auquel on peut fournir une procédure, et qui l'exécute. On parle d'*universalité* pour désigner la capacité de ces machines à exécuter *n'importe quelle* procédure qui soit calculable. En omettant bien sûr les limitations physiques ou temporelles (taille de la mémoire, puissance de calcul), qui sont des limitations contingentes et non des limitations de principe.

Les algorithmes, ces procédures, ou suites d'instruction (ou *recettes de cuisine*, pour suivre la métaphore la plus populaire et peut-être la plus parlante), ne sont pas apparues avec les ordinateurs. On a déjà vu que Hilbert, Gödel, Church, Turing, réfléchissaient déjà aux procédures automatiques de résolution, bien avant l'apparition des ordinateurs. Pour autant, cette notion est encore plus ancienne.

Le terme d'algorithme provient du nom du savant perse Muhammad Ibn Mūsā al-Khuwārizmī, appelé plus généralement al-Khuwārizmī, et latinisé en *algoritmi*. On ne peut pas non plus dire que c'est l'inventeur de l'algorithme, car ses traités consistent davantage en une classification d'algorithmes déjà existants. Le premier algorithme dont il existe une trace est celui d'Euclide (livre VII des *Éléments*), permettant de calculer le plus grand commun diviseur, et encore enseigné dans les cours généraux de mathématiques.

Le point commun entre ces procédures antiques et les algorithmes contempo-

rains, qui ont pris l'importance que l'on sait, est qu'elles peuvent être appliquées par une entité (un ordinateur, un écolier, un cuisinier,...) sans que celle-ci ait besoin de comprendre pourquoi, ou comment la tâche est résolue.

1.2.2.3 La parallélisation du calcul

Si les contraintes d'espace et de puissance de calcul ne sont pas, comme nous l'avons dit, des limitations de principe à la calculabilité, elles sont bien réelles et demandent des solutions qui ne relèvent pas que de l'ingénierie. En effet, si la technologie permet l'implémentation de programmes de plus en plus puissants, la branche scientifique relevant de la conception des algorithmes est également responsable de l'optimisation des exécutions.

Un exemple [2] de cette relation entre développements industriels et enjeux scientifique est la répartition des algorithmes. Jusque dans les années 2000, la puissance des processeurs doublait environ tous les dix-huit mois. Donc un algorithme qui était trop complexe pour être exécuté par une machine donnée pouvait attendre que la puissance augmente jusqu'à ce que son implémentation soit possible. Mais dans les années 2000, justement, il a été remarqué chez Intel que la chaleur dégagée par chaque processeur étant loin d'être négligeable, ce rythme-là ne pourrait être suivi à long terme sans que l'on atteigne en quelques décennies la température de la surface du Soleil.

La solution apportée à ce problème, pour pouvoir continuer à augmenter la puissance des machines, a été la parallélisation du calcul. Plutôt que d'augmenter la puissance des processeurs, la démarche adoptée consistait à utiliser plusieurs processeurs en même temps. Ces nouvelles installations ont demandé de la part des algorithmiciens un changement de paradigme. En effet, développer des algorithmes qui puissent être exécutés sur des machines à plusieurs processeurs de façon à tirer parti de cette distribution est loin d'être une chose triviale.

En particulier, l'aspect *universel* que revêtaient les machines de Turing en pouvant exécuter toute procédure calculable n'est plus d'actualité dans un contexte où le calcul est distribué. Une tâche élémentaire comme la décision d'une valeur commune aux différents processus prenant part au calcul est même impossible. Ce résultat bien connu, sous le nom d'*impossibilité du consensus* sera au cœur du chapitre 4.

1.3 Remarques particulières : contenu et contributions

1.3.1 Logique Linéaire différentielle et à ressources, développement de Taylor

Dans le chapitre 2, nous nous intéressons au fragment multiplicatif exponentiel de la Logique Linéaire (**MELL**), et à la Logique Linéaire à ressources (voir la figure 2.2 pour les règles de dérivation). **MELL** est le fragment dans

2. Emprunté à Rachid Guerraoui (leçon inaugurale au Collège De France `https://www.college-de-france.fr/site/rachid-guerraoui/inaugural-lecture-2018-10-25-18h00.htm`)

lequel s'exprime la duplicabilité des ressources, que nous avons évoquée en section 1.2.1.6, à travers le connecteur « ! ». Cet aspect s'exprime par la règle de promotion qui permet à des sous-preuves entières d'être effacées ou répliquées pendant l'élimination des coupures.

La Logique Linéaire à ressources (que nous écrivons $\mathbf{DLL}_0$, en référence à la Logique Linéaire Différentielle) ne comporte pas cette règle de promotion, et ne permet donc pas à la réduction de dupliquer ou effacer des sous-preuves. La réduction dans ce fragment, que nous appellerons *réduction à ressources*, est essentiellement linéaire en ce sens-là.

Nous nous concentrerons sur l'élimination des coupures dans $\mathbf{DLL}_0$, à travers les réseaux de preuve, pour examiner des propriétés d'approximation et de simulation. En effet, les réseaux de $\mathbf{DLL}_0$ permettent d'exprimer le *développement de Taylor*, qui consiste à associer à une preuve P de **MELL** (qui permet les duplications et effacements arbitraires) une somme – généralement infinie – de preuves de $\mathbf{DLL}_0$ qui, prise comme un tout, se comporte comme P. Cette correspondance se concrétise au niveau de la sémantique opérationnelle et au niveau de la sémantique dénotationnelle. Nous développons ce point en introduction du chapitre 2 (en section 2.1.2 en rappelant les travaux déjà existant sur le développement de Taylor).

Notre contribution se fait en deux temps : nous établissons d'abord des propriétés géométriques sur la réduction à ressources, puis nous les appliquons au développement de Taylor pour montrer que définir des opérations de réduction sur des sommes infinies ne pose pas de problème de coefficients. Ensuite, nous utilisons ces opérations pour montrer que l'élimination des coupures de **MELL** peut être simulée dans le développement de Taylor.

1.3.2 λ-calculs et stratégies

Dans le chapitre 3, nous nous concentrons sur différentes variantes du λ-calcul. Nous étudions principalement *l'Appel-Par-Pousse-Valeur* (*Call-By-Push-Value*), qui permet d'encoder différentes stratégies d'évaluation, en particulier l'Appel-Par-Nom et l'Appel-Par-Valeur.

Nous introduisons un langage de termes à ressources qui permet de définir le développement de Taylor pour l'Appel-Par-Pousse-Valeur. Nous étudions également les propriétés de traduction des stratégies d'évaluation dans ce langage plus général au niveau justement des ressources. Nous utilisons pour cela les développements de Taylor en Appel-Par-Nom et en Appel-Par-Valeur qui existent déjà, et montrons que nos constructions sont compatibles avec ces dernier.

Une des difficultés que nous traitons est la duplicabilité de l'Appel-Par-Pousse-Valeur qui, contrairement au λ-calcul usuel ou au réseaux de preuve, n'est pas entièrement capturée par les opérateurs exponentiels. Cela est lié à la structure de coalgèbre qui intervient dans la sémantique dénotationnelle lorsque l'on interprète les termes. Nous travaillons donc sur la syntaxe à ressources de façon à rendre compte de ces spécificités sémantiques.

Enfin, nous établissons des résultats similaires à ceux du chapitre 2 en montrant que la sémantique opérationnelle de l'Appel-Par-Pousse-Valeur peut être capturée par le développement de Taylor que nous aurons construit.

1.3.3 Algorithmique distribués : outils topologiques pour le consensus et l'accord d'ensemble probabilistes

Dans le chapitre 4, nous nous intéressons aux sémantiques topologiques des systèmes distribués avec mémoire partagée. Nous rappelons les constructions et résultats de Herlihy et Shavit, qui ont développé des outils topologiques permettant de déterminer l'existence ou la non existence d'algorithmes distribués pour une tâche donnée. En particulier, nous nous concentrons sur la reformulation topologique du résultat fondamental de Fischer, Lynch et Patterson : l'impossibilité du consensus.

Nous définissons une approche topologique de ces objets, mais en intégrant cette fois les comportements probabilistes des algorithmes distribués asynchrones. À l'aide de ces outils, nous parvenons à adapter au modèle topologique un résultat de borne inférieure de probabilité pour le consensus aléatoire, initialement dû à Attiya et Censor.

Notre première contribution consiste dans la transposition de l'argument d'Attiya et Censor dans le langage des complexes simpliciaux. Le cœur de cette transposition se situe dans la notion d'*indistinguabilité* entre exécutions asynchrones, qui est particulièrement commode à exprimer à travers les constructions topologiques adaptées d'Herlihy et Shavit.

Notre seconde contribution concerne une autre tâche, l'*accord d'ensemble* k, qui est comparable au consensus, et pour laquelle une preuve topologique d'impossibilité a également été fournie par Herlihy et Shavit. Nous donnons également une borne inférieure de probabilité pour l'accord d'ensemble k, qui est cohérente avec les résultats précédents au sujet du consensus.

1.4 Terminologie et notations

L'ensemble des entiers naturels est écrit $\mathbf{N}$. La cardinalité d'un ensemble X est notée $\mathbf{card}(X)$.

Soit X un ensemble et $k \in \mathbf{N}$. X^k représente le produit cartésien ensembliste $\underbrace{X \times \ldots \times X}_{k}$. Si $k, l \in \mathbf{N}$, $(a_1, \ldots, a_k) \in X^k$, et $(b_1, \ldots, b_l) \in X^l$. On représente par $(a_1, \ldots, a_k) :: (b_1, \ldots, b_l)$ le $k+l$-uplet $(a_1, \ldots, a_k, b_1, \ldots, b_l) \in X^{k+l}$. Pour un k-uplet $\overrightarrow{a} = (a_1, \ldots, a_k)$, on note sa longueur $|\overrightarrow{a}| = k$. L'unique suite $\overrightarrow{a}$ telle que $|\overrightarrow{a}| = 0$ est notée ϵ.

Un multiensemble d'éléments d'un ensemble X est une fonction de X dans $\mathbf{N}$, et est écrit $[x_1, \ldots, x_k]$, et souvent abrégé $\overline{a}$, où les x_i sont des éléments de X. Pour tout $x \in X$, $\overline{a}(x)$ est l'entier représentant le nombre de répétitions de x dans le multiensemble $\overline{a}$. $X^!$ représente l'ensemble des multiensembles finis d'éléments de X, c'est-à-dire $\{[x_1, \ldots, x_k] \mid k \in N, x_i \in X\}$. Pour un multiensemble $\overline{a}$, on note sa taille $|\overline{a}| = \sum_{x \in X} \overline{a}(x)$. On notera abusivement l'apartenance à un multiensemble $x \in \overline{a}$, au lieu de $\overline{a}(x) \neq 0$. La somme de deux multiensembles $\overline{a}_1$ et $\overline{a}_2$ sur X est notée $\overline{a}_1 + \overline{a}_2$, et définie par la somme de fonctions, en posant pour tout $x \in X$, $(\overline{a}_1 + \overline{a}_2)(x) = \overline{a}_1(x) + \overline{a}_2(x)$.

Quand nous comparerons différents types de calcul, nous prendrons soin de ne pas confondre les attributs *pur* et *non typé*. Le premier correspondra en général au λ-calcul qui n'est pas enrichi de constructions pour les entiers, les paires ou injections, ou autres. Le second correspond à un calcul qui n'est pas

muni d'un système de typage.

Pour une notion de réduction définie entre des objets, telle que l'on puisse écrire $a \to b$, on désignera b en parlant du *réduit*, et a en parlant de l'*antiréduit*.

Nous considérerons à plusieurs reprises des combinaisons linéaires infinies d'objets (λ-termes, réseaux), définis dans une syntaxe de termes X particulière. Ces termes prendront des coefficients dans un semi anneau arbitraire $\mathbf{S}$ avec fractions : c'est-à-dire dans lequel tout entier $k \neq 0 \in \mathbf{N}$ admet un inverse multiplicatif, noté $\frac{1}{k}$. Pour une combinaison $\varphi = \sum_{i \in I} a_i \cdot t_i \in \mathbf{S}^X$, et pour un terme $t \in X$, on représente le coefficient de t dans φ par la notation $(\varphi)_t$, qui correspond à $\prod_{t_i = t} a_i$.

Pour un objet t dans lequel est définie la notion d'occurrence de variable, on écrira souvent $t[s_1/x_1, \ldots, s_n/x_n]$ pour la substitution multilinéaire des objets s_i aux différente occurrences d'une variable dans t. C'est-à-dire que dans cette notation, et sauf si explicitement précisé, $x_1, \ldots, x_n$ ne désignent pas des variables distinctes mais des occurrences distinctes d'une même variable.

Chapitre 2

Structures graphiques

Il ne faut jamais parler sèchement à un numide.
Un romain [a]

a. Dans *Le domaine des dieux*, Gosciny et Uderzo, Dargaud, 1971

2.1 Introduction

2.1.1 Les réseaux de preuve et leur syntaxe

2.1.1.1 Grandeur des réseaux (1) : à propos des preuves

Les réseaux de preuve de la Logique Linéaire qui sont l'objet de ce chapitre ont été introduits par Girard dans son article fondateur *Linear Logic* [Gir87]. Les réseaux forment une syntaxe graphique pour les preuves de certains fragments de la Logique Linéaire, de façon à ce que soient représentées par le même réseau plusieurs preuves qui seraient distinctes dans le calcul des séquents (le calcul des séquents pour le fragment de la Logique Linéaire qui nous occupera dans ce chapitre est illustré en figure 2.2). Mais l'étude et l'utilisation des réseaux entérinent généralement le fait que les preuves représentées par le même réseau ont toutes les bonnes raisons d'être ainsi identifiées. En particulier, cela demande d'accepter qu'il existe des preuves qui sont différentes dans le calcul des séquents, mais qui le sont pour de *mauvaises raisons*.

Nous donnons un exemple de trois preuves simples de la sorte en figure 2.1 Ces trois démonstrations (de la même formule) diffèrent sur l'ordre d'introduction des connecteurs. On peut donner au moins deux arguments allant dans le

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash A, A^\perp} \quad \dfrac{}{\vdash B, B^\perp}}{\vdash A, A^\perp \otimes B^\perp, B} \quad \dfrac{}{\vdash C, C^\perp}}{\vdash A, A^\perp \otimes B^\perp, B \otimes C, C^\perp}}{\vdash A \parr (A^\perp \otimes B^\perp), B \otimes C, C^\perp} \qquad \dfrac{\dfrac{\dfrac{}{\vdash A, A^\perp} \quad \dfrac{\dfrac{}{\vdash B, B^\perp} \quad \dfrac{}{\vdash C, C^\perp}}{\vdash B^\perp, B \otimes C, C^\perp}}{\vdash A, A^\perp \otimes B^\perp, B \otimes C, C^\perp}}{\vdash A \parr (A^\perp \otimes B^\perp), B \otimes C, C^\perp}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash A, A^\perp} \quad \dfrac{}{\vdash B, B^\perp}}{\vdash A, A^\perp \otimes B^\perp, B} \quad \dfrac{}{\vdash C, C^\perp}}{\vdash A \parr (A^\perp \otimes B^\perp), B \otimes C, C^\perp}$$

FIGURE 2.1 – Trois preuves similaires de la même formule en calcul des séquents

sens d'un *oubli* de cet ordre. Le premier est relatif à l'intuition, et le second à la sémantique :

1. Ces trois preuves ont la même *forme*, la même *structure*. Elles obtiennent les formules de la conclusion de la même façon.
2. Ces trois preuves ont la même interprétation dans la plupart des modèles dénotationnels.

Pour autant, le second point appelle quelques réflexions. Le fait que deux preuves aient la même interprétation dans un modèle semble être une condition nécessaire à leur identification, mais elle n'est pas suffisante. En effet, la sémantique dénotationelle identifie généralement toutes les preuves ayant la même forme normale. Cela relève de la question complexe (philosophique, mais pas seulement) de l'identité des preuves. Widebäck, dans un ouvrage qui développe différentes approches à ce problème [Wid01] parle de la *conjecture de Prawitz* pour désigner la démarche visant à considérer comme identiques les preuves ayant la même forme normale (renvoyant aux travaux dudit Prawitz [Pra75]. Voir aussi à ce sujet les travaux de Došen [Dos02]).

Si cette approche se défend, et provient notamment des succès de la théorie des catégories relativement aux sémantiques des λ-calculs et des systèmes de preuve, elle peut être contestée, ou au moins nuancée.

L'argument sans doute le plus fort allant contre cette identification concerne la dynamique des preuves. En effet, la correspondance de Curry-Howard est trop présente dans les esprits du XXI$^{\text{e}}$ siècle pour ne pas voir le problème : si une preuve est un objet calculatoire, pourquoi oublier le calcul ? En effet, identifier les preuves de même forme normale donne un quotient que l'on peut juger trop large si l'on s'intéresse aux processus de réduction, comme l'élimination des coupures, vus comme une dynamique non triviale permettant de passer d'une preuve à une autre, en général différente de la première. On peut citer le célèbre article de Boolos *Don't eliminate the cut* [Boo84] montrant que l'on peut écrire une preuve en déduction naturelle tenant sur une page A4, et dont la forme normale contient davantage de caractères que ne se sont écoulées de nanosecondes depuis le Big Bang [1]. L'étude dynamique des preuves ne peut se

1. C'était le cas en 1984, nous n'avons pas refait le calcul pour 2019, l'année d'écriture de ce mémoire.

Groupe identité	$\dfrac{}{\vdash A, A^{\perp}}$ Axiome	$\dfrac{\vdash \Gamma, A \qquad \vdash \Delta, A^{\perp}}{\vdash \Gamma, \Delta}$ Coupure

	Positifs	Négatifs
Groupe multiplicatif	$\dfrac{\vdash \Gamma, A \qquad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B}$ Tenseur $\dfrac{}{\vdash 1}$ Unité	$\dfrac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}$ Parr $\dfrac{\vdash \Gamma}{\vdash \Gamma, \perp}$ Co-unité
Groupe additif	$\dfrac{\vdash \Gamma, A_i}{\vdash \Gamma, A_1 \oplus A_2}$	$\dfrac{\vdash \Gamma, A \qquad \vdash \Gamma, B}{\vdash \Gamma, A \,\&\, B}$ Avec $\dfrac{}{\vdash \Gamma, \top}$ Co-unité
Groupe exponentiel	$\dfrac{\vdash \Gamma, !A \qquad \vdash \Delta, !A}{\vdash \Gamma, \Delta, !A}$ Cocontraction $\dfrac{}{\vdash !A}$ Coaffaiblissement $\dfrac{\vdash \Gamma, A}{\vdash \Gamma, !A}$ Codéréliction $\dfrac{\vdash ?A_1, \ldots, ?A_k, A}{\vdash ?A_1, \ldots, ?A_k, !A}$ Promotion	$\dfrac{\vdash \Gamma, ?A, ?A}{\vdash \Gamma, ?A}$ Contraction $\dfrac{\vdash \Gamma}{\vdash \Gamma, ?A}$ Affaiblissement $\dfrac{\vdash \Gamma, A}{\vdash \Gamma, ?A}$ Déréliction

FIGURE 2.2 – Règles de dérivation pour les séquents de la Logique Linéaire différentielle

satisfaire de la conjecture de Prawitz, car elle cherche à comprendre *ce qui s'est passé* entre ces preuves.

Nous cherchons donc à travailler dans un système qui identifie les preuves ayant la *même forme* (notion qui n'est pas définie clairement, sans quoi le problème de l'identité des preuves que nous venons d'évoquer n'en serait pas un), mais qui permette d'étudier les dynamiques de réduction et de normalisation. Une idée qui paraît acceptable, et souvent acceptée, est que les réseaux de preuve sont une bonne approximation d'un tel système. L'objet de notre étude n'étant pas de discuter ce point, on se contentera de renvoyer le lecteur curieux à une étude que nous avons produite dans un rapport dédié à cette question [Cho16] ainsi qu'à un article de Strassburger, justement intitulé *Proof Nets and the identity of proofs* [Str06].

Les réseaux de preuve identifient deux preuves du calcul des séquents qui diffèrent par l'ordre d'introduction des connecteurs, mais disposent d'une notion de coupure et d'élimination des coupures. Notre étude de la dynamique des démonstrations de la Logique Linéaire se fera donc dans ce cadre. Avant de définir les réseaux plus précisément, on peut les décrire comme des objets

graphiques composés de *nœuds*, correspondant aux règles d'inférence, et de *fils*, correspondant aux conclusions de ces inférences. Les fils peuvent être étiquetés par les formules de la Logique Linéaire. Nous donnons en figure 2.3 l'exemple d'un réseau de preuve, qui représente les trois dérivations de séquents de la figure 2.1. On y constate que les conclusions et l'arbre de construction des formules respectés, mais que l'on n'y voit aucunement apparaître une notion de *dernière règle appliquée*, notion au cœur du calcul des séquents.

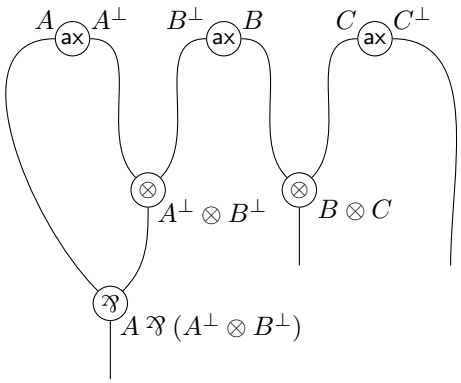


FIGURE 2.3 – Un réseau de preuve qui identifie les dérivations de la figure 2.1.

2.1.1.2 Grandeur des réseaux (2) : à propos du calcul

On admet donc que les réseaux permettent de raisonner sur les preuves de façon au moins satisfaisante. Mais l'une des propriétés qui rendent fécondes ces raisonnements concerne leur aspect calculatoire. Les réseaux en effet fournissent un modèle de calcul d'une souplesse particulièrement intéressante.

Le fragment multiplicatif et exponentiel permet notamment d'encoder le λ-calcul pur [2] dans son intégralité, si l'on munit le langage du type récursif $o = o \multimap o$ (qui rend possible l'application d'un terme à lui même, et par exemple de typer à la Church le terme suivant : $(M^{o \multimap o} M^o)^o$, terme *incestueux* [3] qui ne peut être typé par les types simples). Nous pouvons alors considérer que les propriétés calculatoires établies pour les réseaux sont également établies pour le λ-calcul (si la propriété, bien sûr, est assez générale pour ne pas être spécifique au modèle de calcul). Regnier, dans sa thèse [Reg92], développe cette correspondance. On peut aussi citer les réseaux d'interactions, une variante des réseaux de preuve se concentrant justement sur la dynamique de réécriture, oubliant en quelque sorte l'aspect logique et démonstratif des objets. Lafont, qui les a introduits, a montré que l'on pouvait encoder le λ-calcul et les machines de Turing dans des réseaux d'interaction contenant seulement trois types de nœuds [Laf90, Laf94, Laf97].

L'un des domaines d'application les plus fertiles autour de ces liens entre réseaux et calcul est la complexité algorithmique. En effet, la structure des réseaux

2. cet encodage est illustré au chapitre suivant aux figures 3.1 et 3.2.

3. « le typage est une forme de surmoi interdisant certaines formes d'inceste logique du style $x \in x$. » Girard, *Le Point Aveugle* [Gir06]

de preuve permet d'isoler de façon particulièrement fine des comportements spécifiques, comme la duplication ou suppression des arguments, la profondeur des appels récursifs. On parle de Logiques allégées, et l'on peut citer la Logique Linéaire Élémentaire, qui caractérise les calculs dont la complexité en temps est bornée par une fonction élémentaire, on renvoie aux travaux de Mazza [Maz06] ou à ceux de Baillot et Mazza [BM10] pour des études de ces phénomènes.

On peut attribuer ce succès des réseaux à deux aspects principaux : le contrôle de la duplication et la localité de la réduction. Le premier tient des propriétés de la logique linéaire en général dont héritent naturellement les réseaux. Le second est plus spécifique, il tient au fait que la dynamique associée aux réseaux, qui correspond à l'élimination des coupures dans les preuves de la Logique Linéaire, peut être isolée en des réécritures apparaissant à n'importe quel endroit du réseau. On peut étudier les propriétés de la réduction localement.

Prenons l'exemple de l'élimination des coupures multiplicatives. Elle s'écrit en calcul des séquents comme la règle illustrée en figure 2.4.

$$\dfrac{\dfrac{\dfrac{[\pi_1]}{\vdash \Gamma_1, A, B}}{\vdash \Gamma_1, A \parr B} \qquad \dfrac{\dfrac{[\pi_2]}{\vdash \Gamma_2, A^\perp} \qquad \dfrac{[\pi_3]}{\vdash \Gamma_3, B^\perp}}{\vdash \Gamma_2, \Gamma_3, A^\perp \otimes B^\perp}}{\vdash \Gamma_1, \Gamma_2, \Gamma_3}\ \text{coupure}$$

$$\rightarrow \quad \dfrac{\dfrac{\dfrac{[\pi_1]}{\vdash \Gamma_1, A, B} \qquad \dfrac{[\pi_2]}{\vdash \Gamma_2, A^\perp}}{\vdash \Gamma_1, \Gamma_2, B^\perp}\ \text{coupure} \qquad \dfrac{[\pi_3]}{\vdash \Gamma_3, B^\perp}}{\vdash \Gamma_1, \Gamma_2, \Gamma_3}\ \text{coupure}$$

FIGURE 2.4 – Élimination d'une coupure multiplicative en calcul des séquents.

On remarque qu'un choix a été fait ici, en coupant dans la preuve réduite π_1 avec π_2. On aurait pu couper π_1 avec π_3 et obtenir un réduit similaire, mais qui diffère dans l'ordre d'introduction des connecteurs, ce dont nous avons déjà discuté. On note également que la règle de réduction est donnée *en contexte*, c'est-à-dire qu'il serait imprécis et incomplet de la donner sans expliciter ce qu'il advient des Γ_i et des π_i lors de la réduction. On donne l'équivalent de cette règle dans les réseaux de preuve en figure 2.5.

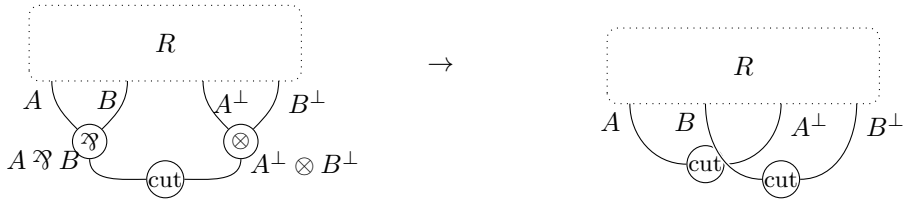


FIGURE 2.5 – Élimination d'une coupure multiplicative dans les réseaux

On peut à nouveau constater que l'ordre d'introduction des connecteurs est ici oublié. Et surtout, on remarque que la réduction ne concerne que les formules

coupées, et que la règle ne fait pas intervenir la structure R, qui correspond aux contextes Γ_i et preuves π_i de la figure 2.4. Cela illustre ce que l'on veut dire quand on prétend que la réduction dans les réseaux peut être faite de façon locale. La règle présentée en figure 2.5 est bien valide pour toute structure R, et on peut considérer les branchements des coupures et des connecteurs principaux comme le seul phénomène observable lors de la réduction. Il n'y a pas d'autre *déplacement*, comme en calcul des séquents.

2.1.1.3 Misères des réseaux

Il n'est pas, ou peu, de système satisfaisant à plusieurs égards qui ne vienne sans son lot d'imperfections. Nous avons décrit plusieurs qualités importantes des réseaux de preuves plus haut, mais il nous faut maintenant, par souci d'honnêteté, nuancer en présentant les difficultés qui se manifestent à plusieurs endroits.

Dans un premier temps, notons qu'une des lacunes les plus remarquables des réseaux de preuves est qu'ils n'englobent qu'une partie de la Logique Linéaire. En effet, le fragment additif ne se prête pas docilement à une présentation graphique. Cela est possible, mais demande des ajustements et des extensions du système non négligeables (voir par exemple les travaux de Tortora de Falco [Tor03], ou de Heijltjes, Hugues et Strassburger [HHS18]).

De plus, nous avons illustré les bonnes propriétés à travers des exemples pris dans le fragment multiplicatif exclusivement, **MLL**. Or, le fragment dans lequel il est possible d'étudier des propriétés calculatoires intéressantes, et dans lequel le λ-calcul peut être encodé, est plus large, il s'agit du fragment multiplicatif exponentiel **MELL**. Et il est faux que **MELL** jouit de toutes les qualités de **MLL**. En particulier, l'introduction des connecteurs exponentiels rend sensibles les réseaux à la notion de « dernière règle appliquée ». Cela se traduit par la notion de *boîte exponentielle*, dont l'étude sera par ailleurs au cœur de ce chapitre.

Le connecteur !, qui exprime la duplicabilité d'une sous-preuve en Logique Linéaire (voir sections 1.1.4 et 1.3.1), demande en effet une règle d'introduction sensible au contexte, appelée règle de *promotion* :

$$\frac{\vdash ?A_1, \ldots, ?A_n, A}{\vdash ?A_1, \ldots, ?A_n, !A}\ \text{Promotion}$$

Et cette règle ne s'applique que si toutes les formules du contexte ont pour connecteur principal ?. Ce qui demande un traitement particulier pour sa traduction dans les réseaux de preuve, que nous illustrons en figure 2.6, où $\pi^\star$ représente la traduction en réseaux de la dérivation du séquent $\vdash ?A_1, \ldots, ?A_n, A$.

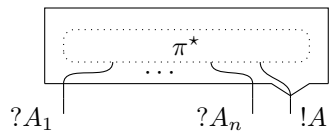


FIGURE 2.6 – Traduction de la règle de promotion dans les réseaux.

La boîte qui est tracée autour de la structure $\pi^\star$ représente le fait que l'on se *rappelle* que le contexte était adéquat (les formules y sont de la forme $?A_i$) lorsqu'on a appliqué la règle de promotion.

Ces boîtes affaiblissent également l'aspect local de l'élimination des coupures. En effet, la règle de réduction promotion/contraction demande par exemple, comme en calcul des séquents, la duplication et le déplacement de tout le contenu de la boîte. Donc sont concernées par l'élimination de cette coupure des parties de la structure qui ne sont pas reliées à la formule coupée, mais à son contexte. Cette règle de réduction est illustrée en figure 2.7. On rappelle que le calcul des séquents pour la Logique Linéaire est donné en introduction (voir la figure 2.2).

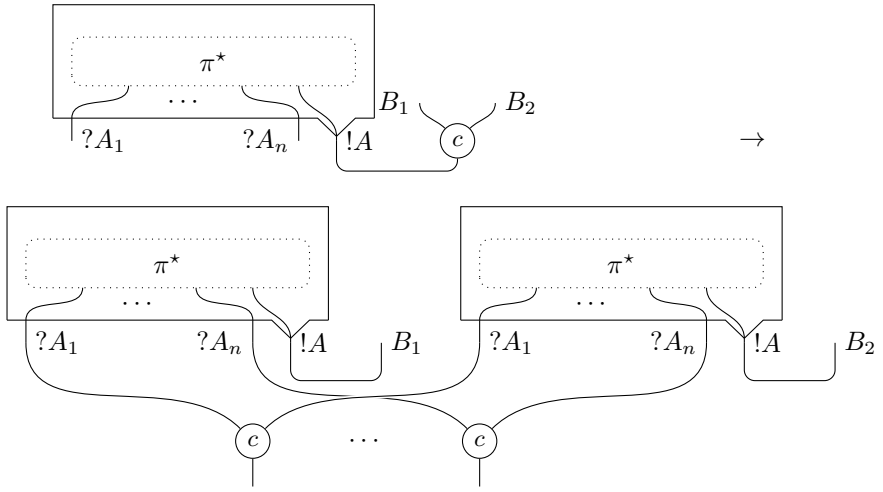


FIGURE 2.7 – Exemple d'élimination d'une coupure dupliquant une boîte.

Ce phénomène, qui correspond dans le λ-calcul à la duplication des arguments, demande donc un soin particulier, et sera traité en détail dans ce chapitre (voir section 2.4).

Une autre difficulté à laquelle on fait face est la gestion des réseaux comportant des composantes non connexes. En particulier, la traduction en réseaux de la règle d'affaiblissement (ou de la règle de co-unité multiplicative) est problématique. L'aspect local des constructions est ici aussi embarrassant. Une des raisons à cela s'exprime en un exemple simple : la structure composée d'un affaiblissement seul (**a**) n'est pas un réseau correct, en ce qu'il ne correspond à aucune preuve de la Logique Linéaire. Le lecteur pourra s'en convaincre en se rapportant aux règles de dérivation en figure 2.2. En revanche, si R est un réseau correct, alors le réseau R (**a**) est, lui, correct. Cela donne un statut particulier aux affaiblissements, dont l'introduction est acceptable en fonction de l'existence ou non d'un contexte. Une solution souvent adoptée pour traiter ce problème consiste, à l'instar des boîtes exponentielles, à introduire de la séquentialité et de la contextualité à ces connecteurs (**a** et $\bot$).

Cela se fait à travers de la notion de *sauts*, qui sont une façon de rattacher systématiquement tout affaiblissement à une structure correcte. Ainsi, la structure donnée en exemple ci-dessus sera enrichie d'une fonction de saut s et illustrée comme suit : R $\overset{s}{\cdots}$ (**a**)

Cette fonction peut envoyer l'affaiblissement sur n'importe quel fil de R, elle n'a pas de sens logique relatif au lien qu'elle exprime. Elle permet à nouveau de *se rappeler* que l'affaiblissement à été introduit de façon légale, car elle assure qu'une structure lui préexistait.

Travailler avec des structures à sauts de la sorte présente d'autres intérêts concernant la géométrie des réseaux, et qui seront cruciaux pour notre étude. C'est l'objet de la section 2.3.3.

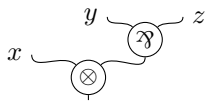
2.1.1.4 Notre choix de syntaxe pour les structures

Avant de parler du développement de Taylor, qui est une clef de voûte de nos travaux, nous présentons la façon dont nous formalisons les réseaux de preuve. Il est de notoriété publique qu'il n'y a pas de syntaxe canonique pour les réseaux qui soit satisfaisante à tous les égards. Grand nombre d'articles introduisent leur propre présentation, adéquate pour des études particulières. Tout choix de la sorte est contestable, et généralement contesté.

Le nôtre est dicté par une souplesse à l'endroit des manipulations que nous souhaitons pratiquer et des raisonnements que nous voulons écrire. Si notre syntaxe facilite l'écriture des réseaux, nous tâcherons d'illustrer nos propos autant que possible avec des dessins explicatifs pour en rendre accessible la lecture.

Nous utilisons une variante d'une syntaxe de termes introduite par Ehrhard [Ehr14], et inspirée par Mackie [Mac94]. Ehrhard propose pour **MLL** une syntaxe construite sur des termes représentants l'arbre syntaxique de construction des formules, avec une construction explicite pour les coupures, et une dualité entre les feuilles de ces arbres pour les axiomes. Nous avons proposé une extension de cette syntaxe pour **MELL** [Cho15] en étendant le critère de correction associé (critère basé sur une étude de cographes colorés par Rétoré [Ret03], puis présenté à travers les espaces de cohérence par Ehrhard [Ehr14]). Notre gestion des boîtes exponentielles diffère de celle de Ehrhard dans un travail ultérieur [Ehr16b], en ce que nous les traiterons comme des axiomes généralisés avec contextes de boîte. De plus, nos structures ne sont pas typées. Elles peuvent l'être, mais cela n'apporte rien à nos résultats principaux. Nous traitons donc des réseaux sans nous occuper des formules qui pourraient en étiqueter les fils.

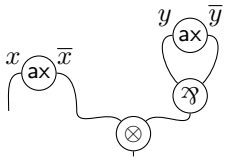
Pour **MLL**, nous considérons donc des constructeurs binaires $\otimes$ et $⅋$, à partir desquels l'on construit des arbres dont les feuilles sont prises dans un ensemble dénombrable de variables $x, y, \ldots$. Ainsi, par exemple l'arbre $x \otimes (y ⅋ z)$ correspondra à la structure représentée ci-dessous :



Mais cette structure, qui est en fait une *préstructure*, comme on le verra au moment de définir formellement ces objets, ne correspond pas à un réseau de preuve, dans la mesure où son sommet n'est pas composé d'axiomes. On dira que la structure n'est pas *close*.

On munit donc l'ensemble des variables d'une involution $\overline{()}$ qui reliera ces dernières par deux, de façon à ce qu'une structure ne soit acceptable que si pour toute variable x y apparaissant, $\overline{x}$ y apparaît aussi. Au cas contraire, on aurait un objet dans lequel les axiomes pourraient ne comporter qu'une conclusion, ce qui n'est pas souhaitable.

Les arbres ainsi construits vont former les conclusions du réseau. Et une structure sera définie comme une famille d'arbres dans laquelle aucune variable n'apparaît si sa duale n'y apparaît pas. Par exemple, la suite $(x, \overline{x} \otimes (y \parr \overline{y}))$ est une structure close, représentée ci-dessous :



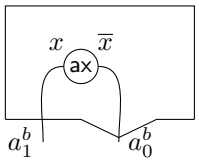
Cette définition suffit à caractériser les réseaux sans coupures. Les coupures sont définies comme des paires d'arbres notées $\langle t_1 | t_2 \rangle$. Nous renvoyons à la section 2.2 pour la définition complète et des exemples plus exhaustifs.

La gestion des boîtes exponentielles, pour l'extension à **MELL**, se fera par une définition par induction sur la profondeur des boîtes, et par l'enrichissement de la syntaxe par des constantes correspondant aux noms des portes de la boîte.

L'idée est la suivante : on considère les boîtes comme des axiomes généralisés, et les feuilles des arbres de nos structures (que nous appellerons *atomes*) sont soit des variables, conclusions d'axiomes, soit des constantes, conclusions de boîtes. Ces constantes sont étiquetées par des noms b, correspondant aux identifiants des boîtes en question, et les structures sont munies d'un *contexte de boîte*. Ces contextes sont des fonctions qui à un nom de boîte b associent le contenu de cette boîte.

Le contenu d'une boîte étant lui-même une structure de preuve, la définition des contextes est par induction mutuelle avec celle des structures. Prenons l'exemple d'un axiome, présenté dans notre syntaxe comme une suite $(x, \overline{x})$, et d'une boîte correspondant à la promotion de cet axiome. La structure ainsi obtenue s'écrira $(\Theta, (a_1^b, a_0^b))$ (l'indice 0 distinguant la porte principale de la boîte, est placé en dernier par convention). Θ est le contexte de boîte, à savoir la fonction qui à b associe la structure $(\Theta', (x, \overline{x}))$. Θ' est simplement un contexte vide, car il n'y a plus de boîte à la profondeur courante.

La structure décrite de cette façon peut s'illustrer ainsi ci-dessous. On renvoie à la section 2.4 pour les constructions détaillées relatives à **MELL**.



Concluons cette présentation de notre syntaxe en déclarant que si aucune ne semble faire consensus, celle-là paraît adaptée pour nos raisonnements sur la taille des structures, et les chemins dans ces structures. Ce qui graphiquement correspond à un *fil* dans un réseau ne sera rien d'autre qu'un *sous-arbre* dans notre syntaxe.

2.1.2 Le développement de Taylor

2.1.2.1 Histoire et enjeux

Au cœur des sémantiques quantitatives qui cherchent à rendre compte de la consommation de ressources nécessaires au calcul, à la normalisation, aux *quantités* à l'œuvre en général dans la dynamique de la réduction, se trouve le développement de Taylor. D'abord introduite par Ehrhard et Regnier pour le λ-calcul [ER03], cette construction a été adaptée par les mêmes auteurs aux réseaux de preuve [ER05].

On a déjà vu que l'une des propriétés remarquables de la Logique Linéaire était d'isoler la *duplicabilité* d'une ressource en la caractérisant par un connecteur dédié, à savoir !. Nous venons de voir par ailleurs que l'introduction d'une formule de la forme $!A$ se faisait en calcul des séquents par la règle de promotion, et dans les réseaux par la construction des boîtes exponentielles.

Le développement de Taylor est une interface entre la syntaxe d'un système de preuve ou d'un modèle de calcul, et sa sémantique, qui rend compte de la duplication et consommation de ressources lors de l'exécution du calcul ou de l'élimination de coupures correspondant. Ce qui sous-tend sa définition est la possibilité de donner des approximations linéaires (en général multilinéaires) d'une preuve ou d'un terme.

En ce qui concerne les réseaux, les composantes sur lesquelles sera définie cette approximation sont les boîtes exponentielles (qui comme leur nom l'indique sont responsables de l'aspect non linéaire de l'élimination des coupures). Nous avons déjà vu en figure 2.7 un exemple de duplication d'une boîte. Dans ce réseau, la boîte est dupliquée, mais les deux boîtes issues de cette duplication peuvent elles-mêmes être sujettes à une duplication ultérieure, que nous ne pouvons pas contrôler depuis cette vue partielle de la structure.

Une approximation n-linéaire d'une boîte consiste en un réseau, sans boîte, dans lequel le contenu de ladite boîte apparaît n fois. Au lieu de disposer d'un objet infiniment duplicable, on dispose d'un objet déjà dupliqué, qui peut être consommé un nombre fini de fois.

Les approximations des réseaux de **MELL** sont définies dans des réseaux ne comportant pas de boîte, mais permettant néanmoins de démontrer des formules de la forme $?A$. **MLL** est donc insuffisant. Le fragment dans lequel le développement de Taylor correspond aux réseaux différentiels sans boîte, que l'on appellera $\mathbf{DLL}_0$, ou encore *réseaux à ressources*, par analogie avec le λ-calcul à ressources. Le développement de Taylor pour le λ-calcul usuel est défini dans ce dernier, qui correspond au fragment du λ-calcul différentiel dans lequel toutes les applications sont multilinéaires.

En plus des connecteurs de **MELL**, $\mathbf{DLL}_0$ comporte des opérateurs différentiels qui sont la cocontraction, $\overline{c}$, le coaffaiblissement, $\overline{\mathbf{a}}$, et la codéréliction, $\overline{d}$. Les réseaux définis dans cette nouvelle syntaxe ne permettent pas la duplication ni la suppression de composantes du réseau, mais permettent de simuler le comportement des boîtes. Les réseaux différentiels complets, avec les boîtes, sont étudiés à part pour leurs propriétés sémantiques et calculatoires, mais ne sont pas l'objet de nos travaux car ils n'ont pas la propriété d'être munis de réductions exclusivement linéaires (à cause de la présence des boîtes justement). Pour autant, les réseaux à ressources sont inclus dans les réseaux différentiels, et en cela héritent de tous les résultats généraux concernant ces derniers. On peut

citer par exemple le résultats de confluence de Tranquilli [Tra09], et le théorème de conservation de Pagani et Tranquilli [PT17] (bien que la confluence pour les réseaux à ressources fusse établie isolément par Ehrhard et Regnier).

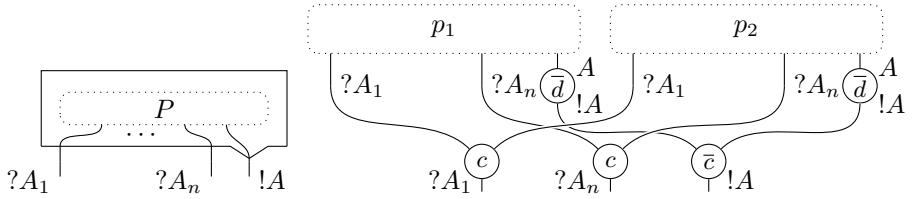


FIGURE 2.8 – Une boîte exponentielle et son approximation 2-linéaire.

Nous illustrons en figure 2.8 une boîte exponentielle et son approximation 2-linéaire. C'est-à-dire le réseau à ressources comportant deux copies du contenu de la boîte. p_1 et p_2 doivent être définis comme des approximations 2-linéaires de P, de façon à ce que ce mécanisme d'approximation se propage à toutes les profondeurs. On remarquera que le typage est préservé : P est un réseau de conclusions $?A_1, \ldots, ?A_n, A$, et l'exponentielle $!A$ est issue de la boîte. Le type du réseau à ressources est le même, mais obtenu par les connecteurs différentiels.

Le développement de Taylor consiste alors à partir d'un réseau de **MELL**, et de considérer (inductivement sur la profondeur) pour chaque boîte une approximation n-linéaire pour un $n \in \mathbf{N}$ arbitrairement choisi pour cette boîte (le nombre de copies n'est pas uniforme sur l'ensemble des boîtes). On capture donc le comportement d'un réseau avec boîtes exponentielles en prenant une infinité d'approximations multilinéaires. Le développement de Taylor est un outil puissant pour la sémantique de la Logique Linéaire. Il permet par exemple d'établir des résultats d'injectivité du modèle relationnel, comme le fait de Carvalho [dC15], et après lui Guerrieri, Pellissier et Tortora de Falco [GPT16]. Les mêmes auteurs ont également proposé une élégante présentation du développement de Taylor à partir de produits fibrés dans les catégories, en partant d'une syntaxe pour les réseaux basée sur la théorie des graphes [GPT19].

De plus amples détails ainsi que les définitions générales seront donnés en section 2.5.

2.1.2.2 Le problème de la convergence

La difficulté qui se pose concerne la dynamique que nous voulons construire dans le développement de Taylor. En effet, nous traitons avec des ensembles infinis de réseaux à ressources, qui correspondent à des points de la sémantique relationnelle, mais dont on veut étudier les mécanismes de réduction et de normalisation.

Nous voulons rendre concrète l'intuition selon laquelle une infinité d'approximations d'un réseau, prise dans sa totalité, se comporte de la même façon que le réseau approché. Pour cela, le résultat vers lequel nous tendons est le suivant : si un réseau P de **MELL** se réduit en un réseau Q, alors le développement de Taylor de P se réduit dans le développement de Taylor de Q. Cette propriété est assez bien comprise quand elle renvoie à la sémantique, et elle se vérifie dans certains modèles quantitatifs. En revanche, elle n'est pas aussi claire dans la syn-

taxe. En particulier parce qu'elle demande de définir une notion de réduction entre des séries infinies de structures.

Cela est loin d'aller de soi. En effet, considérons une série $\sum_{i\in I} a_i p_i$ de réseaux à ressources p_i, prenant leurs coefficients a_i dans un semi anneau $\mathbf{S}$. Nous considérons un semi anneau (avec fractions, voir section 1.4) quelconque de façon à rester assez général pour que nos résultats restent valables dans la plus grande variété possible de modèles.

Pour montrer qu'une somme de réseaux comme celle présentée ci-dessus revêt un certain comportement dynamique, il faut s'assurer que définir une réduction qui agit sur une telle somme est possible, ne pose pas de contre-indication algébrique.

Avant de comprendre le problème relatif aux coefficients, il nous faut comprendre pourquoi l'on ne peut se contenter d'une réduction simple sur les réseaux à ressources et l'on doit introduire une réduction parallèle. Dans la mesure ou nous souhaitons montrer dans un premier temps qu'une approximation d'un réseau de **MELL** peut simuler sa dynamique, nous devons considérer toutes les réductions de **MELL** possibles et en particulier les réductions internes.

Revenons à la figure 2.8, et imaginons qu'il existe une réduction interne à la boîte, de façon à ce qu'on ait $P \to P'$. On comprend bien dans ces conditions que le réseau à ressources qui était une approximation de P n'en est pas un de P'. Pour simuler la réduction, il faut que les deux copies de la boîte soient soumises à une étape de réduction, de façon à ce que l'on ait deux copies d'une approximation de P' après réduction. On définit donc une réduction notée $\rightrightarrows$ qui autorise la réduction d'un nombre arbitraire de coupures apparaissant dans un réseau à ressources (les nouvelles coupures, engendrées par la réduction ne peuvent cependant pas être réduites dans la même étape).

Donnons maintenant l'exemple d'une série de réseaux $\phi = \sum_{n\in\mathbf{N}} p_n$ où p_n est défini comme une suite de n axiomes reliés deux-à-deux par une coupure, illustré en figure 2.9.

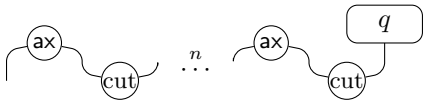


FIGURE 2.9 – Un exemple de de réseau p_n, se réduisant en parallèle en un même réseau q pour tout $n \in \mathbf{N}$

On constate que pour tout n, $p_n \rightrightarrows q$: la réduction parallèle autorise l'élimination simultanée de toutes les coupures d'axiomes présentes dans p_n. Nous voulons maintenant étendre la réduction parallèle aux sommes infinies de réseaux, de façon à pouvoir considérer le développement de Taylor dans son ensemble et en étudier le comportement

Si l'on considère que dans la combinaison ϕ, chaque p_n a un coefficient 1, et que la réduction parallèle peut être appliquée à chaque argument de la somme, en une unique étape, notée $\Rightarrow$, alors on peut écrire $\phi \Rightarrow \infty \cdot q$. En effet, une infinité de réseaux dans ϕ se réduisent en q, et le coefficient obtenu est alors bien $\sum_{n\in\mathbf{N}} 1$, qui n'est pas un coefficient défini dans n'importe quel semi-anneau.

Si le semi anneau $\mathbf{S}$ n'est pas complet (ce peut être le cas, comme pour le modèle relationnel pondéré de Laird, Manzonetto, McCusker et Pagani [LMMP13],

mais ça ne l'est pas dans les espaces de finitude de Ehrhard [Ehr05], par exemple), toutes les sommes ne convergent pas, et en particulier le coefficient donné ci-dessus n'apparaît pas dans **S**.

La démarche que nous nous emploierons alors à suivre consiste à montrer que le développement de Taylor ne donne pas lieu à des sommes arbitraires de réseaux à ressources. Ses éléments jouissent de certaines propriétés structurelles et géométriques qui empêchent les coefficients infinis d'apparaître pendant la réduction parallèle appliquée à toute la série.

Ce problème, ainsi que diverses méthodes de résolution, a déjà été présenté pour différents λ-calculs. D'abord par Ehrhard et Regnier pour le λ-calcul pur [ER08], puis pour des calculs avec choix non déterministes, par Ehrhard [Ehr10] puis par Pagani, Tasson et Vaux Auclair [PTV16], et plus tard pour le λ-calcul algébrique [Vau09] par Vaux Auclair [Vau17]. La méthode que nous développerons est inspirée de ce dernier. Elle consiste à remplacer la notion de profondeur structurelle des λ-termes par une mesure relative aux chemins d'interrupteurs (dans le sens de Danos-Regnier) dans les réseaux.

L'essentiel de notre approche, qui suit celle de Vaux Auclair, consiste alors à contrôler la perte de taille des réseaux sous réduction parallèle en utilisant des mesures sur les chemins. La propriété de convergence sera alors conséquence du fait qu'il n'existe qu'un nombre fini de structures n'excédant pas une taille donnée.

On peut, à titre d'exemple, revenir à la figure 2.9 pour se convaincre que la perte de taille par réduction parallèle entre chaque p_n et q n'est pas bornée. Le fait qu'il existe dans chaque p_n un chemin rencontrant un nombre non borné (car supérieur à n) de coupures sera le point clef de notre approche, pour exclure des séries dégénérées de structures comme celle-là.

2.1.3 Contenu du chapitre

Nous commençons ce chapitre en traitant le fragment multiplicatif connexe des réseaux de la Logique Linéaire (section 2.2), c'est-à-dire le fragment épuré, dans lequel tout se passe bien. Nous établissons une série de résultats concernant la réduction parallèle dans ces structures, concernant les chemins d'interrupteurs et l'évolution de la taille des structures pendant l'élimination des coupures.

Cela permet d'obtenir en particulier un résultat original sur **MLL**, à savoir une version quantitative de la préservation de l'acyclicité des structures : il est montré que lorsque l'on a une réduction parallèle entre deux réseaux, la longueur des chemins augmente de l'ordre de $2n!$. De plus, et c'est la motivation première, les réseaux à ressources, qui sont introduits en section 2.3, présentent des propriétés combinatoires très proches de celles des structures multiplicatives, et on peut alors adapter les résultats concernant **MLL** à $\mathbf{DLL}_0$.

Ceci fait, nous introduisons les structures exponentielles et leur dynamique associée en section 2.4. Nous donnons ensuite la construction du développement de Taylor en section 2.5, en y appliquant les résultats obtenus sur les réseaux à ressources. Enfin, nous rassemblons toutes les constructions et tous les résultats du chapitre pour parvenir au résultat principal, à savoir la simulation de l'élimination des coupures de **MELL** dans le développement de Taylor.

2.2 Réseaux multiplicatifs sans unités

2.2.1 Constructions

On commence donc par présenter les réseaux de preuve pour la logique linéaire multiplicative sans unités, notés $\mathbf{MLL}^-$. Ce choix est orienté par plusieurs propriétés que revêt ce système, didactiques et structurelles. Didactiques pour sa légèreté et la possibilité de donner une présentation claire des réseaux avant de considérer des fragments plus problématiques de la Logique Linéaire. Structurelles car toutes les bonnes propriétés des réseaux (localité de l'élimination des coupures, quotient sur les règles de commutation du calcul des séquents) sont présentes dans $\mathbf{MLL}^-$, et sont altérées dès que l'on considère des structures plus riches, munies d'unités ou d'exponentielles (voir sections 2.1.1.1 et 2.1.1.2).

Définition 1 (Structures de $\mathbf{MLL}^-$)**.**
On considère un ensemble dénombrable de variables $\mathcal{V}$ notées $x, y, x_1, \ldots$ muni d'une involution $\overline{()} : \mathcal{V} \to \mathcal{V}$. Les *pré-arbres de* $\mathbf{MLL}^-$ sont les termes construits sur la syntaxe suivante :

$$t, s ::= x \mid t \otimes s \mid t \parr s$$

Pour tout pré-arbre t de $\mathbf{MLL}^-$, on définit inductivement l'ensemble $\mathbf{SA}(t)$ de ses *sous-arbres* : pour tout $x \in \mathcal{V}$, $\mathbf{SA}(x) = \{x\}$. Pour tout $t = s\alpha s'$, si $\alpha \in \{\otimes, \parr\}$, $\mathbf{SA}(t) = \mathbf{SA}(s) \cup \mathbf{SA}(s') \cup \{t\}$.

Un *arbre de* $\mathbf{MLL}^-$ est soit une variable de $\mathcal{V}$, soit un terme $s\alpha s'$, où s et s' sont des arbres, tels que $\alpha \in \{\otimes, \parr\}$ et tel que $\mathbf{SA}(s) \cap \mathbf{SA}(s') = \emptyset$.

Une *coupure de* $\mathbf{MLL}^-$ est une paire d'arbres notée $\langle t|s\rangle$ telle que $\mathbf{SA}(t) \cap \mathbf{SA}(s) = \emptyset$. Les sous-arbres d'une coupure sont définis comme suit : $\mathbf{SA}(\langle t|s\rangle) = \mathbf{SA}(t) \cup \mathbf{SA}(s)$.

Une *pré-structure de* $\mathbf{MLL}^-$ est la donnée d'une famille de coupures $c_1, \ldots, c_n$ et d'une famille d'arbres $t_1, \ldots, t_m$ telles que pour tous $i, j \in \{1, \ldots, n\}$, $\mathbf{SA}(c_i) \cap \mathbf{SA}(c_j) = \emptyset$ dès que $i \neq j$, pour tous $k, l \in \{1, \ldots, m\}$, $\mathbf{SA}(t_k) \cap \mathbf{SA}(t_l) = \emptyset$ dès que $k \neq l$, et telles que $\mathbf{SA}(c_i) \cap \mathbf{SA}(t_l) = \emptyset$ pour tous $i \in \{1, \ldots, n\}$ et $l \in \{1, \ldots, m\}$. C'est-à-dire que tout arbre de $\mathbf{MLL}^-$ à au plus une occurrence dans une préstructure en tant que sous-arbre. Les préstructures sont alors notées $p = (c_1, \ldots, c_n; t_1, \ldots, t_m)$. On note $\mathbf{C}(p)$ l'ensemble des coupures de p. On définit l'ensemble des sous-arbres de p comme suit : $\mathbf{SA}(p) = \bigcup_{i\in\{1,\ldots,n\}} \mathbf{SA}(c_i) \cup \bigcup_{j\in\{1,\ldots,m\}} \mathbf{SA}(t_j)$.

Enfin, une *structure de* $\mathbf{MLL}^-$ est une pré-structure p telle que pour toute variable $x \in \mathcal{V}$, si $x \in \mathbf{SA}(p)$ alors $\overline{x} \in \mathbf{SA}(p)$.

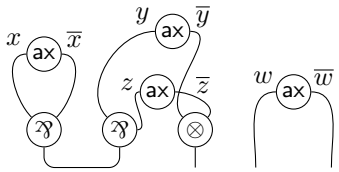
L'égalité entre structures correspond à l'α-équivalence, c'est-à-dire au renommage des paires de variables duales $x, \overline{x}$. Nous travaillerons comme c'est l'usage sur des représentants de ces classes d'équivalence.

Une première observation que l'on peut faire est que les structures définies ci-dessus sont plus générales que les présentations usuelles de $\mathbf{MLL}$: en l'absence de typage, les coupures ne sont pas nécessairement réductibles. En effet, on peut dire qu'une coupure $c = \langle t_1|t_2\rangle$ est réductible si $t_i \in \mathcal{V}$ pour un $i \in \{1, 2\}$, ou si $c = \langle t \otimes t'|s \parr s'\rangle$. Mais ce critère de réductibilité ne serait pas préservé par réduction. Dans nos structures, rien n'empêche deux arbres ayant un tenseur comme connecteur le plus extérieur d'être coupés entre eux. Pour autant,

notre étude géométrique de la réduction n'en sera pas affectée, car nous nous intéresserons aux mécanismes qui modifient la structure pendant la réduction : les coupures non réductibles n'interviendront alors pas dans nos arguments.

Remarque 1. On peut vérifier immédiatement que la définition des structures implique que pour toute variable $x \in \mathcal{V}$, x a au plus une occurrence dans une structure.

Exemple 1. Ci-dessous, la structure p respecte bien les critères de la définition 1. $p = (\langle x \parr \overline{x} | y \parr z \rangle; \overline{y} \otimes \overline{z}, w, \overline{w})$. Elle correspond à la représentation graphique suivante, où l'on indique à quelle conclusion d'un lien axiome correspond une variable, et où les coupures ne sont pas explicitées par des noeuds mais par la jonction de la racine des arbres correspondant :



Définition 2 (Substitutions et élimination des coupures dans $\mathbf{MLL}^-$)**.**
La *substitution* $[t/x]$ *d'un arbre* t *à une variable* x est d'abord définie sur les arbres : $x[t/x] = t$, $y[t/x] = y$ si $y \neq x$. $(s \alpha s')[t/x] = s[t/x] \alpha s'[t/x]$ pour $\alpha \in \{\parr, \otimes\}$, puis étendue de façon naturelle aux préstructures :

$$(\langle t_1 | t'_1 \rangle, \ldots, \langle t_n | t'_n \rangle; s_1, \ldots, s_m)[t/x] =$$
$$(\langle t_1[t/x] | t'_1[t/x] \rangle, \ldots, \langle t_n[t/x] | t'_n[t/x] \rangle; s_1[t/x], \ldots, s_m[t/x])$$

On définit maintenant les relations de réductions $\to_{\mathsf{ax}}$ et $\to_m$ correspondant à l'élimination des coupures d'axiomes et es coupures multiplicatives :

— $(\overrightarrow{c}, \langle x | t \rangle; \overrightarrow{s}) \to_{\mathsf{ax}} (\overrightarrow{c}; \overrightarrow{s})[t/\overline{x}]$ si $\overline{x} \notin \mathbf{SA}(t)$

— $(\overrightarrow{c}, \langle t \otimes t' | s \parr s' \rangle; \overrightarrow{u}) \to_m (\overrightarrow{c}, \langle t | s \rangle, \langle t' | s' \rangle; \overrightarrow{u})$

On écrira parfois simplement $\langle t \otimes t' | s \parr s' \rangle \to_m \langle t | s \rangle, \langle t' | s' \rangle$ quand seul le comportement local de la réduction nous intéressera.

Définition 3 (Interrupteurs, adjacence et chemins dans $\mathbf{MLL}^-$)**.**
Soit p une structure de $\mathbf{MLL}^-$. Une *Position d'interrupteurs de* p I_p est une fonction partielle de domaine $\mathbf{SA}(p)$ et de codomaine $\mathbf{SA}(p)$, telle que pour tout $t \in \mathbf{SA}(p)$, s'il existe s, s' tels que $t = s \parr s'$, alors $I_p(t) \in \{s, s'\}$. Nous écrirons le plus souvent I au lieu de de I_p s'il n'y a pas d'ambiguïté.

La relation d'adjacence dans une structure p est relative à une position d'interrupteurs. Soient $t, s \in \mathbf{SA}(p)$ et I une position d'interrupteurs de p. On dit que t est *adjacent à* s pour la position I, et l'on note $t \sim^{p,I}_{t,s} s$ si :

— $t \in \mathcal{V}$ et $s = \overline{t}$.

— $t = s \parr s'$ ou $t = s' \parr s$, et $I(t) = s$.

— $t = s \otimes s'$ ou $t = s' \otimes s$.

et l'on note $t \sim^{p,I}_{\langle t|s \rangle} s$ si :

— $\langle t | s \rangle \in \mathbf{C}(p)$.

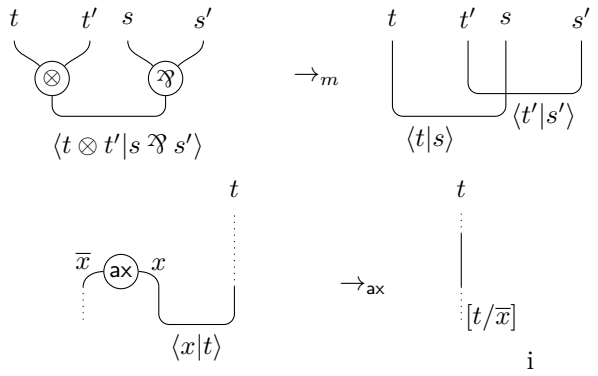


FIGURE 2.10 – Élimination des coupures dans $\mathbf{MLL}^-$

Pour deux étiquettes d'adjacence $l, l' \in (\mathbf{SA}(p) \times \mathbf{SA}(p)) \cup \mathbf{C}(p)$, on écrit $l \equiv l'$ si $l = l'$ ou si $l = (x, \overline{x})$ et $l' = (\overline{x}, x)$.

Un *chemin* dans p pour une position I est une suite de sous-arbres de p $t_1, \ldots, t_n$ telle que :

1. Pour tout $i \in \{1, \ldots, n-1\}$, $t_i \sim^{p,I} t_{i+1}$.
2. Pour tout $i \in \{1, \ldots, n-2\}$, si $t_i \sim_l^{p,I} t_{i+1} \sim_{l'}^{p,I} t_{i+2}$, $l \not\equiv l'$.

L'ensembles des chemins dans p pour toutes les positions d'interrupteurs est noté $\mathbf{Ch}(p)$. Ses éléments seront le plus souvent notés $\chi, \zeta, \xi, \chi_1, \chi_2, \ldots$.

Si $\chi_1 = (t_1, \ldots, t_n), \chi_2 = (s_1, \ldots, s_m)$, alors la notation χ_1, u, χ_2 représentera le chemin $(t_1, \ldots, t_n, u, s_1, \ldots, s_m)$, et χ_1, c, χ_2 représente les deux chemins χ_1, t, s, χ_2 et χ_1, s, t, χ_2, si $c = \langle t|s \rangle$.

Si $\chi = t, \chi', s$, t et s sont appelés les *extrémités* de χ.

Si $\chi = t_1, \ldots, t_k$, alors le chemin à rebours $t_k, \ldots, t_1$ est noté $\overline{\chi}$.

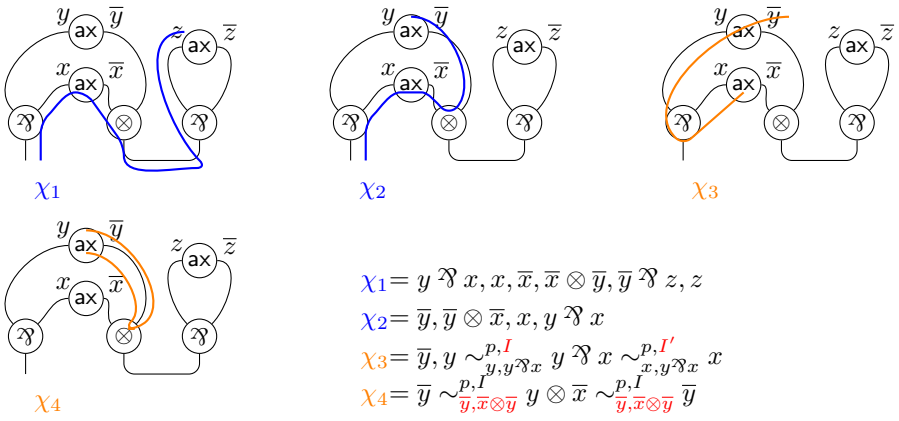


FIGURE 2.11 – Exemples de chemins et d'autres choses

Dans la figure 2.11, nous avons explicité les étiquettes d'adjacence de façon à indiquer en quoi χ_3 et χ_4 ne sont pas des chemins, bien qu'ils soient des

suites d'arbres adjacents. χ_3 emprunte les deux prémisses d'un lien $\parr$, il n'existe donc pas de position d'interrupteur telle que χ_3 soit un chemin pour cette position. Quant à χ_4, il viole la deuxième condition de la définition 3, qui permet justement d'exclure les demi-tours sur un même fil de la structure.

Par la suite, nous omettrons souvent d'indiquer les interrupteurs et/ou les étiquettes d'adjacence, et écrirons $t \sim^p s$, ou encore $t \sim s$, si le contexte ne laisse pas de place à l'ambiguïté.

2.2.2 Élimination des coupures parallèle

De notre objectif à long terme, qui est l'étude du développement de Taylor des réseaux avec exponentielles et unités, nous dérivons un objectif à moyen terme : déterminer pour une réduction parallèle dans $\mathbf{MLL}^-$ certaines propriétés relatives à l'évolution des chemins et à la taille des structures.

Définition 4 (Élimination des coupures parallèle dans $\mathbf{MLL}^-$)**.**
On définit trois réductions : $\rightrightarrows_{\mathsf{ax}}$ la réduction parallèle axiomatique, $\rightrightarrows_m$ la réduction parallèle multiplicative, et $\rightrightarrows$ la réduction parallèle générale, pour toute structure de preuve p de $\mathbf{MLL}^-$.

- $p \rightrightarrows_{\mathsf{ax}} p$, $p \rightrightarrows_m p$ et $p \rightrightarrows p$
- Si $p = (\overrightarrow{c}, \langle x|t\rangle; \overrightarrow{s}) \rightrightarrows_{\mathsf{ax}} p' = (\overrightarrow{c}', \langle x|t\rangle; \overrightarrow{s}')$, alors $p \rightrightarrows_{\mathsf{ax}} (\overrightarrow{c}'; \overrightarrow{s}')[t/\overline{x}]$
- Si $p = (\overrightarrow{c}, \langle x|t\rangle; \overrightarrow{s}) \rightrightarrows p' = (\overrightarrow{c}', \langle x|t\rangle; \overrightarrow{s}')$, alors $p \rightrightarrows (\overrightarrow{c}'; \overrightarrow{s}')[t/\overline{x}]$
- Si $p = (\overrightarrow{c}, \langle t \otimes s|t' \parr s'\rangle; \overrightarrow{u}) \rightrightarrows_m p' = (\overrightarrow{c}', \langle t \otimes s|t' \parr s'\rangle; \overrightarrow{u}')$, alors $p \rightrightarrows_m (\overrightarrow{c}', \langle t|t'\rangle, \langle s|s'\rangle; \overrightarrow{u}')$
- Si $p = (\overrightarrow{c}, \langle t \otimes s|t' \parr s'\rangle; \overrightarrow{u}) \rightrightarrows p' = (\overrightarrow{c}', \langle t \otimes s|t' \parr s'\rangle; \overrightarrow{u}')$, alors $p \rightrightarrows (\overrightarrow{c}', \langle t|t'\rangle, \langle s|s'\rangle; \overrightarrow{u}')$

Remarque 2. Si $p \rightrightarrows_{\mathsf{ax}} q$, alors il existe une famille de coupures $\langle x_1|t_1\rangle, \ldots, \langle x_k|t_k\rangle$ et une permutation $f \in \mathfrak{S}_k$ telles que $p = (\overrightarrow{c}, \langle x_1|t_1\rangle, \ldots, \langle x_k|t_k\rangle; \overrightarrow{t})$ et $q = (\overrightarrow{c}; \overrightarrow{t})[t_{f(1)}/\overline{x}_{f(1)}] \ldots [t_{f(k)}/\overline{x}_{f(k)}]$.

En effet, il faut et il suffit d'ordonner les substitutions de façon à ce que $\overline{x}_{f(i+1)} \in \mathcal{V}\left((\overrightarrow{c}; \overrightarrow{t})[t_{f(1)}/\overline{x}_{f(1)}] \ldots [t_{f(i)}/\overline{x}_{f(i)}]\right)$ pour tout $i \in \{1, \ldots, k\}$, et à ce que $\overline{x}_{f(1)} \in \mathcal{V}((\overrightarrow{c}; \overrightarrow{t}))$. Cela est toujours possible, en vertu de la définition des structures.

La définition de la réduction $\rightrightarrows$ correspond à une élimination arbitraire et simultanée de toutes les coupures apparaissant dans une structure. $\rightrightarrows_{\mathsf{ax}}$ et $\rightrightarrows_m$ en sont respectivement les restrictions aux coupures axiomatiques et multiplicatives. Elle ne doit pas être confondue avec une réduction itérée, car les coupures qui apparaissent ne sont pas éliminées dans la même étape.

En particulier, la réduction est définie de façon à pouvoir être standardisée, comme l'établit le lemme suivant :

Lemme 1.

1. Soit p une structure telle que $p \rightrightarrows_m q \rightrightarrows_{\mathsf{ax}} r$. Il existe q' telle que $p \rightrightarrows_{\mathsf{ax}} q' \rightrightarrows_m r$.
2. Soit p une structure telle que $p \rightrightarrows_{\mathsf{ax}} q \rightrightarrows_m r$. Il existe q' telle que $p \rightrightarrows_m q' \rightrightarrows_{\mathsf{ax}} r$.

Démonstration. Posons $p = (\overrightarrow{d}, c_1, \ldots, c_k, \langle x_1|t_1\rangle, \ldots, \langle x_l|t_l\rangle; \overrightarrow{t})$, où pour tout i, c_i est une coupure multiplicative telle que $c_i \to_m (c_{i,1}, c_{i,2})$. Et posons $r = (\overrightarrow{d}, c_{1,1}, c_{1,2}, \ldots, c_{k,1}, c_{k,2}; \overrightarrow{t})[t_1/\overline{x}_1], \ldots, [t_l/\overline{x}_l]$.

(1) Si $q = (\overrightarrow{d}, c_{1,1}, c_{1,2}, \ldots, c_{k,1}, c_{k,2}, \langle x_1|t_1\rangle, \ldots, \langle x_l|t_l\rangle; \overrightarrow{t})$ il suffit de poser $q' = (\overrightarrow{d}, c_1, \ldots, c_k; \overrightarrow{t})[t_1/\overline{x}_1], \ldots, [t_l/\overline{x}_l]$.

(2) Si $q = (\overrightarrow{d}, c_1, \ldots, c_k, \langle x_1|t_1\rangle, \ldots, \langle x_l|t_l\rangle; \overrightarrow{t})$ il suffit alors de poser $q' = (\overrightarrow{d}, c_{1,1}, c_{1,2}, \ldots, c_{k,1}, c_{k,2}, \langle x_1|t_1\rangle, \ldots, \langle x_l|t_l\rangle; \overrightarrow{t})$.

□

De la même façon, on montre que l'on peut toujours choisir une décomposition de la réduction parallèle en deux phases :

Lemme 2. Soient p, q deux structures telles que $p \rightrightarrows q$. Il existe p' telle que $p \rightrightarrows_m p' \rightrightarrows_{\mathsf{ax}} q$, et p'' telle que $p \rightrightarrows_{\mathsf{ax}} p'' \rightrightarrows_m q$.

Démonstration. Ce lemme se démontre en développant un argument similaire à celui du lemme 1. □

Le premier résultat qui nous occupe au sujet de la réduction parallèle concerne donc l'évolution des chemins d'un réseau à un autre. On cherche à capturer les comportements de la réduction qui sont susceptibles de rallonger les chemins, et de générer des chaînes de coupures. Nous avons besoin d'introduire deux notions qui nous guideront dans les preuves à suivre, et conduisant au résultat principal de cette partie : il existe une borne à l'augmentation de la longueur des chemins pendant la réduction parallèle.

La première de ces notions est une mesure sur les chemins, c'est le nombre $\mathbf{ln}(\chi)$, qui correspond à leur longueur. On verra que cette mesure est satisfaisante à plusieurs égards. D'abord, l'existence d'une borne pour cette dernière est préservée par réduction parallèle. Ensuite, quand l'objet de notre étude se muera en structures assez complexes pour décrire le développement de Taylor syntaxique des réseaux avec exponentielles, unités et constructions différentielles, nous verrons que malgré les difficultés techniques, cette mesure présentera encore les propriétés nécessaires à nos considérations combinatoires.

La deuxième notion est celle de *nœud coulant* (*slipknot* en anglais, dans nos travaux avec Vaux Auclair [CV18]). Il s'agit d'une caractérisation des fragments de chemins qui sont responsables de l'augmentation de leur longueur. C'est-à-dire que pour deux structures p, q telles que $p \rightrightarrows_m q$, il existe dans q des chemins qui n'existaient pas dans p. Pour autant, les sous-arbres de q sont des sous-arbres de p : on peut en effet vérifier facilement que la réduction multiplicative ne fait que supprimer des sous-arbres et n'en crée jamais. On peut donc comparer les chemins de p avec ceux de q, car ceux-ci sont des compositions de ceux-là. Un nœud coulant de q est, en substance, une suite de sous-arbres qui forme un chemin dans q mais pas dans p. On remarque, et c'est essentiel, que cette définition demande à ce que la réduction de p à q soit fixée au préalable.

2.2.2.1 Propriétés des chemins et des nœuds coulants

Nous caractérisons dans un premier temps les nœuds coulants à l'aide des coupures résiduelles, qui sont les liaisons permettant de voir apparaître de nouveaux chemins : les nouvelles coupures issues de coupures multiplicatives.

Définition 5 (Coupures résiduelles et nœuds coulants)**.**
Soient p et q deux structures de $\mathbf{MLL}^-$ telles que $p \rightrightarrows_m q$, et soit $c = \langle t \otimes t' | s \parr s' \rangle \in \mathbf{C}(p)$. Si si c est réduite, c'est-à-dire si $d = \langle t|s\rangle$ et $d' = \langle t'|s'\rangle$ sont dans $\mathbf{C}(q)$, alors d et d' sont appelés *coupures résiduelles* de c.

Si $\xi \in \mathbf{Ch}(q)$, et si ξ peut s'écrire $\chi_1, d, \chi_2, d', \chi_3$, pour d et d' deux coupures résiduelles d'une même coupure c de p, alors la paire d, d' est appelée un *nœud coulant* de ξ.

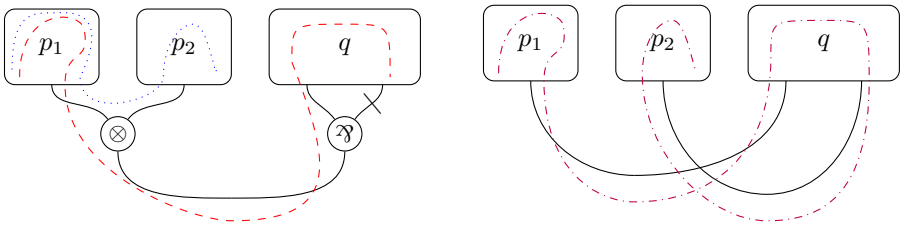


FIGURE 2.12 – Une coupure, le nœud coulant qui en résulte, et les chemins possibles

Lemme 3. Soient p une structure acyclique et q, telles que $p \rightrightarrows_m q$, et $\xi \in \mathbf{Ch}(q)$. Il existe un chemin $\xi^- \in \mathbf{Ch}(p)$ de mêmes extrémités que ξ.

Démonstration. On cherche à formaliser l'intuition visible à la figure 2.12 : les chemins peuvent être rallongés par la réduction, mais aucune accessibilité n'est créée. On considère un chemin quelconque ξ de q, pour une position d'interrupteurs I, et l'on construit ξ^-, un chemin dans p pour une position d'interrupteurs J.

La définition de ξ^- est par induction sur le nombre de coupures résiduelles dans ξ. Une difficulté est de s'assurer que les positions d'interrupteurs restent cohérentes par rapport à notre construction.

Supposons dans un premier temps $\xi = \chi_1, d, \chi_2, d', \chi_3$, pour un nœud coulant (d, d'), où d et d' sont des résidus de $c \in \mathbf{C}(p)$. On peut supposer sans perte de généralité qu'aucune coupure $c' \neq c$ n'a de résidus dans χ_1 et dans χ_3. Par définition des résidus, on peut écrire $c = \langle t \otimes t' | s \parr s' \rangle$, $d = \langle t|s\rangle$, et $d' = \langle t'|s'\rangle$. Il est alors suffisant de prouver que $\xi = \chi_1, t, s, \chi_2, s', t', \chi_3$, auquel cas on pourra écrire $\xi^- = \chi_1^-, t, t \otimes s, s, \chi_3^-$, avec χ_1^- et χ_3^- obtenus par hypothèse d'induction (ou par l'égalité $\epsilon^- = \epsilon$ pour le chemin vide). Voir la figure 2.13.

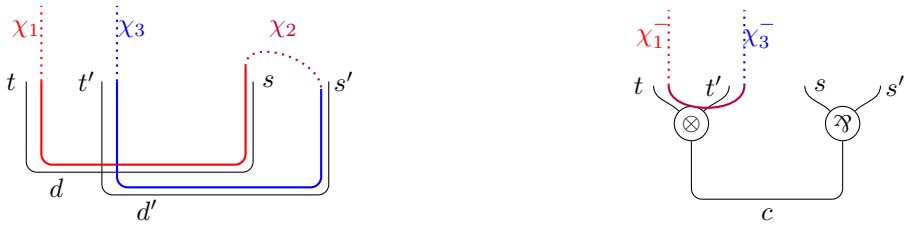


FIGURE 2.13 – Construction de ξ^- quand ξ comporte un nœud coulant

Les trois autres possibilités d'ordonnancement pour les résidus dans ξ sont les suivantes :

(a) $\chi_1, s, t, \chi_2, t', s', \chi_3$

(b) $\chi_1, s, t, \chi_2, s', t', \chi_3$

(c) $\chi_1, t, s, \chi_2, t', s', \chi_3$

On peut déjà remarquer que χ_2 ne peut pas être vide. En effet, si $t \sim_l^q t'$ (ou $t \sim_l^q s'$, ou $s \sim_l^q t'$), qlors l ne peut pas être une coupure de q car $\langle t|s\rangle$ et $\langle t'|s'\rangle \in \mathbf{C}(q)$; l ne peut pas être de la forme $(t \alpha u, t)$ pour $\alpha \in \{\otimes, \parr\}$, car t, t', s, s' sont deux-à-deux disjoints (donc l'un ne peut être sous-arbre de l'autre, en particulier) ; et l ne peut pas être un axiome, car si c'était le cas, on aurait par exemple $t = x$ et $t' = \overline{x}$, ce qui entraînerait un cycle dans p, par l'arbre $t \otimes t'$. Or, p est acyclique.

Comme χ_2 est non vide, on peut considérer u et v ses extrémités, et $\chi_2^- \in \mathbf{Ch}(p)$ d'extrémités u et v, obtenu par hypothèse d'induction. Dans les cas (a) et (b), nous avons nécessairement $t \sim_l^{q,J} u$; dans le cas (c) : $s \sim_l^{q,J} u$; dans les cas (a) et (c) : $v \sim_m^{q,J} t'$; et dans le cas (b) :$v \sim_m^{q,J} s'$, où $l \not\equiv m$ et ni l ni m n'est une coupure : il s'ensuit que les mêmes adjacences sont valables dans p pour toute extension J de I. On observe également que $t \otimes t' \notin \chi_2^-$, car on aurait alors un chemin de t (ou (t')) jusqu'à $t \otimes t'$ qui s'étendrait naturellement en un cycle. Dans le cas (a), on obtiendrait directement un cycle dans p : $t, \chi_2^-, t', t \otimes t', t$. Dans les cas (b) et (c), on déduit que $s \parr s' \notin \chi_2^-$ et l'on obtient à nouveau un cycle. Par exemple pour (b) : $t, \chi_2^-, s', s \parr s', t \otimes t', t$, pour tout interrupteur I tel que $I(s \parr s') = s'$. On a donc exclu les possibilités qu'un nœud coulant suive l'ordonnancement (a), (b) ou (c), et on peut construire ξ^- comme annoncé au début de la preuve de ce lemme.

Supposons maintenant que chaque coupure de p a au plus une coupure résiduelle apparaissant dans ξ. Si ξ ne comporte pas de résidu, on pose $\xi^- = \xi$. Sinon, soit $c = \langle t \otimes t'|s \parr s'\rangle \in \mathbf{C}(p)$, supposons (sans perte de généralité, sinon on considère $\overline{\xi}$, ou indifféremment, t' ou s') que $\xi = \chi_1, t, s, \chi_2$. On fixe alors $I(s \parr s') = s$, et $\xi^- = \chi_1^-, t, c, s, \chi_2^- \in \mathbf{Ch}(p)$. Voir figure 2.14.

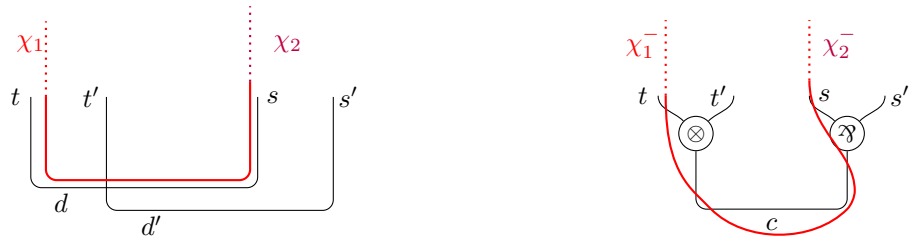


FIGURE 2.14 – Construction de ξ^- quand ξ comporte un seul résidu d'une coupure donnée

On remarque que ce dernier cas est le seul où l'on précise quelle doit être la valeur de I pour un $\parr$ donné. Ce choix, et celui fait pour former χ_1^- et χ_2^- sont indépendants. □

Du lemme précédent, on peut dériver immédiatement la propriété suivante :

Corollaire 1. Soient p une structure acyclique, et q tels que $p \rightrightarrows_m q$. q est également acyclique.

Démonstration. Il suffit en effet de remarquer que si un chemin $\xi \in \mathbf{Ch}(q)$ comporte deux occurrences d'un arbre t, alors on peut exhiber un chemin dans p ayant deux extrémités égales à t. □

À partir de maintenant, nous ne parlerons de structures que pour traiter de structures acycliques (nous vérifions également plus loin que l'acyclicité est préservée également par réduction axiomatique).

Lemme 4. Soient $p \rightrightarrows_m q$, et $\xi \in \mathbf{Ch}(p)$. Il existe $k \in \mathbf{N}$ tel que

$$\xi = \zeta_1, c_1, \chi_1, c'_1, \ldots, \zeta_k, c_k, \chi_k, c'_k, \zeta_{k+1}$$

où les paires (c_i, c'_i) sont des nœuds coulants, où les ζ_i ne comportent pas de nœud coulant, et où tout nœud coulant n'appartenant pas à $\{(c_i, c'_i) \mid 1 \leq i \leq k\}$ est un nœud coulant de l'un des χ_i.

Démonstration. Il suffit de montrer que les nœuds se suivent et s'enlacent mais ne s'emmêlent point. C'est-à-dire que si (c_1, c_2) et (d_1, d_2) sont des nœuds coulants, alors il n'existe pas de chemin de la forme $c_1, \chi_1, d_1, \chi_2, c_2, \chi_3, d_2$. En effet, supposons l'existence d'un tel chemin, pour $c = \langle t_1 \otimes s_1 | t'_1 \parr s'_1 \rangle, d = \langle t_2 \otimes s_2 | t'_2 \parr s'_2 \rangle, c_1 = \langle t_1 | t'_1 \rangle, c2 = \langle s_1 | s'_1 \rangle, d_1 = \langle t_2 | t'_2 \rangle, d_2 = \langle s_2 | s'_2 \rangle$. Alors, en utilisant le même raisonnement que dans la preuve du lemme 3, on peut affirmer que ce chemin est de la forme $t_1, t'_1, \chi_1, t_2, t'_2, \chi_2, s'_1, s_1, \chi_3, s'_2, s_2$. Considérons maintenant les chemins correspondant dans l'antiréduit p, à l'aide de la construction $()^-$ du lemme 3. Comme les extrémités sont préservées, on peut constuire le chemin de p suivant : $t'_1, \chi_1^-, t_2, d, s'_2, \overline{\chi_3}^-, s_1, c, t'_1$, qui est un cycle, ce qui contredit nos hypothèses, car nous ne considérons que des structures acycliques.

Cette caractérisation des chemins à travers les nœuds coulants est illustrée plus loin en figure 2.15. □

Corollaire 2. Pour toute réduction $p \rightrightarrows_m q$ et tout $\xi \in \mathbf{Ch}(q)$, ξ^- est unique et toujours défini.

Démonstration. Le lemme 3 assure que ξ^- soit toujours défini. Quant à l'unicité, elle découle d'une simple vérification, subsidiaire du lemme 4. □

Remarque 3. La notation ξ^-, pour un chemin $\xi \in \mathbf{Ch}(q)$, n'a de sens que relativement à un antiréduit $p \rightrightarrows_m q$.

2.2.2.2 Variations des chemins sous réduction parallèle

La mesure qui nous permettra d'étudier plus tard l'effondrement de la taille des structures consiste à compter le nombre d'arbres que peuvent comporter les chemins d'une structure.

Ce qui rend cette mesure adéquate est que, si l'on dispose d'une borne sur cette mesure, alors on sait que les chaînes de coupures d'axiomes, qui sont responsables de la diminution de la taille, n'excèdent pas une certaine longueur. De plus, on peut montrer que cette mesure reste bornée après la réduction parallèle.

Plus précisément :

Définition 6.
Soit p une structure de $\mathbf{MLL}^-$, et $\chi \in \mathbf{Ch}(p)$. On définit $\mathbf{ln}(\chi)$ la longueur de χ, c'est-à-dire le nombre d'arbres qui le composent, et l'on étend cette définition aux structures :

$$\mathbf{ln}(p) = \max\{\mathbf{ln}(\chi) \mid \chi \in \mathbf{Ch}(p)\}$$

Avant de rentrer dans les détails du lemme 6, qui montre que notre mesure $\mathbf{ln}$ sur les structures reste bornée sous réduction parallèle, et qui est un point central de cette partie, il reste une propriété technique à établir :

Lemme 5. Soient p,q tels que $p \rightrightarrows q$, et $\xi \in \mathbf{Ch}(q)$. Si ξ n'a pas de nœud coulant, alors $\mathbf{ln}(\xi) \leq \mathbf{ln}(\xi^-)$.

Démonstration. Cette propriété de montre aisément par induction sur le nombre de résidus rencontrés par ξ. Si ce nombre est égal à 0, alors $\xi^- = \xi$. S'il est égal à $k > 0$, alors il existe une coupure $c \in \mathbf{C}(p)$ telle que $\xi = \chi_1, d, \chi_2$, pour $d = /cutts$ une coupure résiduelle de $c = \langle t \otimes t' | s \parr s' \rangle$, et que, par le lemme 3, $\xi^- = \chi_1^-, t, c, s, \chi_2^-$. C'est la situation illustrée en figure 2.14.

Par définition, $\mathbf{ln}(\xi) = \mathbf{ln}(\chi_1) + \mathbf{ln}(\chi_2) + 2$, et $\mathbf{ln}(\xi^-) = \mathbf{ln}(\chi_1^-) + \mathbf{ln}(\chi_2^-) + 4$ (car la coupure correspond à un sous-chemin de taille 2). On peut appliquer l'hypothèse d'induction et conclure.

Remarquons que dans ce cas, le chemin dans le réduit est même strictement plus petit que le chemin dans l'antiréduit, le premier passant par un sous-chemin de la forme $t, t \otimes t', s \parr s', s$, et le second par un sous-chemin de la forme t, s. □

La preuve du lemme 6 se fera à travers une mesure supplémentaire sur les chemins :

Définition 7.
Soient $p \rightrightarrows_m q$ et $\xi \in \mathbf{Ch}(q)$. On définit la *largeur* de ξ : $\mathbf{larg}(\xi) = \max\{\mathbf{ln}(\chi^-) \mid \chi$ est un préfixe de $\xi\}$.

Lemme 6. Il existe une fonction croissante $f_{\mathbf{ln}} : \mathbf{N} \to \mathbf{N}$ telle que si $p \rightrightarrows_m q$ et $\xi \in \mathbf{Ch}(q)$, alors $\mathbf{ln}(\xi) \leq f_{\mathbf{ln}}(\mathbf{larg}(\xi))$.

Démonstration. Par le lemme 4, on peut écrire :

$$\xi = \zeta_1, c_1, \chi_1, c_1', \zeta_2, \ldots, c_k, \chi_k, c_k', \zeta_{k+1}$$

Où les c_i, c_i' sont des nœuds coulants, respectivement des coupures $d_i = \langle t_i \otimes t_i' | s_i \parr s_i' \rangle$, et les ζ_i ne comportent pas de nœud coulant. On peut également poser le chemin de mêmes extrémités dans l'antiréduit, par définition :

$$\xi^- = \zeta_1^-, t_1, t_1 \otimes t_1', t_1', \ldots, \zeta_k^-, t_k, t_k \otimes t_k', t_k, \zeta_{k+1}^-$$

Voir la figure 2.15.

Commençons par lister quelques propriétés que l'on peut déduire des définitions et des résultats précédents :

(a) $\sum_{i=1}^{k+1} \mathbf{ln}(\zeta_i) \leq \mathbf{ln}(\xi^-) - 3k$. En effet, on peut observer que $\mathbf{ln}(\xi^-) = \sum_{i=1}^{k+1} \mathbf{ln}(\zeta_i^-) + 3k$, et par le lemme 5, $\mathbf{ln}(\zeta_i) \leq \mathbf{ln}(\zeta_i^-)$ pour tout i.

(b) Pour tout $i \in \{1, \ldots, k\}$, $\mathbf{larg}(\chi_i) \leq \mathbf{larg}(\xi) - 4$. Pour cela, on considère χ_i' un préfixe quelconque de χ. Par définition, $\eta_i = \zeta_1, c_1, \chi_1, \ldots, \zeta_i, c_i, \chi_i'$ est un préfixe de ξ. Or, $\eta_i^- = \zeta_1^-, \ldots, \zeta_i^-, t_i, t_i \otimes t_i', s_i \parr s_i', s_i, \chi_i'^-$, ce qui implique en particulier que $\mathbf{ln}(\chi_i'^-) \leq \mathbf{ln}(\eta_i^-) - 4$.

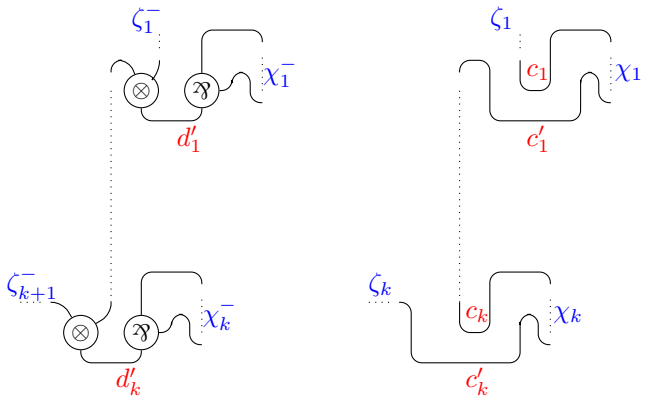


FIGURE 2.15 – Nœuds coulants et chemins dans l'antiréduit.

(c) $k \leq \frac{\mathbf{larg}(\xi)}{3}$. Observons que pour tout $i \in \{1, \ldots, k\}$, on a $t_i, t_i \otimes t'_i, t'_i$ qui est sous chemin de ξ^-, donc la longueur de (ξ^-) est d'au moins $3k$. Or, $\mathbf{ln}(\xi^-) \leq \mathbf{larg}(\xi)$.

(d) Si $k > 0$, c'est-à-dire si ξ comporte au moins un nœud coulant, alors $\mathbf{larg}(\xi) \geq 4$. En effet, dans ce cas, il existe au moins un sous-chemin de ξ c_i, χ_i, c'_i, et les coupures c_i comportent deux arbres chacune.

Soit $f_{\mathbf{ln}} : \mathbf{N} \to \mathbf{N}$ définie comme suit :

— $f_{\mathbf{ln}}(n) = n$ si $n < 4$

— $f_{\mathbf{ln}}(n) = \frac{4n}{3} + \frac{n}{3} f_{\mathbf{ln}}(n-4)$ sinon.

On va maintenant procéder à une récurrence sur la largeur de ξ pour montrer que $\mathbf{ln}(\xi) \leq f_{\mathbf{ln}}(\mathbf{larg}(\xi))$. Si $\mathbf{larg}(\xi) < 4$, alors ξ n'a pas de nœud coulant (point (d)). Par le lemme 5, on a $\mathbf{ln}(\xi) \leq \mathbf{ln}(\xi^-)$. Or, on a vu que, par définition, $\mathbf{ln}(\xi^-) \leq \mathbf{larg}(\xi) = f_{\mathbf{ln}}(\mathbf{larg}(\xi))$.

Si $\mathbf{larg}(\xi) \geq 4$, alors ξ peut contenir des nœuds coulants. Par définition de la longueur, on a :

$$\mathbf{ln}(\xi) = \sum_{i=1}^{k+1} \mathbf{ln}(\zeta_i) + \sum_{j=1}^{k} \mathbf{ln}(\chi_j) + 4k$$

En effet, ξ comporte en plus des chemins ζ_i et χ_j les $2k$ coupures c_i, c'_i, qui sont chacune de longueur 2.

Par le point (a), on peut poser :

$$\mathbf{ln}(\xi) \leq \mathbf{ln}(\xi^-) - 3k + \sum_{j=1}^{k} \mathbf{ln}(\chi_j) + 4k$$

et, comme $\mathbf{ln}(\xi^-) \leq \mathbf{larg}(\xi)$, on obtient :

$$\mathbf{ln}(\xi) \leq \mathbf{larg}(\xi) + k + \sum_{j=1}^{k} \mathbf{ln}(\chi_j)$$

Par le point (b), $\mathbf{larg}(\chi_i) \leq \mathbf{larg}(\xi) - 4$. Or, par hypothèse de récurrence, on a $\mathbf{ln}(\chi_i) \leq f_{\mathbf{ln}}(\mathbf{larg}(\chi_i))$. Donc :

$$\mathbf{ln}(\xi) \leq \mathbf{larg}(\xi) + k + k\,(f_{\mathbf{ln}}(\mathbf{larg}(\xi) - 4))$$

Par le point (c), on peut borner k en fonction de la largeur de ξ, et conclure :

$$\mathbf{ln}(\xi) \leq \frac{4\mathbf{larg}(\xi)}{3} + \frac{\mathbf{larg}(\xi)}{3} f_{\mathbf{ln}}(\mathbf{larg}(\xi) - 4) = f_{\mathbf{ln}}(\mathbf{larg}(\xi))$$

□

On a donc une borne sur l'augmentation de la longueur des chemins sous réduction parallèle. Nous nous intéressons à la finitude, donc la grandeur de ces bornes n'est pas pertinente pour nos arguments. Mais signalons néanmoins que $f_{\mathbf{ln}}(n)$ est de l'ordre de $n!$.

On peut désormais déduire de ce qui précède la propriété globale qui nous intéresse.

Lemme 7. Soient $p \rightrightarrows_m q$. $\mathbf{ln}(q) \leq f_{\mathbf{ln}}(\mathbf{ln}(p))$.

Démonstration. Soit ξ un chemin de q. Par le lemme 6, $\mathbf{ln}(\xi) \leq f_{\mathbf{ln}}(\mathbf{larg}(\xi))$. Or, la largeur de ξ est défini comme un maximum de longueur sur des chemins de p (voir définition 7). En particulier, pour tout $\chi \in \mathbf{Ch}(q)$, on a $\mathbf{larg}(\chi) \leq \mathbf{ln}(p)$.

On conclut bien $\mathbf{ln}(\xi) \leq f_{\mathbf{ln}}(\mathbf{ln}(p))$, et donc $\mathbf{ln}(q) \leq f_{\mathbf{ln}}(\mathbf{ln}(p))$ comme voulu. □

Cette propriété de borne sur le rallongement des chemins d'interrupteurs est plus commode à établir lorsqu'il s'agit de la réduction axiomatique. Il suffit en effet de formaliser le fait que dans ce cas, les chemins sont contractés, et non déliés comme dans le cas multiplicatif. Ils rencontrent donc moins d'arbres dans le réduit.

Lemme 8. Soient $p \to_{\mathsf{ax}} q$ tels que $p = (\langle x|u\rangle, \overrightarrow{c}; \overrightarrow{t})$ soit acyclique, et $q = (\overrightarrow{c}; \overrightarrow{t})[u/\overline{x}]$. Alors à tout chemin $\chi \in \mathbf{Ch}(q)$, on peut associer un chemin χ^+ dans p qui est plus long que χ, et tel que si χ a pour extrémités t, s, alors χ^+ a pour extrémités $t[u/\overline{x}]$ et $s[u/\overline{x}]$.

Démonstration. Remarquons dans un premier temps que les occurrences de sous-arbres étant toujours linéaires dans nos structures, il n'y a pas de contre-indication à utiliser la notation $t[\overline{x}/u]$ pour désigner l'antécédent d'un arbre $t \in \mathbf{SA}(q)$ par la substitution $[u/\overline{x}]$, définie de la façon naturelle. On étend cette substitution aux étiquettes d'adjacence et aux chemins.

Une simple vérification permet alors de constater que si $t \sim_l^q s$ et si $t \neq u \neq s$, alors $t[\overline{x}/u] \sim^p_{l[\overline{x}/u]} s[\overline{x}/u]$. Il est en effet naturel que des substitutions dans les éventuels sous-arbres stricts n'entravent pas l'adjacence entre les parents.

Soit donc $\chi \in \mathbf{Ch}(q)$, tel que pour tout arbre t traversé par χ, $t \neq u$. Alors on pose $\chi^+ = \chi[\overline{x}/u]$.

N'importe quel autre chemin de q est alors nécessairement de la forme $\chi = \chi'_1, u, \chi'_2$. Posons $\chi = \chi_1, t, u, s, \chi_2$ (les cas où χ'_1 ou χ'_2 est vide s'en déduisent), et plusieurs configurations sont possibles :

— $t, s \in \mathbf{SA}(u)$. Dans ce cas on pose $\chi^+ = \chi_1^+, t, u, s, \chi_2^+$. C'est bien un chemin de p, car par hypothèse de récurrence, on considère construit, $(\chi_1, t)^+ = \chi_1^+, t[\overline{x}/u]$, qui est un chemin de p, or $t[\overline{x}/u] = t$ (t ne peut pas être sous-arbre de u et avoir u pour sous-arbre). Même argument pour $(s, \chi_2)^+$. On vérifie bien que les adjacences entre arbres de χ^+ sont correctes dans p.

— u est un sous-arbre immédiat de t. Dans ce cas, s est sous-arbre de u, car u ne peut être sous-arbre immédiat de deux arbres distincts, ni prémisse d'une coupure.

Dans ce cas, $t[\overline{x}/u] \neq t$, et $\overline{x}$ est un sous arbre immédiat de $t[\overline{x}/y]$. On a donc $t \sim^p_{l[\overline{x}/u]} \overline{x}$, où l est l'étiquette telle que $t \sim^q_l u$. On pose alors $\chi^+ = \chi_1^+, t[\overline{x}/u], \overline{x}, x, u, s, \chi_2^+$.

Il s'agit bien d'un chemin de p car $\overline{x} \sim^p_{x,\overline{x}} x, x \sim_{\langle x|u\rangle} u$, s est sous-arbre de u, donc $s[\overline{x}/u] = s$, et $(s, \chi_2)^+ = s[\overline{x}/u], \chi_2^+$ est un chemin de p.

— $\langle t|u\rangle \in \mathbf{C}(q)$. Le même raisonnement que celui du cas précédent s'applique.

— Les autres possibilités sont symétriques à celles déjà traitées.

□

Corollaire 3. Soient p, q tels que $p \rightrightarrows_{\mathsf{ax}} q$. $\mathbf{ln}(q) \leq \mathbf{ln}(p)$.

Démonstration. Il suffit de remarquer dans la preuve du lemme 8 que pour $p \to_{\mathsf{ax}} p_1$, à chaque chemin de p_1 on associe un chemin de p qui visite au moins autant d'arbres que le premier.

On itère ce résultat autant de fois que nécessaire, en se rappelant que si $p \rightrightarrows_{\mathsf{ax}} q$, il existe une suite d'arbres $(p_1, \ldots, p_k)$ telle que $p \to_{\mathsf{ax}} p_1 \to_{\mathsf{ax}} p_2 \ldots \to_{\mathsf{ax}} p_k \to_{\mathsf{ax}} q$. □

Corollaire 4. L'acyclicité est préservée par la réduction axiomatique.

Démonstration. De la preuve du lemme 8, on peut déduire de l'existence d'un cycle dans le réduit un cycle dans l'antiréduit. Le résultat s'étend à la réduction parallèle de la même façon que ci-dessus. □

On peut maintenant donner des résultats généraux sur la réduction parallèle, grâce notamment au résultat de standardisation établi plus tôt.

Théorème 1. Soient $p \rightrightarrows q$. $\mathbf{ln}(q) \leq f_{\mathbf{ln}}(\mathbf{ln}(p))$.

Démonstration. Par le lemme 2 (standardisation), il existe p' tel que $p \rightrightarrows_m p' \rightrightarrows_{\mathsf{ax}} q$. Le reste découle de la combinaison du lemme 7 et du Corollaire 3. □

Corollaire 5. Soient $p \rightrightarrows q$ et p acyclique, q est acyclique.

Démonstration. Par standardisation et combinaison des corollaires 1 et 4 □

2.2.3 Parlons de taille

Nous allons désormais, à l'aide des résultats précédents, pouvoir caractériser l'évolution de la taille des structures durant la réduction. Pour cela, nous traitons séparément les phases multiplicative et axiomatique. Nous commençons par définir la taille des structures.

Nous prenons cependant la précaution de définir la taille d'objets plus généraux que les structures, de façon à pouvoir parler indépendamment de la taille de $(\overrightarrow{c}; \overrightarrow{t})$ et de la taille de $(\overrightarrow{c}; \overrightarrow{t})[u/x]$ par exemple, alors qu'en général, si l'un de ces deux objets est une structure, l'autre n'en est pas une. Ceci est dû aux conditions de fermeture.

Définition 8.
Nous définissons la mesure **taille**() pour les arbres, les coupures, et les familles d'arbres et de coupures.

- Pour toute variable x, $\mathbf{taille}(x) = 1$.
- $\mathbf{taille}(t \alpha s) = \mathbf{taille}(t) + \mathbf{taille}(s) + 1$, pour $\alpha \in \{\otimes, \parr\}$.
- $\mathbf{taille}(\langle t|s\rangle) = \mathbf{taille}(t) + \mathbf{taille}(s)$.
- $\mathbf{taille}\left((c_1, \ldots, c_k; t_1, \ldots, t_l)\right) = \sum_{i=1}^{k} \mathbf{taille}(c_i) + \sum_{j=1}^{l} \mathbf{taille}(t_j)$.

2.2.3.1 Taille des structures et réduction multiplicative

Rappelons que l'on cherche à évaluer l'effondrement de la taille des structures sous réduction parallèle. Si l'on commence cette étude en se restreignant à la réduction parallèle multiplicative, on peut montrer aisément que la perte de taille est constante.

Lemme 9. Soient $p \rightrightarrows_m q$. $\mathbf{taille}(p) \leq \frac{3\mathbf{taille}(q)}{2}$.

Démonstration. Il suffit de remarquer que si $c = \langle t \otimes t' | s \parr s' \rangle$ est une coupure multiplicative, alors la taille de t, t', s, s' vaut au moins 1. De plus, si $d = \langle t|s\rangle$ et $d' = \langle t'|s'\rangle$, alors il suit que $\mathbf{taille}(c) = \mathbf{taille}(d) + \mathbf{taille}(d') + 2$.

Posons $p = (c_1, \ldots, c_k, \overrightarrow{c}; \overrightarrow{t})$ et $q = (d_1, d'_1, \ldots, d_k, d'_k, \overrightarrow{c}; \overrightarrow{t})$, avec pour tout i, $c_i = \langle t_i \otimes t'_i | s_i \parr s'_i \rangle$, $d_i = \langle t_i | s_i \rangle$ et $d'_i = \langle t'_i | s'_i \rangle$. Par le point précédent, $\mathbf{taille}(p) = \mathbf{taille}(q) + 2k$. Or, on a bien $\mathbf{taille}(q) + 2k \leq \frac{3\mathbf{taille}(q)}{2}$ dès que $\mathbf{taille}(q) \geq 4k$, ce qui est le cas car les arbres sont de taille non nulle. □

2.2.3.2 Taille des structures et réduction axiomatique

La situation est tout à fait différente dans le cas de la réduction des axiomes, car les chaînes de coupures se contractent, et voient la taille de la structure de réduire d'autant d'arbres qu'il y avait de variables le long de la chaîne en question.

Si l'on considère une structure $p = (\langle x_1|\overline{x}_2\rangle, \ldots, \langle x_{k-1}|\overline{x}_k\rangle, \langle x_k|u\rangle, \overrightarrow{c}; \overrightarrow{t})$, on constate que $p \rightrightarrows_{\mathsf{ax}} q = (\overrightarrow{c}; \overrightarrow{t})[u/\overline{x}_1]$. Dans ce cas, il apparaît que q contient $2k$ variables de moins que p. La perte de taille n'est donc plus constante. Cette situation est illustrée en figure 2.16.

La difficulté ici sera donc d'isoler ces cas de figure pour en caractériser le comportement, et d'en extraire des bornes satisfaisantes. L'idée est d'utiliser la mesure **ln**() étudiée précédemment. On pourra en effet voir que le cas critique

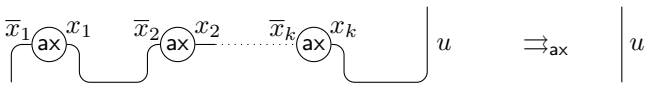


FIGURE 2.16 – Effondrement de la taille lors de la réduction d'une chaîne d'axiomes.

illustré en figure 2.16 sera contrôlé par le fait que les chaînes d'axiomes de longueur k (k axiomes, $2k$ variables) induisent un chemin longeant cette chaîne, et composé d'au moins $2k$ arbres. Si nous avons connaissance de $\mathbf{ln}(p)$, nous pouvons alors borner la longueur maximale des chaînes d'axiomes y apparaissant. Ainsi, le lemme 10 traite ces cas critiques responsables de l'effondrement de la taille des structures.

L'idée de la preuve est ici de caractériser précisément la forme des structures contenant des chaînes de coupures, pour détailler l'évolution de la taille en s'appuyant sur le fait que la longueur de ces chaînes sont bornées. En particulier, il s'agit d'écrire les substitutions issues de la réduction dans un ordre particulier. On peut déjà remarquer dans la figure 2.16 qu'effectuer la suite de substitutions $([\overline{x}_2/\overline{x}_1]\dots[\overline{x}_k/\overline{x}_{k-1}],[u/\overline{x}_k])$, qui correspond à une exécution correcte de l'élimination des coupures, revient à procéder à l'unique substitution $[u/\overline{x}_1]$. Isoler donc les chaînes de coupures d'axiomes les plus longues permet de ramener chaque chaîne à une seule substitution, et de traiter l'évolution de la taille en conséquence, en utilisant le fait que $\mathbf{taille}(q[t/x]) = \mathbf{taille}(q)-2+\mathbf{taille}(t)$ (car les deux variables x et $\overline{x}$ sont effacées).

Lemme 10. Soient $p \Rightarrow_{\mathsf{ax}} q$. $\mathbf{taille}(p) \leq \mathbf{ln}(p)\mathbf{taille}(q)$.

Démonstration. Posons $p = (\langle x_1|t_1\rangle,\dots,\langle x_n|t_n\rangle,\overrightarrow{c};\overrightarrow{s})$ et posons son réduit $q = (\overrightarrow{c};\overrightarrow{s})[t_1/\overline{x}_1]\cdots[t_n/\overline{x}_n]$ avec $\overline{x}_i \notin \mathcal{V}(t_j)$ pour $1 \leq i \leq j \leq n$. Pour établir le résultat dans ce cas, on rend explicites les chaînes d'axiomes.

Grâce à la condition sur les variables libres dans les structures, il existe une permutation (nécessairement unique) de $\langle x_1|t_1\rangle,\dots,\langle x_n|t_n\rangle$ menant à une famille de la forme $\overrightarrow{c}_1,\dots,\overrightarrow{c}_k$ telle que :

- pour $1 \leq i \leq k$, on peut écrire $\overrightarrow{c}_i = \langle x^i_0|\overline{x}^i_1\rangle,\dots,\langle x^i_{n_i-1}|\overline{x}^i_{n_i}\rangle,\langle x^i_{n_i}|t^i\rangle$;
- chaque $\overrightarrow{c}_i$ est maximal avec cette forme, *i.e.* $\overline{x}^i_0 \notin \{x_1,\dots,x_n,t_1,\dots,t_n\}$ et, si t^i est une variable, $\overline{t}^i \notin \{x_1,\dots,x_n,t_1,\dots,t_n\}$;
- si $i<j$, alors $\langle x^i_{n_i}|t^i\rangle$ apparaît avant $\langle x^j_{n_j}|t^j\rangle$ dans $\langle x_1|t_1\rangle,\dots,\langle x_n|t_n\rangle$.

Il suit que si $\overline{x}^i_0 \in \mathcal{V}(t_j)$, alors $j<i$, et donc $q = (\overrightarrow{c};\overrightarrow{s})[t^1/\overline{x}^1_0]\cdots[t^k/\overline{x}^k_0]$, en appliquant la même permutation aux substitutions, de la même façon qu'avec les coupures : Nous pouvons procéder ainis car par un argument standard, si $x \neq y$, $x \notin \mathcal{V}(u)$ et $y \notin \mathcal{V}(u)$ alors $\gamma[u/x][v/y] = \gamma[v/y][u/x]$. Pour $1 \leq i \leq k$, comme $\overrightarrow{c}_i$ induit un chemin composé de $2(n_i)+3$ arbres (le chemin $\overline{x}^i_0, x^i_0,\dots,\overline{x}^i_{n_i}, x^i_{n_i}, t^i$), il suit que $2n_i \leq \mathbf{ln}(p)-3$.

Par définition, on a :

$$\mathbf{taille}(p) = \mathbf{taille}(\overrightarrow{c}) + \mathbf{taille}(\overrightarrow{s}) + \sum_{i=1}^{k}(\mathbf{taille}(t^i)+2n_i+2)$$

Par ce qui précède, on peut donc poser :

$$\begin{aligned}\mathbf{taille}(p) &\leq \mathbf{taille}(\overrightarrow{c}) + \mathbf{taille}(\overrightarrow{s}) + \sum_{i=1}^{k} \mathbf{taille}(t^i) + k\left((\mathbf{ln}(p) - 3) + 2\right) \\ &= \mathbf{taille}(\overrightarrow{c}) + \mathbf{taille}(\overrightarrow{s}) + \sum_{i=1}^{k} \mathbf{taille}(t^i) + k(\mathbf{ln}(p) - 1)\end{aligned}$$

De plus, $\mathbf{taille}(q) = \mathbf{taille}(\overrightarrow{c}) + \mathbf{taille}(\overrightarrow{s}) + \sum_{i=1}^{k} \mathbf{taille}(t^i) - k$. On a donc $\mathbf{taille}(p) \leq \mathbf{taille}(q) + k(\mathbf{ln}(p) - 1)$ et, pour conclure, il sera suffisant de montrer que $\mathbf{taille}(q) \geq k$.

Pour $1 \leq i \leq k$, soit $A_i = \{j > i \mid \overline{x}_0^j \in \mathcal{V}(t^i)\}$, et soit $A_0 = \{i \mid \overline{x}_0^i \in \mathcal{V}(\overrightarrow{c}, \overrightarrow{s})\}$. Il découle de la construction que $\{A_0, \ldots, A_{k-1}\}$ est une partition (pouvant contenir des ensembles vides) de $\{1, \ldots, k\}$. Par construction, $\mathbf{taille}(t^i) > \mathbf{card}(A_i)$. Considérons maintenant $q_i = (\overrightarrow{c}; \overrightarrow{s})[t^1/\overline{x}_0^1] \cdots [t^i/\overline{x}_0^i]$ pour $0 \leq i \leq k$ tel que $q = q_k$. Pour $1 \leq i \leq k$, on obtient $\mathbf{taille}(q_i) = \mathbf{taille}(q_{i-1}) + \mathbf{taille}(t^i) - 1 \geq \mathbf{taille}(q_{i-1}) + \mathbf{card}(A_i)$. Observons également que $\mathbf{taille}(q_0) = \mathbf{taille}(\overrightarrow{c}; \overrightarrow{s}) \geq \mathbf{card}(A_0)$. On peut enfin conclure : $\mathbf{taille}(q) = \mathbf{taille}(q_k) \geq \sum_{i=0}^{k} \mathbf{card}(A_i) = k$.

□

2.2.3.3 Taille des structures en général et finitude de l'antiréduction

Nous pouvons grâce aux résultats précédents donner un premier résultat de finitude de l'antiréduction parallèle des réseaux. Celui-ci est conditionné à l'existence d'une borne sur $\mathbf{ln}(p)$, pour les structures p concernées. En tant que résultat sur **MLL**, cette contrainte n'a pas de signification particulière ; mais rappelons que l'objet d'étude vers lequel nous nous dirigeons, le développement de Taylor des réseaux avec exponentielles et unités, consistera en des combinaisons linéaires infinies de réseaux dont le comportement est essentiellement multiplicatif. Il s'agira en tant voulu de montrer que ces combinaisons respectent une certaine borne sur $\mathbf{ln}(p)$, pour tous les p y apparaissant, et d'importer les résultats suivants en les adaptant à la syntaxe adéquate pour obtenir les propriétés voulues.

Commençons par une propriété naturelle, mais essentielle et qui justifie notre obsession relative aux questions de taille.

Propriété 1. Soit k un entier naturel. $\{p \in \mathbf{MLL}^- \mid \mathbf{taille}(p) \leq k\}$ est un ensemble fini.

Démonstration. Il suffit de se rappeler que nos structures sont quotientées par le renommage des variables. En particulier, il n'existe aucune structure de taille 1, et il existe deux structures de taille 2 : $(;x,\overline{x})$ et $(\langle x|\overline{x}\rangle;)$ (si l'on considère les structures cycliques, ce qui ne change pas le résultat).

On remarque ensuite qu'il n'existe pour toute structure de taille k qu'un nombre fini d'extensions en une structure de taille $k+1$ ou $k+2$ (il faut considérer le cas $+2$ pour l'ajout d'un axiome). □

Théorème 2. Soient $k, l \in \mathbf{N}$, et q une structure de $\mathbf{MLL}^-$. $\{p \in \mathbf{MLL}^- \mid \mathbf{ln}(p) \leq l \wedge p \rightrightarrows^k q\}$ est un ensemble fini.

Démonstration. Par induction sur k. Si $k = 0$, alors l'ensemble ne contient que p.

Si $k = k' + 1$, alors $\{p \in \mathbf{MLL}^- \mid \mathbf{ln}(p) \leq l \wedge p \rightrightarrows^k q\} = \{p \in \mathbf{MLL}^- \mid \mathbf{ln}(p) \leq l \wedge \exists p_1 \in \mathbf{MLL}^- \wedge p \rightrightarrows p_1 \rightrightarrows^{k'} q\}$. Par le théorème 1, $\{\mathbf{ln}(p_1) \mid p \rightrightarrows p_1\}$ est borné par $f_{\mathbf{ln}}(\mathbf{ln}(p))$. Donc par hypothèse d'induction, $X = \{p_1 \in \mathbf{MLL}^- \mid \exists p \rightrightarrows p_1 \wedge \mathbf{ln}(p) \leq l \wedge p_1 \rightrightarrows^{k'} q\} \subseteq \{p_1 \in \mathbf{MLL}^- \mid \mathbf{ln}(p_1) \leq f_{\mathbf{ln}}(l) \wedge p_1 \rightrightarrows^{k'} q\}$ est fini.

Soit $p_1 \in X$. Par le lemme 2, $\{p \mid p \rightrightarrows p_1\} = \{p \mid \exists p', p \rightrightarrows_{\mathsf{ax}} p' \rightrightarrows_m p_1\}$. Or, par le lemme 9, $\{\mathbf{taille}(p') \mid p' \rightrightarrows_m p_1\}$ est borné par $\frac{3\mathbf{taille}(p_1)}{2}$. En particulier, la propriété 1 permet d'affirmer que $\{p' \mid p' \rightrightarrows_m p_1\}$ est fini.

De même, par le lemme 10, $\{\mathbf{taille}(p) \mid p \rightrightarrows_{\mathsf{ax}} p' \wedge \mathbf{ln}(p) \leq l\}$ est borné par $l(\mathbf{taille}(p'))$, et par la propriété 1, $\{p \mid p \rightrightarrows_{\mathsf{ax}} p' \wedge \mathbf{ln}(p) \leq l\}$ est fini pour tout p'.

On a donc, en regroupant les deux points qui précèdent, $\{p \mid p \rightrightarrows p_1 \wedge \mathbf{ln}(p) \leq l\}$ fini pour tout p_1. Or, $X \subseteq \{p_1 \in \mathbf{MLL}^- \mid \mathbf{ln}(p_1) \leq f_{\mathbf{ln}}(l) \wedge p_1 \rightrightarrows^k q\}$ est fini, donc $\{p \in \mathbf{MLL}^- \mid p \rightrightarrows p_1 \in X\} = \{p \in \mathbf{MLL}^- \mid \mathbf{ln}(p) \leq l \wedge p \rightrightarrows^k q\}$ est également fini. □

2.3 Fragment multilinéaire des réseaux différentiels

Dans cette section, nous présentons les réseaux à ressources, qui correspondent aux réseaux différentiels définis par Ehrhard et Régnier [ER05]. Ce système correspond à celui dans lequel le développement de Taylor est défini. Il suit que pour obtenir le résultat de finitude des coefficients dans le développement, il nous faut établir un résultat similaire au théorème 2 de la section 2.2.3.3.

Deux différences importantes vont rendre la tâche plus compliquée qu'elle ne l'était dans $\mathbf{MLL}^-$. La première est que nous considérons désormais des structures avec unités, co-unités, affaiblissements et coaffaiblissements, ce qui demande un traitement particulier des éléments non connexes des structures. Le comportement de la réduction à cet endroit sera problématique, nous en parlerons, pour des raisons qui deviendront claires le moment venu, dans les termes de *réductions évanescentes*.

La seconde consiste dans la réduction entre les connecteurs co-structurels, qui en faisant intervenir des nœuds d'arité arbitraire sera plus délicate à décrire sous une forme générale. Pour autant, la difficulté sera justement de montrer que malgré sa complexité apparente, les éléments critiques à considérer pour établir nos résultats sont en essence contenus dans la réduction multiplicative que nous avons déjà examinée.

2.3.1 Constructions

2.3.1.1 Structures

On continue à considérer les connecteurs multiplicatifs binaires, de façon à pouvoir appliquer les résultats de la section 2.2 directement. Nous attirons cependant l'attention du lecteur sur le fait qu'étendre au cas n-aire demande quelques ajustements techniques, qui sont traités dans notre travail avec Vaux

Auclair [CV18]. Par ailleurs, ces détails seront réglés pour la réduction des coupures entre contraction et cocontraction, qui sont d'arité arbitraire.

Cependant, nous utiliserons la notation fonctionnelle pour les arbres multiplicatifs, par confort et pour davantage d'uniformité dans les notations.

Définition 9 (Réseaux à ressources, structures simples)**.**
On définit les réseaux à ressources, que l'on appellera structures simples de $\mathbf{DLL}_0$. Les connecteurs d'arité 0 seront traités à part et abrégés ainsi : $\mathbf{1} = \otimes(), \bot = \parr(), \mathbf{a} = c(), \overline{\mathbf{a}} = \overline{c}()$.

On se donne donc ici cinq ensembles dénombrables, deux-à-deux disjoints, $\mathcal{V}, \mathbf{U_1}, \mathbf{U}_\bot, \mathbf{U}_?, \mathbf{U}_!$, ainsi qu'un ensemble $\mathbf{A}$ d'*atomes*, consititué de la réunion de ces derniers.

On décrit les pré-arbres de $\mathbf{DLL}_0$ ci-dessous, où $\mu \in \mathbf{U_1}, \nu \in \mathbf{U}_\bot, \kappa \in \mathbf{U}_?, \lambda \in \mathbf{U}_!$, et $x \in \mathcal{V}$. $\mathcal{V}$ est muni d'une involution $\overline{()}$.

$$t, s ::= x \mid \mathbf{1}_\mu \mid \bot_\nu \mid \mathbf{a}_\kappa \mid \overline{\mathbf{a}}_\lambda \mid \otimes(t,s) \mid \parr(t,s) \mid d(t) \mid \overline{d}(t) \\ \mid c(t_1, \ldots, t_k) \mid \overline{c}(t_1, \ldots, t_k)$$

où $k \neq 0$. Les connecteurs 0-aires ne sont pas définis, car ils sont traités en tant qu'atomes. On identifiera le plus souvent les (co)unités et (co)affaiblissements avec leur indice, de façon à distinguer systématiquement leurs occurrences. Ainsi, l'ensemble $\mathbf{A}$ sera considéré comme un ensemble de pré-arbres, dits alors *atomiques* et non un ensemble d'indices.

On définit $\mathbf{SA}(t)$ pour tout préarbre t. Si $t \in \mathbf{A}$ est un pré-arbre atomique, alors $\mathbf{SA}(t) = \{t\}$. Si $t = \alpha(s_1, \ldots, s_n)$ pour $\alpha \in \{\otimes, \parr, d, \overline{d}, c, \overline{c}\}$, alors $\mathbf{SA}(t) = \{t\} \cup \bigcup_{i=1}^n \mathbf{SA}(s_i)$.

On écrit de plus $\mathbf{X}(t)$ pour $\mathbf{SA}(t) \cap \mathbf{X}$, si $\mathbf{X} \in \{\mathcal{V}, \mathbf{U_1}, \mathbf{U}_\bot, \mathbf{U}_?, \mathbf{U}_!\}$.

Un *arbre* de $\mathbf{DLL}_0$ est soit un pré-arbre atomique, soit un préarbre de la forme $\alpha(t_1, \ldots, t_n)$ tel que pour tous $i \neq j \in \{1, \ldots, n\}, \mathbf{SA}(t_i) \cap \mathbf{SA}(t_j) = \emptyset$.

Une *coupure* de $\mathbf{DLL}_0$ est une paire d'arbres commutative notée $\langle t | s \rangle$ telle que $\mathbf{SA}(t) \cap \mathbf{SA}(s) = \emptyset$. On pose $\mathbf{SA}(\langle t | s \rangle) = \mathbf{SA}(t) \cup \mathbf{SA}(s)$.

Une *pré-structure de* $\mathbf{DLL}_0$ est la donnée d'une suite de coupures $c_1, \ldots, c_n$ et d'une suite d'arbres $t_1, \ldots, t_m$ telles que pour tous $i, j \in \{1, \ldots, n\}$, $\mathbf{SA}(c_i) \cap \mathbf{SA}(c_j) = \emptyset$ dès que $i \neq j$, et pour tous $k, l \in \{1, \ldots, m\}$, $\mathbf{SA}(t_k) \cap \mathbf{SA}(t_l) = \emptyset$ dès que $k \neq l$. Les structures sont alors notées $p = (c_1, \ldots, c_n; t_1, \ldots, t_m)$. On note $\mathbf{C}(p)$ l'ensemble des coupures de p. On définit l'ensemble des sous-arbres de p comme suit : $\mathbf{SA}(p) = \bigcup_{i=1}^n \mathbf{SA}(c_i) \cup \bigcup_{j=1}^m \mathbf{SA}(t_j)$.

Enfin, une *structure simple* de $\mathbf{DLL}_0$ est une pré-structure p telle que pour toute variable $x \in \mathcal{V}$, si $x \in \mathbf{SA}(p)$ alors $\overline{x} \in \mathbf{SA}(p)$.

Une *structure à sauts* est une structure simple munie d'une *fonction de saut* notée S_p, de domaine $\mathbf{SA}(p) \cap (\mathbf{U}_\bot \cup \mathbf{U}_?)$ et de codomaine $\mathbf{SA}(p)$.

Les structures sont considérées à α-équivalence près (voir la définition 1), et à renommage des (co)unités et des (co)affaiblissements près. Il faut néanmoins prendre soin à ce que les structures restent bien définies pendant ces renommages, au niveau des sauts notamment. C'est-à-dire que si $t \notin \mathbf{U_1}(p) \cup \mathbf{U}_!(p)$ et $u \in \mathbf{U_1} \cup \mathbf{U}_!$, pour une structure p, alors $(p, S_p) = (p[t/u], ([\{s \to t\}/\{s \to u\}])_{\{s \to u\} \in S_p})$. Si $t \notin \mathbf{U}_\bot(p) \cup \mathbf{U}_?(p)$ et $u \in \mathbf{U}_\bot \cup \mathbf{U}_?$, alors $(p, S_p) = (p[t/u], S_p[\{t \to r\}/\{u \to r\}], ([\{s \to t\}/\{s \to u\}])_{\{s \to u\} \in S_p})$ (on conserve les sauts lors du renommage).

Nous développerons en section 2.3.3 les constructions relatives aux sauts qui permettront de contrôler le comportement des counités et des affaiblissements. Mais dans un premier temps, en section 2.3.2, nous nous concentrerons sur les structures simples et ne traiterons pas, le plus souvent, de structures à sauts. Exception faite de certaines propriétés clefs (par exemple le lemme 11), que nous verrons dans le cas le plus général pour éviter d'inutiles répétitions.

Définition 10.
L'ensemble des *structures* de $\mathbf{DLL}_0$ correspond à l'ensemple des sommes finies de structures simples, et est noté $\mathbf{DLL}_0^+$.

2.3.1.2 Chemins

Les chemins sont définis de façon similaire à la définition 3, en étendant le comportement du tenseur à la cocontraction, celui du parr à la contraction, et en intégrant les sauts. Il suffit de les définir dans les structures à sauts, et non dans $\mathbf{DLL}_0^+$, dans la mesure où les chemins ne traversent pas les sommes de réseaux.

Définition 11 (Interrupteurs, adjacence et chemins dans $\mathbf{DLL}_0$)**.**
Soit p une structure à sauts. Une *Position d'interrupteurs de* p I_p est une fonction partielle de domaine $\mathbf{SA}(p)$ et de codomaine $\mathbf{SA}(p)$, telle que pour tout $t \in \mathbf{SA}(p)$, s'il existe s, s' tels que $t = ⅋(s, s')$, alors $I_p(t) \in \{s, s'\}$, et s'il existe $s_1, \ldots, s_k$ tels que $t = c(s_1, \ldots, s_k)$, alors $I_p(t) \in \{s_1, \ldots, s_k\}$.

La relation d'adjacence dans une structure p est relative à une position d'interrupteurs. Soient $t, s \in \mathbf{SA}(p)$ et I une position d'interrupteurs de p. On dit que t est *adjacent à* s pour la position I, et l'on note $t \sim_{t,s}^{p,I} s$ si :

- $t \in \mathcal{V}$ et $s = \overline{t}$.
- $t = ⅋(s, s')$ ou $t = ⅋(s', s)$, et $I(t) = s$.
- $t = \otimes(s, s')$ ou $t = \otimes(s', s)$.
- $t = c(s_1, \ldots, s_k)$, $I(t) = s$ et $s \in \{s_1, \ldots, s_k\}$.
- $t = \overline{c}(s_1, \ldots, s_k)$ et $s \in \{s_1, \ldots, s_k\}$.
- $t = d(s)$.
- $t = \overline{d}(s)$.
- $t = S_p(s)$.

et l'on note $t \sim_{\langle t|s\rangle}^{p,I} s$ si :

- $\langle t|s\rangle \in \mathbf{C}(p)$.

Pour deux étiquettes d'adjacence $l, l' \in (\mathbf{SA}(p) \times \mathbf{SA}(p)) \cup \mathbf{C}(p)$, on écrit $l \equiv l'$ si $l = l'$ ou si $l = (x, \overline{x})$ et $l' = (\overline{x}, x)$.

Les *chemins* et leurs attributs (chemins à rebours, extrémités) sont définis ensuite de la même façon que pour $\mathbf{MLL}^-$ (définition 3). En particulier, on rappelle que $\mathbf{ln}(\chi)$ désigne le nombre d'arbres qui composent le chemin χ.

2.3.1.3 Réductions

Définition 12 (Élimination usuelle des coupures de $\mathbf{DLL}_0$)**.**
On définit les règles d'élimination des coupures pour les structures, qui sont de domaine $\mathbf{DLL}_0$ et de codomaine $\mathbf{DLL}_0^+$. La définition de la substitution d'un arbre à une variable est la même qu'en définition 2 (section 2.2). On définit d'abord ces règles de façon locale, c'est-à-dire comme relations entre une coupure et une somme finie de suites de coupures :

— $\langle \otimes(t,t') | \parr (s,s') \rangle \to (\langle t|s\rangle, \langle t'|s'\rangle)$.
— $\langle d(t) | \overline{d}(s)\rangle \to (\langle t|s\rangle)$.
— $\langle \mathbf{a}_\kappa | \overline{\mathbf{a}}_\lambda\rangle \to \epsilon$.
— $\langle \mathbf{1}_\mu | \bot_\nu\rangle \to \epsilon$.
— $\langle c(t_1,\dots,t_n) | \overline{d}(s)\rangle \to \sum_{i=1}^n (\langle t_1|\overline{\mathbf{a}}_{\lambda_1}\rangle, \dots, \langle t_i|\overline{d}(s)\rangle, \dots, \langle t_n|\overline{\mathbf{a}}_{\lambda_n}\rangle)$
— $\langle \overline{c}(t_1,\dots,t_n) | d(s)\rangle \to \sum_{i=1}^n (\langle t_1|\mathbf{a}_{\kappa_1}\rangle, \dots, \langle t_i|d(s)\rangle, \dots, \langle t_n|\mathbf{a}_{\kappa_n}\rangle)$.
— $\langle c(t_1,\dots,t_n) | \mathbf{a}_\kappa\rangle \to (\langle t_1|\mathbf{a}_{\kappa_1}\rangle, \dots, \langle t_n|\mathbf{a}_{\kappa_n}\rangle)$.
— $\langle \overline{c}(t_1,\dots,t_n) | \overline{\mathbf{a}}_\lambda\rangle \to (\langle t_1|\overline{\mathbf{a}}_{\lambda_1}\rangle, \dots, \langle t_n|\overline{\mathbf{a}}_{\lambda_n}\rangle)$.
— $\langle d(t) | \overline{\mathbf{a}}_\lambda\rangle \to 0$.
— $\langle \overline{d}(t) | \mathbf{a}_\kappa\rangle \to 0$.
— $\langle c(t_1,\dots,t_n) | \overline{c}(s_1,\dots,s_m)\rangle \to$

$$(\langle t_1|\overline{c}(x_{1,1},\dots,x_{1,m})\rangle, \dots, \langle t_n|\overline{c}(x_{n,1},\dots,x_{n,m})\rangle,$$
$$\langle s_1|c(\overline{x}_{1,1},\dots,\overline{x}_{n,1})\rangle, \dots, \langle s_m|c(\overline{x}_{1,m},\dots,\overline{x}_{n,m})\rangle)$$

On étend la réduction aux structures en posant, si $c \to \sum_{i=1}^n (\overrightarrow{c}_i)$ par l'une des réductions définies ci-dessus, $(\overrightarrow{d}, c; \overrightarrow{t}) \to \sum_{i=1}^n (\overrightarrow{d}, \overrightarrow{c}_i; \overrightarrow{t})$.

On ajoute enfin les deux règles globales suivantes :

— $(\overrightarrow{c}, \langle x|t\rangle; \overrightarrow{s}) \to (\overrightarrow{c}; \overrightarrow{s})[t/\overline{x}]$.
— Si $p \to \sum_{i=1}^n p_i$, alors $p + \sum_{j=1}^m q_j \to \sum_{i=1}^n p_i + \sum_{j=1}^m q_j$.

Nous avons donné les règles générales de réduction, que nous nous apprêtons à étendre dans un cadre parallèle. Rappelons que le calcul qui suit à pour vocation la capacité de simuler l'élimination des coupures exponentielles dans le développement de Taylor.

À cette fin, il nous faut disposer d'une notion de réduction qui n'est pas seulement parallèle, mais à laquelle on autorise l'élimination de certaines coupures créées dans la foulée. On parle alors de réduction *à grand pas*.

Nous restreignons la forme des coupures possibles de façon à simplifier l'exposition : nous allons considérer à partir de maintenant que les cocontractions ont toujours pour sous-arbres immédiats des codérélictions. Pour cela nous introduisons le connecteur $!(t_1,\dots,t_n)$, qui est à comprendre comme $\overline{c}(\overline{d}(t_1),\dots,\overline{d}(t_n))$, et toujours comme un coaffaiblissement $\overline{\mathbf{a}}_\kappa$ pour le cas 0-aire. La forme gérérale de la réduction en l'absence de cette refonte ne pose pas de problème profond, mais serait noyée dans un formalisme moins digeste à seule fin de considérer des structures plus générales, qui n'apparaîtront pas dans notre étude du développement de Taylor.

Notre syntaxe pour les réseaux à ressources est alors restreinte aux atomes et aux connecteurs $\otimes, \parr, c, d, !$. Notons que ! n'est plus un connecteur symétrique à la contraction.

La définition de l'élimination des coupures que nous considérons pour la suite est en fait un cas particulier de la première. En effet, le lecteur peut vérifier que les réductions qui sont définies pour $!(t_1, \dots, t_n)$ correspondent à deux étapes de réduction parallèle pour $\overline{c}(\overline{d}(t_1), \dots, \overline{d}(t_n))$. C'est-à-dire que l'on élimine la coupure correspondant à la cocontraction, puis celle correspondant aux codérélictions, en une seule et même étape. La correspondance n'est pas exacte pour la coupure avec une contraction, car l'élimination de la coupure d'une contraction avec le nouveau connecteur ! efface tous les coaffaiblissements qui apparaîtraient si l'on remplaçait ! par une cocontraction et des codérélictions. Plus tard, nous étendrons notre notion de réseau de façon à ce que les (co)affaiblissements soient traités comme des éléments neutres de la (co)contraction (voir section 2.6.2). À cet égard, la correspondance entre la réduction pour $!(t_1, \dots, t_n)$ et pour $\overline{c}(\overline{d}(t_1), \dots, \overline{d}(t_n))$ sera complète.

En revanche, les chemins pour cette syntaxe sont exactement les mêmes que précédemment, en remplaçant $s = !(t_1, \dots, t_n)$ par $\overline{c}(\overline{d}(t_1), \dots, \overline{d}(t_n))$. C'est-à-dire que pour tout i, $t_i \sim_{t_i,s} s$.

Définition 13 (Élimination des coupures parallèle à grand pas dans $\mathbf{DLL}_0^+$)**.**

$$\langle !(t_1, \dots, t_n) | c(s_1, \dots, s_m) \rangle \to_{\langle !|c \rangle} \sum_{\substack{(\overrightarrow{\delta}_1, \dots, \overrightarrow{\delta}_m) \\ \in \Delta^m((t_1, \dots, t_n))}} \langle !(\overrightarrow{\delta}_1) | s_1 \rangle, \dots, \langle !(\overrightarrow{\delta}_m) | s_m \rangle$$

Où pour toute suite ordonnée d'arbres $(u_1, \dots, u_n)$, $\Delta^m((u_1, \dots, u_n))$ est l'ensemble des suites $(u'_{1,1}, \dots, u'_{1,k_1}), \dots, (u'_{m,1}, \dots, u'_{m,k_m})$ qui forment une partition de $(u_1, \dots, u_n)$ et telles que pour tout i, $(u'_{i,1}, \dots, u'_{i,k_i})$ est une sous-suite de $(u_1, \dots, u_n)$ (l'ordre relatif est respecté).

De plus, $!(\overrightarrow{\delta}_i) = \overline{\mathbf{a}}_{\lambda_i}$ si $\overrightarrow{\delta}_i = \epsilon$. Cette règle de réduction correspond, dans la réduction usuelle des réseaux différentiels, à l'itération des réductions contraction/cocontraction, axiome, et à l'absorbtion des coaffaiblissements par les cocontractions, menées en parallèle.

— $\langle !(t_1, \dots, t_n) | d(s) \rangle \to_{\langle !|d \rangle} \begin{cases} \langle t_1 | s \rangle \text{ si } n = 1 \\ 0 \text{ sinon} \end{cases}$

— $\langle !(t_1, \dots, t_n) | \mathbf{a}_\kappa \rangle \to_0 0$

— $\langle d(t) | \overline{\mathbf{a}}_\lambda \rangle \to_0 0$.

— $\langle \mathbf{a}_\kappa | \overline{\mathbf{a}}_\lambda \rangle \to_\epsilon \epsilon$.

— $\langle \mathbf{1}_\mu | \bot_\nu \rangle \to_\epsilon \epsilon$.

— $\langle c(t_1, \dots, t_n) | \overline{\mathbf{a}}_\lambda \rangle \to_{\langle c|\overline{\mathbf{a}} \rangle} (\langle t_1 | \overline{\mathbf{a}}_{\lambda_1} \rangle, \dots, \langle t_n | \overline{\mathbf{a}}_{\lambda_n} \rangle)$.

— $\langle \otimes(t, t') | \parr (s, s') \rangle \to_m (\langle t | s \rangle, \langle t' | s' \rangle)$.

On définit maintenant la réduction parallèle à grand pas dans $\mathbf{DLL}_0^+$.

— Pour tout p, $p \rightrightarrows p$, et pour tout $\alpha \in \{\mathsf{ax}, \epsilon, m, \langle !|c \rangle, \langle !|d \rangle, 0, \langle \mathbf{a}|c \rangle\}$, $p \rightrightarrows_\alpha p$.

— Soit $\alpha \in \{\epsilon, m, \langle !|c \rangle, \langle !|d \rangle, 0, \langle \mathbf{a}|c \rangle\}$ et $p \rightrightarrows (\overrightarrow{d}, c; \overrightarrow{t})$. Si $c \to_\alpha \sum_{i \in I} \overrightarrow{c}_i$, alors $p \rightrightarrows \sum_{i \in I} (\overrightarrow{d}, \overrightarrow{c}_i; \overrightarrow{t})$. Si, de plus, $p \rightrightarrows_\alpha (\overrightarrow{d}, c; \overrightarrow{t})$, alors $p \rightrightarrows_\alpha \sum_{i \in I} (\overrightarrow{d}, \overrightarrow{c}_i; \overrightarrow{t})$.

— Si $p \rightrightarrows (\overrightarrow{d}, \langle x|s \rangle; \overrightarrow{t})$, alors $p \rightrightarrows (\overrightarrow{d}; \overrightarrow{t})[s/\overline{x}]$. Si, de plus, $p \rightrightarrows_{\mathsf{ax}} (\overrightarrow{d}, \langle x|s \rangle; \overrightarrow{t})$, alors $p \rightrightarrows_{\mathsf{ax}} (\overrightarrow{d}; \overrightarrow{t})[s/\overline{x}]$

— Si $p \rightrightarrows \sum_{i\in I} p_i$, alors $p + \sum_{j\in J} q_j \rightrightarrows \sum_{i\in I} p_i + \sum_{j\in J} q_j$. Si, de plus, $p \rightrightarrows_\alpha \sum_{i\in I} p_i$, alors $p + \sum_{j\in J} q_j \rightrightarrows_\alpha \sum_{i\in I} p_i + \sum_{j\in J} q_j$, pour tout indice de réduction α mentionné ci-dessus.

On note que la réduction est définie sur les structures de $\mathbf{DLL}_0^+$, ce qui implique en particulier que lorsque la réduction crée de nouveaux arbres (c'est le cas pour $\rightarrow_{\langle c|\overline{\mathbf{a}}\rangle}$, et ça peut l'être pour $\rightarrow_{\langle !|c\rangle}$), alors de nouveaux noms sont introduits. Au cas contraire, le réduit ne satisferait pas les conditions de la définition 9.

La réduction doit être définie pour les structures à sauts, ce qui n'est pas évident ici. En effet, supposons que $p = (\langle \otimes(t,t')|\ ⅋\ (s,s')\rangle, \overrightarrow{c}; \overrightarrow{t})$, et que $S_p(\nu) = \otimes(t,t')$ pour une co-unité ν. Et bien si p se réduit à q en éliminant cette coupure, $S_q(\nu)$ doit être redéfini, car l'arbre vers lequel il pointait n'existe plus. On définit donc pour toutes les coupures éliminées, une redirection des sauts [4]. Soient donc (p, S_p) et q tels $p \rightarrow q$, examinons les différentes réductions possibles, et définissons S_q en fonction :

— Si $\langle \otimes(t_1,t_2)|\ ⅋\ (s_1,s_2)\rangle \in \mathbf{C}(p)$ est réduite, alors pour tout $\gamma \in \mathbf{U}_\perp \cup \mathbf{U}_?$ tel que $S_p(\gamma) = \otimes(t_1,t_2)$ (respectivement $⅋(s_1,s_2)$), on pose $S_q(\gamma) = t_1$ (respectivement s_1). S_q coïncide avec S_p pour les autres affaiblissements et co-unités.

— Si $\langle x|t\rangle$ est réduite, alors pour tout γ tel que $S_p(\gamma) \in \{x, \overline{x}\}$, on pose $S_q(\gamma) = t$.

— Si $\langle !(t)|(s)\rangle$ est réduite, alors pour tout γ tel que $S_p(\gamma) = !(t)$ (respectivement $d(s)$), on pose $S_q(\gamma) = t$ (respectivement, s).

— Si $\langle \mathbf{a}_\kappa|\overline{\mathbf{a}}_\lambda\rangle$ est réduite alors pour tout γ tel que $S_p(\gamma) \in \{\mathbf{a}_\kappa, \overline{\mathbf{a}}_\lambda\}$, on pose $S_q(\gamma) = S_p(\mathbf{a}_\kappa)$.

— Si $\langle c(t_1,\ldots,t_n)|\overline{\mathbf{a}}_\lambda\rangle$ est réduite, alors pour tout atome γ tel que $S_p(\gamma) = c(t_1,\ldots,t_n)$ (respectivement, $\overline{\mathbf{a}}_\lambda$), on pose $S_q(\gamma) = t_1$ (respectivement, $\overline{\mathbf{a}}_{\lambda_1}$).

— Si $\langle c(t_1,\ldots,t_n)|!(s_1,\ldots,s_m)\rangle$ est réduite, alors pour tout $(\overrightarrow{\delta}_1,\ldots,\overrightarrow{\delta}_m) \in \Delta^m(t_1,\ldots,t_n)$, et pour tout γ tel que $S_p(\gamma) = !(s_1,\ldots,s_m)$ (respectivement, $= c(t_1,\ldots,t_n)$), on pose $S_q(\gamma) = !(\overrightarrow{\delta}_1)$ (respectivement, $= s_1$).

Nous étudierons en détail les permutations et sommes relatives aux ensembles Δ^m quand nous en viendrons à traiter la simulation de l'élimination des coupures de **MELL**, en section 2.6. Dans la présente section, nous nous concentrerons sur la forme des arguments des sommes impliquées. L'étude quantitative des réductions en sommes viendra à temps pour préparer l'examen de combinaisons linéaires de réseaux, avec coefficients.

Définition 14.
Soit p une structure simple de $\mathbf{DLL}_0$. On définit sa taille, notée $\mathbf{taille}(p)$.

— Pour tout arbre atomique $t \in \mathbf{A}, \mathbf{taille}(t) = 1$.

— $\mathbf{taille}(\alpha(t_1,\ldots,t_n)) = (\sum_{i=1}^n \mathbf{taille}(t_i)) + 1$, pour $\alpha \in \{\otimes, ⅋, !, c, \overline{d}, c\}$.

— $\mathbf{taille}(\langle t|s\rangle) = \mathbf{taille}(t) + \mathbf{taille}(s)$.

— $\mathbf{taille}\,((c_1,\ldots,c_k;t_1,\ldots,t_l)) = \sum_{i=1}^k \mathbf{taille}(c_i) + \sum_{j=1}^l \mathbf{taille}(t_j)$.

4. Attirons l'attention du lecteur sur le fait que cette redirection revêt un aspect arbitraire, et qu'un choix est fait quand un saut remonte sur une prémisse d'un arbre donné par exemple.

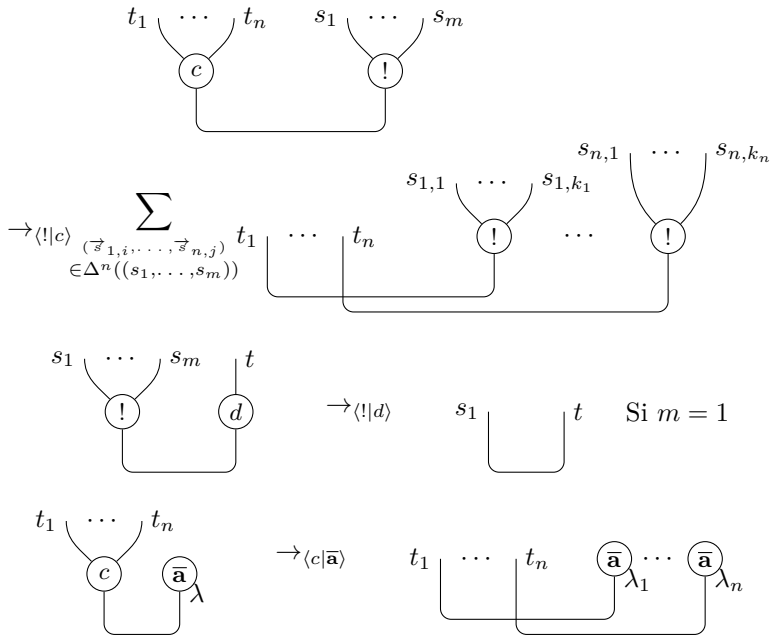


FIGURE 2.17 – Élimination des coupures des réseaux à ressources

Propriété 2 (Standardisation de la réduction). Soient p et q deux structures de $\mathbf{DLL}_0$ telles que $p \rightrightarrows q$. Pour toute suite d'indices $(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \alpha_7) \in \{\sigma(\mathsf{ax}, m, \langle\overline{\mathbf{a}}|c\rangle, \langle !|c\rangle, \langle !|d\rangle, 0, \epsilon) \mid \sigma \in \mathfrak{S}_7\}$, il existe $p_1, \ldots, p_6$ tels que $p \rightrightarrows_{\alpha_1} p_1 \rightrightarrows_{\alpha_2} p_2 \rightrightarrows_{\alpha_3} p_3 \rightrightarrows_{\alpha_4} p_4 \rightrightarrows_{\alpha_5} p_5 \rightrightarrows_{\alpha_6} p_6 \rightrightarrows_{\alpha_7} q$

Démonstration. De la même façon que pour la preuve du lemme 1, il suffit de remarquer que pour deux indices de réduction α, α', si $p \to_\alpha p_1 \to_{\alpha'} q$, il existe p_1' tel que $p \to_{\alpha'} p_1' \to_\alpha q$. L'argument s'étend facilement à la réduction parallèle. □

À nouveau, cette propriété va nous permettre d'examiner les propriétés de la réduction en étudiant séparément différentes phases :

— Phase dite du *déjà vu*. Elle correspond aux réductions $\rightrightarrows_{\mathsf{ax}}$ et $\rightrightarrows_m$, que nous avons déjà vues en section 2.2.

— Phase *persistante*. Elle correspond aux réductions $\rightrightarrows_{\langle\overline{\mathbf{a}}|c\rangle}$, $\rightrightarrows_{\langle !|c\rangle}$, et $\rightrightarrows_{\langle !|d\rangle}$, qui engendrent de nouvelles coupures dans la structure réduite.

— Phase *évanescente*. Elle correspond aux réductions $\rightrightarrows_\epsilon$, qui sont des parties du réseau de départ qui disparaissent sans rien laisser derrière elles. Cette phase est traitée à l'aide des mécanismes de sauts.

Nous traitons dans un premier temps les réductions persistantes, puis les réductions évanescentes. Les réductions de la phase du déjà vu seront incorporées au cas général, car il s'agira essentiellement d'importer les résultats de la section 2.2 aux nouvelles structures.

Pour les examens que nous nous apprêtons à pratiquer, il nous faut considérer les strutures simples apparaissant dans les réduits de structures simples. C'est-à-dire que nous nous concentrerons sur les formes des structures sans considérer les sommes dans lesquelles elles peuvent apparaître. Plus précisément :

Définition 15.
Soit p une structure simple de $\mathbf{DLL}_0$. Si $p \rightrightarrows q + \sum_{i\in I} q_i$, alors on écrit $p \gg q$. Si de plus, $p \rightrightarrows_\alpha q$ pour un indice de réduction α, on écrit $p \gg_\alpha q$.

2.3.2 Combinatoire des réductions persistantes à grands pas

Dans cette partie, nous allons établir deux résultats centraux de notre étude. D'abord, le fait que si $p \gg q$ par une réduction persistante, alors on peut borner la longueur des chemins de q en fonction de la longueur des chemins de p. La réduction des réseaux à ressources étant similaire à plusieurs égards à celle de $\mathbf{MLL}^-$, nous pourrons utiliser les méthodes introduites pour les résultats de la section 2.2.2.2

Ensuite, nous caractériserons l'évolution de la taille des structures pour cette phase de réductions.

Dans cette section 2.3.2, nous n'écrirons $p \gg q$ que lorsque les réductions de p à q sont persistantes.

2.3.2.1 Évolution des chemins sous réduction persistante

Nous devons caractériser ici la forme possible que peuvent prendre les chemins dans les structures de $\mathbf{DLL}_0$. Nous allons en fait étendre la notion de résidu et de nœud coulant, introduits pour $\mathbf{MLL}^-$, de façon à obtenir également les résultats souhaités.

On peut remarquer en premier lieu que lorsque $p \gg q$, les sous-arbres de q sont *presque* inclus dans ceux de p. Presque, car la réduction $\rightrightarrows_{\langle \overline{\mathbf{a}}|c\rangle}$ crée des coaffaiblissements sur son passage, et la réduction $\rightrightarrows_{\langle !|c\rangle}$ crée de nouvelles cocontractions (mais qui n'ont pas de nouveaux sous-arbres).

Définition 16 (Coupures résiduelles et nœuds coulants dans $\mathbf{DLL}_0$)**.**
Soient p, q deux structures simples de $\mathbf{DLL}_0$ telles que $p \gg_{\langle !|c\rangle} q$, et $c = \langle !(t_1, \ldots, t_n)|c(s_1, \ldots, s_m)\rangle \in \mathbf{C}(p)$. Si c est réduite, alors la réduction donne les m coupures $d_1 = \langle !(\overrightarrow{\delta}_1)|s_1\rangle, \ldots, d_m = \langle !(\overrightarrow{\delta}_m)|s_m\rangle \in \mathbf{C}(q)$. Dans ce cas, $d_1, \ldots, d_n$ sont appelées *coupures résiduelles* de c.

Si $\xi \in \mathbf{Ch}(q)$ peut s'écrire $\chi_1, d_i, \chi_2, d_j, \chi_3$ pour d_i et d_j deux coupures résiduelles d'une même coupure c de p, alors la paire d_i, d_j est appelée un *nœud coulant* de ξ.

Pour le lemme suivant, il nous faut considérer les chemins sous une légère variante, que l'on appellera *chemins longs* :

Définition 17.
Soient p une structure, et $\xi \in \mathbf{Ch}(p)$. On dit que ξ est un *chemin long* si les extrémités de ξ ne sont pas des cocontractions qui apparaissent dans une coupure de p. On note $\mathbf{Ch}^\bullet(p)$ les chemins longs de p.

Remarque 4. La propriété première des chemins longs, est que tout chemin est sous-chemin d'un chemin long. En effet, soit $\xi = \chi, t$ un chemin tel que $t = !(t_1, \dots, t_n)$ apparaisse dans une coupure $\langle t|s\rangle$. Si $\chi = \chi', t_i$ pour $i \in \{1, \dots, n\}$, alors on peut prolonger ξ en ξ, s. Si $\chi = \chi', s$, on peut prolonger ξ en ξ, t_i. Si $\chi = \chi', \mathbf{a}_\kappa$ avec $S(\kappa) = t$, on peut prolonger ξ par s ou par t_i. On vérifie aisément que ces prolongements n'ont pas d'incidence sur les interrupteurs.

La deuxième propriété des chemins longs, celle qui a motivé leur introduction, est que si $p \gg q$ et que $\xi \in \mathbf{Ch}^\bullet(q)$, alors les extrémités de ξ sont dans $\mathbf{SA}(p)$, ou sont des coaffaiblissements. En effet, les arbres de $\mathbf{SA}(q)$ qui ne sont pas dans $\mathbf{SA}(p)$ sont précisément les cocontractions qui sont créées par la réduction $\rightrightarrows_{\langle !|c\rangle}$, et les coaffaiblissements créés par la réduction $\rightrightarrows_{\langle \overline{\mathbf{a}}|c\rangle}$.

Lemme 11. Soient p et q deux structures simples de $\mathbf{DLL}_0$ telles que $p \gg q$ par réductions persistantes. Pour tout $\xi \in \mathbf{Ch}^\bullet(q)$, il existe $\xi^- \in \mathbf{Ch}(p)$ de mêmes extémités que ξ, à renommage des coaffaiblissements près.

Démonstration. La définition de ξ^- est par cas sur la réduction, puis par induction sur le nombre de coupures rédiduelles dans ξ. Par la propriété 2, et car nous ne considérons que les réductions persistantes, il existe p_1, p_2 tels que $p \gg_{\langle !|d\rangle} p_1 \gg_{\langle \overline{\mathbf{a}}|c\rangle} p_2 \gg_{\langle !|c\rangle} q$. Nous pouvons donc raisonner par cas, en associant aux chemin de q des chemins dans p_2 par une transformation $()^{-\langle !|c\rangle}$, aux chemins dans p_2 des chemins dans p_1 par une transformation $()^{-\langle \overline{\mathbf{a}}|c\rangle}$, et aux chemins dans p_1 des chemins dans p par une transformation $()^{-\langle !|d\rangle}$. On pourra alors conclure par composition de ces transformations. On aura soin de veiller à ce que, partant de chemins longs, on obtienne toujours un chemin long par ces transformations.

On commence donc par montrer qu'il existe dans p_2 un chemin ξ_2 de mêmes extrémités que ξ. Comme la coupure contraction/cocontraction est celle qui crée des nœuds coulants, la situation est similaire à celle de la réduction multiplicative. Si ξ ne comporte pas de résidu, alors ξ peut tout de même contenir des sous-arbres qui ne sont pas dans p_2. C'est le cas si $\xi = \chi_1, !(t_1, \dots, t_n), \chi_2$, tel que la cocontraction soit issue d'une réduction $\langle !(\overrightarrow{t})|c(\overrightarrow{s})\rangle \gg_{\langle !|c\rangle} (\langle !(t_1, \dots, t_n)|s_1\rangle, \overrightarrow{d})$. Dans ce cas, les s_i n'apparaissent pas dans ξ (sinon l'on aurait un résidu), et il passe par la cocontraction, par les t_j, ou par des sauts. alors on pose $\xi^{-\langle !|c\rangle} = \chi_1^{-\langle !|c\rangle}, !(\overrightarrow{t}), \chi_2^{-\langle !|c\rangle}$. Comme ξ est un chemin long, χ_1 ni χ_2 n'est vide, et les extrémités de ξ sont bien dans p_2, par la remarque 4. Voir figure 2.18 Notons que ξ^- est toujours un chemin long.

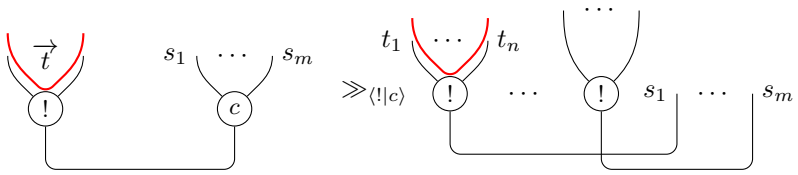


FIGURE 2.18 – ξ n'a pas de résidus, mais contient de nouvelles cocontractions.

Supposons que $\xi = \chi_1, d_1, \chi_2, d_2, \chi_3$, pour d_1 et d_2 deux résidus dans q de $c \in \mathbf{C}(p_2)$, et *s.p.g.* qu'aucune coupure $c \neq c'$ n'a de résidu dans χ_1 et dans χ_3. On peut poser $c = \langle !(t_1, \dots, t_n)|c(s_1, s_2, \dots, s_m)\rangle$ et $d_1 = \langle !(\overrightarrow{\delta}_1)|s_1\rangle, d_2 = \langle \overline{c}(\overrightarrow{\delta}_2)|s_2\rangle$ pour $\overrightarrow{\delta}_1$ et $\overrightarrow{\delta}_2$ des sous suites de $(t_1, \dots, t_n)$. Nous n'avons fait que

supposer un ordre sur les suites d'arbres pour simplifier la lecture. Naturellement, toutes les coupures résiduelles et tous les nœuds coulants sont de cette forme, à permutation des sous-arbres près.

Maintenant, il suffit de remarquer que $\xi = \chi_1, !(\overrightarrow{\delta}_1), s_1, \chi_2, s_2, !(\overrightarrow{\delta}_2), \chi_3$. On se réfère à la preuve du lemme 3 pour remarquer que les autres ordonnancements des coupures d_1 et d_2 dans ξ sont impossibles. Comme ξ est un chemin long, χ_1 et χ_3 ne sont pas vides, on suppose alors $\chi_1 = \chi_1', u_1$ et $\chi_3 = u_3, \chi_3'$. On remarque que u_1 est soit dans $\overrightarrow{\delta}_1$ soit dans $\mathbf{U}_\perp \cup \mathbf{U}_?$ (et dans ce dernier cas, $S_{p_2}(u_3) = !(t_1, \dots, t_n)$), et que u_3 est nécessairement dans $\overrightarrow{\delta}_2$, car lors de l'élimination des coupures, les sauts ne sont redirigés que vers le premier arbre créé $!(\overrightarrow{\delta}_1)$. Si $!(\overrightarrow{(}\delta_2))$ était ce premier arbre, la situation serait symétrique. On peut donc écrire $\xi^{-\langle \overline{c}|c\rangle} = \chi_1^{-\langle \overline{c}|c\rangle}, !(t_1, \dots, t_n), \chi_3^{-\langle \overline{c}|c\rangle}$ dans tous les cas.

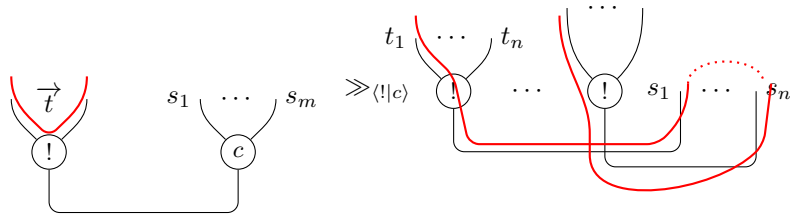


FIGURE 2.19 – ξ comporte au moins un nœud coulant

Si maintenant on suppose que ξ ne comporte jamais plus d'une coupure résiduelle, alors soit $c = \langle !(t_1, \dots, t_n)|c(s_1, \dots, s_m)\rangle \in \mathbf{C}(p_2)$ tel que $\xi = \chi_1, d, \chi_2$, en supposant à nouveau, sans perte de généralité, que $d = \langle !(\overrightarrow{\delta}_1)|s_1\rangle$ soit un résidu de c dans ξ. On pose alors $\xi^{-\langle !|c\rangle} = \chi_1^{-\langle !|c\rangle}, c, \chi_2^{-\langle !|c\rangle}$, car à nouveau, comme ξ est un chemin long, $\chi_1 = \chi_1', u_1$ avec $u_1 \in \overrightarrow{\delta}_1 \cup \mathbf{U}_? \cup \mathbf{U}_\perp$ et $\chi_3 = u_3, \chi_3'$ avec $u_3 \in \overrightarrow{\delta}_2$.

Examinons maintenant les autres réductions, qui ne présentent pas de comportement similaire à celui des nœuds coulants. Soit $\xi \in \mathbf{Ch}(p_2)$ (on rappelle que $p_1 \gg_{\langle \overline{\mathbf{a}}|c\rangle} p_2$). Si ξ ne comporte pas de résidu de coupure de p, alors soit les extrémités de ξ sont dans $\mathbf{SA}(p_1)$, et on pose $\xi^{-\langle \overline{\mathbf{a}}|c\rangle} = \xi$, soit $\xi = \chi, \overline{\mathbf{a}}_{\lambda_1}$ (l'argument est symétrique si $\xi = \overline{\mathbf{a}}_{\lambda_1}, \chi$) avec $\overline{\mathbf{a}}_{\lambda_1} \notin \mathbf{SA}(p)$. C'est le cas si $\mathbf{a}_{\lambda_1}$ est un coaffaiblissement créé par la réduction dans p d'une coupure $\langle c(t_1, \dots, t_n)|\overline{\mathbf{a}}_\lambda\rangle \gg_{\langle \overline{\mathbf{a}}|c\rangle} (\langle t_1|\overline{\mathbf{a}}_{\lambda_1}\rangle, \dots, \langle t_n|\overline{\mathbf{a}}_{\lambda_n}\rangle)$ et si $\chi = \chi', \gamma$, pour $\gamma \in \mathbf{U}_\perp \cup \mathbf{U}_?$ tel que $S_{p_2}(\gamma) = \overline{\mathbf{a}}_{\lambda_1}$. Et c'est le cas seulement dans cette configration, car le chemin ne comportant pas de résidu, il ne peut passer par le coaffaiblissement à travers ume coupure $\langle t_i|\overline{\mathbf{a}}_{\lambda_i}\rangle$. Et ce dernier n'ayant pas de sous-arbre, il n'est donc possible d'y accéder que par un saut. Dans ce cas donc, on pose $\xi^{-\langle \overline{\mathbf{a}}|c\rangle} = \chi^{-\langle \overline{\mathbf{a}}|c\rangle}, \overline{\mathbf{a}}_\lambda$ (car $S_{p_1}(\gamma) = \overline{\mathbf{a}}_\lambda$).

Sinon, il existe $c = \langle c(t_1, \dots, t_n)|\overline{\mathbf{a}}_\lambda\rangle \in \mathbf{C}(p_1)$, et $i \in \{1, \dots, n\}$ tel que $d = \langle t_i|\overline{\mathbf{a}}_{\lambda_i}\rangle \in \mathbf{C}(p_2)$ apparaît dans ξ. Supposons donc que $\xi = \chi_1, t_i, \overline{\mathbf{a}}_{\lambda_i}, \chi_2$. On remarque à nouveau que si χ_2 n'est pas vide, alors $\chi_2 = \gamma, \chi_2'$ avec $S_{p_2}(\gamma) = \overline{\mathbf{a}}_{\lambda_i}$. Dans ce cas, on pose $\xi^{-\langle \overline{\mathbf{a}}|c\rangle} = \chi_1^{-\langle \overline{\mathbf{a}}|c\rangle}, t_i, c(t_1, \dots, t_n), \overline{\mathbf{a}}_\lambda, \chi_2^{-\langle \overline{\mathbf{a}}|c\rangle}$. En effet, $S_{p_1}(\gamma) = \overline{\mathbf{a}}$, par définition de la redirection des sauts, ce qui en fait bien un chemin de p_1, comme illustré en figure 2.20.

Soit $\xi \in \mathbf{Ch}(p_1)$ (on rappelle que $p \gg_{\langle !|d\rangle} p_1$). Si ξ ne comporte aucun résidu

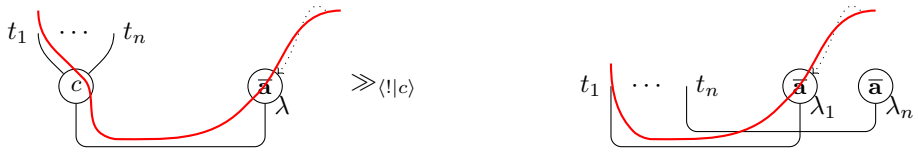


FIGURE 2.20 – ξ passe par un saut et un nouveau coaffaiblissement

d'une coupure de p, alors des sauts peuvent néanmoins avoir été redéfinis entre p et p_1. Si par exemple, $\xi = \chi_1, \gamma, t, \chi_2$, avec $S_{p_1}(\gamma) = t$, $\langle !(t)|d(s)\rangle \in \mathbf{C}(p)$ et $S_p(\gamma) = !(t)$), alors ξ n'est pas un chemin de p. On pose dans ce cas $\xi^{-\langle !|d\rangle} = \chi_1^{-\langle !|d\rangle}, \gamma, !(t), t, \chi_2^{-\langle !|d\rangle}$ si $\chi_2^{-\langle !|d\rangle}$ ne commence pas par un atome γ' dont le saut pointe sur $!(t)$, car dans ce cas on pose $\xi^{-\langle !|d\rangle} = \chi_1^{-\langle !|d\rangle}, \gamma, !(t), \chi_2^{-\langle !|d\rangle}$ On procède de même si $S_p(\gamma) = d(t)$.

Sinon, il existe $c = \langle !(t))|d(s)\rangle$ telle que $d = \langle t|s\rangle \in \mathbf{C}(p_1)$ apparaît dans ξ. Si donc $\xi = \chi_1, t, s, \chi_2$, alors on pose $\xi^{-\langle \bar{c}|d\rangle} = \chi_1^{-\langle !|d\rangle}, t, !(t), d(s), s, \chi_2^{-\langle !|d\rangle}$ si χ_1 ne termine pas par un atome γ dont le saut pointe vers $!(t)$ (respectivement, si χ_2 ne commence pas par un γ' dont le saut pointe vers $d(s)$), auquel cas $\xi^{-\langle \bar{c}|d\rangle}$ commence par $\chi_1^{-\langle !|d\rangle}, !(t)$ (respectivement, finit par $d(s), \chi_2^{-\langle !|d\rangle}$), sinon le chemin est mal défini car devant passer deux fois par $!(t)$ ou par $d(s)$.

Enfin, on peut poser pour tout $\xi \in \mathbf{Ch}(q)$, $\xi^- = \left(((\xi)^{-\langle !|c\rangle})^{-\langle \bar{\mathbf{a}}|c\rangle}\right)^{-\langle !|d\rangle}$ qui respecte les propriétés voulues. □

Le lemme qui précède va nous permettre de caractériser la forme des chemins d'une structure donnée en fonction de ceux de sa structure antiréduite. Nous allons établir que les chemins peuvent s'écrire comme des successions de nœuds coulants qui ne s'emmêlent pas, à la façon de la preuve du lemme 4. Quelques ajustements sont à apporter toutefois, notamment car les coupures de $\mathbf{DLL}_0$ peuvent avoir plus de deux résidus.

De cette façon, nous pourrons importer un résultat central de la section 2.2, à savoir le théorème 1 en montrant que $\mathbf{ln}(q)$ est borné par $f_{\mathbf{ln}}(\mathbf{ln}(p))$ quand $p \gg q$.

Mais si nous ne voulons pas inutilement multiplier cette borne par elle-même en passant à la réduction générale, nous devons établir la caractérisation et la borne pour les nœuds coulants issus de réductions multiplicatives et persistantes. En effet, plusieurs nœuds des différentes sortes présents sur un chemin incrémentent $\mathbf{ln}()$ de la même façon, il est donc plus raisonnable de considérer les deux types de nœuds sur un même chemin. On parlera d'ailleurs de *nœuds coulants multiplicatifs ou différentiels* selon s'ils proviennent d'une coupure $\otimes/\parr$ ou d'une coupure $c/!$. Cette généralisation ne complique par ailleurs pas nos arguments dans la mesure où l'essentiel du résultat a déjà été établi, et qu'il s'agit ici de montrer qu'il s'applique aussi bien aux deux réductions.

Dans le lemme qui suit, on utilise le fait que la construction $()^-$ soit définie pour les réductions différentielles et multiplicatives. Si $p \rightrightarrows_m p' \gg q$ et si $\xi \in \mathbf{Ch}(q)$, alors on écrira abusivement ξ^- pour désigner la composition de la construction du lemme 3 et de celle du lemme 11 sur ξ. Une simple vérification permet d'établir que cette composition est bien définie, et donne bien un chemin dans p de mêmes extrémités que ξ à renommage des affaiblissements près.

Lemme 12. Soient p, q tels que q soit obtenu par réduction parallèle multiplicatives et persistante de p (*i.e* il existe p' tel que $p \rightrightarrows_m p' \gg q$), et $\xi \in \mathbf{Ch}(q)$. Il existe une façon d'écrire ce chemin de la façon suivante :

$$\xi = \zeta_1, c_1, \chi_1, c'_1, \ldots, \zeta_k, c_k, \chi_k, c'_k, \zeta_{k+1}$$

où chaque paire (c_i, c'_i) est un nœud coulant multiplicatif ou différentiel, où les ζ_i ne comportent pas de nœud coulant, et où tout nœud coulant de ξ est soit l'un des (c_i, c'_i), soit un nœud coulant de l'un des χ_i.

Démonstration. Remarquons déjà qu'une coupure de p ne peut pas avoir plus de deux résidus dans q. En effet, si le chemin $\xi = \chi_1, d_1, \chi_2, d_2, \chi_3$, où $(d_1 = \langle \overline{c}(\overline{d}(t_1), \overrightarrow{\delta}_1)|s_1\rangle, d_2 = \langle !((t_2), \overrightarrow{\delta}_2)|s_2\rangle)$ est un nœud coulant différentiel issu d'une coupure $c = \langle !(t_1, \ldots, t_n)|c(s_1, \ldots, s_m)\rangle$, on a établi lors des lemmes précédents que dans ce cas $\xi = \chi_1, !(t_1, \overrightarrow{\delta}_1), s_1, \chi_2, s_2, !(t_2, \overrightarrow{\delta}_2), \chi_3$. On peut vérifier aisément que si c comportait un troisième résidu dans ξ, on aboutirait à une contradiction, ne pouvant ordonner les arbres de façon convenable.

Ensuite, on peut constater qu'un nœud coulant différentiel (c, c') et un nœud coulant multiplicatif (d, d') ne peuvent pas s'emmêler entre eux, c'est-à-dire que l'on n'a pas de chemin de la forme $c, \chi_1, d, \chi_2, c', \chi_3, d'$. En effet, on remarque à nouveau grâce à la construction $()^-$, que cela entraînerait l'existence d'un cycle dans p, en suivant exactement la preuve du lemme 4.

Les points ci-dessus nous permettent de conclure, car on a donné une description suffisante des chemins de q pour s'assurer qu'ils respectent toujours les propriétés énoncées. □

Théorème 3. Soient p, q tels que $p \rightrightarrows_m p' \gg q$, pour une structure p'. $\mathbf{ln}(q) \leq f_{\mathbf{ln}}(\mathbf{ln}(p))$.

Démonstration. La caractérisation du lemme 12 nous permet de répéter exactement l'étude de cas de la preuve du lemme 6 dans le cas où $p \rightrightarrows_m p' \gg_{\langle !|c\rangle} q$. Nous nous épargnons cette redite, et nous concentrons sur les autres réductions, à savoir $\gg_{\langle !|d\rangle}$ et $\gg_{\langle \overline{\mathbf{a}}|c\rangle}$.

Si donc $p \gg_{\langle !|d\rangle} p_1 \gg_{\langle \overline{\mathbf{a}}|c\rangle} q$, il suffit de vérifier, en revenant aux transformations de chemins de la preuve du lemme 11, que si $\xi \in \mathbf{Ch}(q)$, alors $\xi^{-\langle c|\overline{\mathbf{a}}\rangle}$ est au moins aussi long que ξ. Même observation pour $\chi^{-\langle !|d\rangle}$ si $\chi \in \mathbf{Ch}(p_1)$. On en déduit immédiatement que $\mathbf{ln}(q) \leq \mathbf{ln}(p)$

On conclut en rassemblant les points ci-dessus, pour le cas général où $p \rightrightarrows_m p_1 \gg_{\langle !|c\rangle} p_2 \gg_{\langle !|d\rangle} p_3 \gg_{\langle \overline{\mathbf{a}}|c\rangle} q$. □

2.3.2.2 Évolution de la taille sous réduction persistante

tâchons maintenant à borner la taille des antiréduits sous réduction persistante.

Lemme 13. Soient p, q tels que $p \rightrightarrows_{\langle !|d\rangle} q$. $\mathbf{taille}(p) \leq 2\mathbf{taille}(q)$.

Démonstration. Remarquons déjà que si $c \to_{\langle !|d\rangle} c'$, $c = \langle d(t)|!(s)\rangle$ et $c' = \langle t|s\rangle$. Donc pour toute coupure $c \to_{\langle !|d\rangle} c'$, $\mathbf{taille}(c) = \mathbf{taille}(c') + 2$. Par ailleurs, toute coupure est de taille au moins deux. Donc, posons $p = (c_1, \ldots, c_n, \overrightarrow{d}; \overrightarrow{t})$ et $q = (c'_1, \ldots, c'_n, \overrightarrow{d}; \overrightarrow{t})$, où $c_i \to_{\langle !|d\rangle} c'_i$ pour tout i. On a bien $\mathbf{taille}(p) \leq 2\mathbf{taille}(q)$. □

Lemme 14. Soient p, q tels que $p \gg_{\langle !|c\rangle} q$. $\mathbf{taille}(p) \leq 2\mathbf{taille}(q)$.

Démonstration. Commençons par mesurer la taille des coupures. $\mathbf{taille}(c = \langle !(t_1, \ldots, t_n)|c(s_1, \ldots, s_m)\rangle) = \sum_{i=1}^{n} \mathbf{taille}(t_i) + \sum_{j=1}^{m} \mathbf{taille}(s_j) + 2$. Or, $c \rightarrow_{\langle !|c\rangle} \overrightarrow{d} = (\langle !(\overrightarrow{\delta}_1)|s_1\rangle, \ldots, \langle !(\overrightarrow{\delta}_m)|s_m\rangle)$, pour une certaine suite $(\overrightarrow{\delta}_1, \ldots, \overrightarrow{\delta}_m) \in \Delta^m((t_1, \ldots, t_n))$. De plus, $\mathbf{taille}(\overrightarrow{d}) = \sum_{i=1}^{n} \mathbf{taille}(t_i) + \sum_{j=1}^{m} \mathbf{taille}(s_j) + m$.

Donc, $\mathbf{taille}(c) \leq \mathbf{taille}(\overrightarrow{d}) + 1$, car m est non nul (sinon $c(s_1, \ldots, s_m)$ serait un affaiblissement, et dans ce cas il n'y a pas dé réduit car n est également non nul, étant donné que les coàffaiblissements seront traités à part). Comme $p \gg_{\langle !|c\rangle} q$, $p = (c_1, \ldots, c_k, \overrightarrow{c}; \overrightarrow{t})$ et $q = (\overrightarrow{d}_1, \ldots, \overrightarrow{d}_k, \overrightarrow{c}; \overrightarrow{t})$, avec $c_i \rightarrow_{\langle !|c\rangle} \overrightarrow{d}_i$ pour tout i. En raisonnant comme précédemment, on obtient $\mathbf{taille}(p) \leq \mathbf{taille}(q) + k$, et naturellement, $\mathbf{taille}(q) \geq k$, donc on conclut. □

Lemme 15. Soient $p \gg_{\langle \overline{\mathbf{a}}|c\rangle} q$. $\mathbf{taille}(p) \leq \frac{3}{2}\mathbf{taille}(q)$.

Démonstration. Soit $c = \langle c(t_1, \ldots, t_n)|\overline{\mathbf{a}}_\lambda\rangle$, la réduction donne : $c \rightarrow_{\langle c|\overline{\mathbf{a}}\rangle} \overrightarrow{c} = (\langle t_1|\overline{\mathbf{a}}_{\lambda_1}\rangle, \ldots, \langle t_n|\overline{\mathbf{a}}_{\lambda_n}\rangle)$. $\mathbf{taille}(c) \leq \sum_{i=1}^{n} \mathbf{taille}(t_i) + 2$. Les coaffaiblissements étant naturellement de taille 1, on a $\mathbf{taille}(\overrightarrow{c}) = \sum_{i=1}^{n} \mathbf{taille}(t_i) + n$. Comme $n > 0$, on a bien $\mathbf{taille}(c) \leq \frac{3}{2}\mathbf{taille}(\overrightarrow{c})$.

Comme $p \gg_{\langle \overline{\mathbf{a}}|c\rangle} q$, $p = (c_1, \ldots, c_k, \overrightarrow{c}; \overrightarrow{t})$ et $q = (\overrightarrow{d}_1, \ldots, \overrightarrow{d}_k, \overrightarrow{c}; \overrightarrow{t})$, avec $c_i \rightarrow_{\langle c|\overline{\mathbf{a}}\rangle} \overrightarrow{d}_i$ pour tout i. En raisonnant comme précédemment on conclut bien $\mathbf{taille}(p) \leq \frac{3}{2}\mathbf{taille}(q)$. □

On peut maintenant établir un premier résultat sur l'évolution de la taille dans les réseaux à ressources, en supposant ici que les coupures évanescentes ne sont pas réduites.

Lemme 16. Soient p, q deux structures simples de $\mathbf{DLL}_0$ telles que $p \gg q$ (par réductions persistantes), $\mathbf{taille}(p) \leq 6(\mathbf{taille}(q))$.

Démonstration. Par combinaison des lemmes 13, 14, 15, et par standardisation, grâce à la propriété 2. □

2.3.3 Évanescence et sauts

Nous considérons maiteneant la réduction parallèle évanescente, c'est-a-dire le cas où $p \rightrightarrows_\epsilon q$, pour p et q deux structures simples à sauts. Notons que l'on n'a pas besoin ici de considérer une réduction dans le support (comme $\gg$, précédemment) dans la mesure où $\rightrightarrows_\epsilon$ ne génère pas de sommes.

Lors des argumemts précédents, nous avons pu borner la taille d'une structure p par une fonction de la taille de son réduit q et d'une mesure sur p, à savoir $\mathbf{ln}(p)$. La situation est différente ici, car dans le cas d'une réduction évanescente, $p = (c_1, \ldots, c_n, \overrightarrow{d}; \overrightarrow{t}) \rightrightarrows_\epsilon q = (\overrightarrow{d}; \overrightarrow{t})$ si les c_i sont des coupures évanescentes ($\langle \mathbf{1}_\mu|\bot_\nu\rangle$ ou $\langle \mathbf{a}_\kappa|\overline{\mathbf{a}}_\lambda\rangle$).

Les coupures évanescentes étant de taille 2, on a $\mathbf{taille}(p) = \mathbf{taille}(q) + 2n$. On a ajouté une constante à la taille du réduit, ce qui n'est pas une opération linéaire et nous demande de réfléchir différemment. Nous devons rajouter de la structure pour former une nouvelle mesure qui permettra de contrôler ce comportement problématique, comme on a contrôlé les contractions de chaînes d'axiomes grâce à $\mathbf{ln}(p)$.

Nous allons utiliser le fait que nos structures sont munies de sauts. L'idée est de mesurer dans l'antiréduit p le nombre de sauts qui peuvent pointer sur le même sous-arbre t de p. On appelle ce nombre le *degré de saut* de t, et on l'étend à p en prenant le degré de saut maximal de l'un de ses sous-arbres. Comme toutes les coupures évanescentes pointent toutes sur un sous-arbre de p, on peut multiplier le nombre de sous arbres dans q par le degré de saut de p, ainsi, si l'on pose $p = (c_1, \ldots, c_n; x, \overline{x}) \rightrightarrows_\epsilon q = (; x, \overline{x})$, et que l'on suppose par exemple que le degré de p est égal à 2, alors on sait que p comportait au plus 4 affaiblissements ou unités dont le saut était dirigé vers un arbre de q. On peut donc supposer que p est égal à $(c_1, \ldots, c_4; x, \overline{x})$ comme en figure 2.21, où l'on omet les indices des atomes pour plus de lisibilité et où la fonction de saut est représentée par les flèches pointillées.

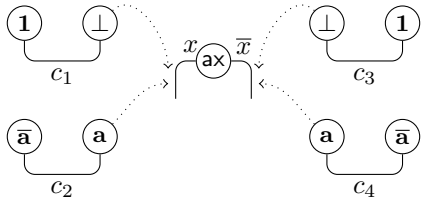


FIGURE 2.21 – Exemples de sauts pour une structure de degré 2.

Si l'on multiplie alors le nombre de sous-arbres de q, 2, par deux fois le degré de p, 4, on obtient bien le nombre de sous arbres qui ont été effacés par la réduction, et cela nous permet de borner la taille de p.

Seulement, la situation est encore un peu plus délicate, car rien n'interdit à la fonction de saut de se diriger vers un sous-arbre d'une coupure évanescente. Par exemple, en figure 2.22, la structure a toujours un degré de saut de 2, mais contient plus d'arbres que $\mathbf{taille}(q) \times 2(\text{degré}(p))$, contrairement à l'exemple précédent.

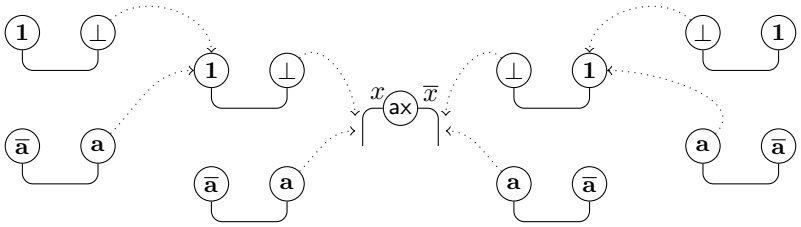


FIGURE 2.22 – Exemples de sauts chaînés pour une structure de degré 2.

On devine que la situation peut vite devenir explosive, ce schéma pouvant se répéter. Mais pour autant, tout n'est pas perdu, car si l'on regarde attentivement les configurations ici à l'œuvre, on peut voir que les chaînes de sauts de notre exemple induisent un chemin naturel entre les unités à l'extérieur et les conclusions de l'axiome.

Nous allons donc pouvoir raisonner à l'aide des deux mesures que sont le

degré de saut et la longueur des chemins. La borne deviendra exponentielle, comme on peut le deviner, mais on parviendra néanmoins à déterminer un nombre fini d'arbres effacés par la réduction.

Dans la suite de cette section, nous considérons que nos structures ne comportent pas d'affaiblissements et donc que toutes les coupures évanescentes sont entre unités multiplicatives. L'extension aux coupures entre affaiblissements et coaffaiblissements est triviale, mais la simplification que nous nous autorisons ici permet d'alléger l'exposition.

De plus, nous parlerons de structure sans préciser qu'il s'agira toujours dans la suite de structures à sauts.

2.3.3.1 Évolution de la taille sous réduction évanescente

On définit d'abord les notions décrites informellement plus tôt.

Définition 18.
Soit p une structure et $t \in \mathbf{SA}(p)$. Le *degré de saut* de t dans p est défini comme suit : $\mathbf{deg}_p(t) = \mathbf{card}\{\mu \in \mathbf{U}_\perp(p) \mid S_p(\mu) = t\}$ (on se rappelle que l'on identifie $\perp_\mu$ avec μ). On écrira simplement $\mathbf{deg}(t)$ s'il n'y a pas d'ambiguïté.

On pose ensuite $\mathbf{deg}(p) = \max\{\mathbf{deg}(t) \mid t \in \mathbf{SA}(p)\}$.

On définit ensuite pour chaque arbre t une hiérarchie d'ensembles de termes qui pointent héréditairement vers cet arbre à travers des suites de coupures évanescentes. C'est-à-dire que l'on cherche à isoler les co-unités dont le saut pointe vers notre arbre t, ou vers une coupure évanescente dont la co-unité pointe vers t, et ainsi de suite. Formellement :

Définition 19.
Soit p une structure, et $t \in \mathbf{SA}(p)$. On définit $\mathbf{E}^n(t)$ l'ensemble des arbres *entrant dans* t par une suite de n coupures évanescentes. $\mathbf{E}^0(t) = S_p^{-1}(t) = \{\nu \in \mathbf{U}_\perp(p) \mid S_p(\nu) = t\}$, et $\mathbf{E}^{n+1}(t) = \{\nu' \in \mathbf{U}_\perp(p) \mid S_p(\nu') \in \{\nu, \mu\}, \langle\nu|\mu\rangle \in \mathbf{C}(p), \nu \in \mathbf{E}^n(t)\}$. La définition est paramétée par S_p, on pourra donc écrire $\mathbf{E}_p^n(t)$ si nécessaire.

On peut déjà observer que $\mathbf{card}(\mathbf{E}^0(t)) = \mathbf{deg}(t)$. On a désormais les ingrédients nécessaires pour borner la taille des antiréduits sous réduction évanescente :

Lemme 17. Soient p, q deux structures telles que $p \rightrightarrows_\epsilon q$.

— Pour tout $t \in \mathbf{SA}(p)$, $\mathbf{card}\left(\bigcup_{i\in\mathbf{N}} \mathbf{E}_p^i(t)\right) \leq (2\mathbf{deg}(p))^{\mathbf{ln}(p)}$;

— p comporte au plus $\mathbf{taille}(q) \times (2\mathbf{deg}(p))^{\mathbf{ln}(p)}$ coupures évanescentes.

Démonstration. On établit d'abord que l'ensemble $\{n \in \mathbf{N} \mid \mathbf{E}^n(t) \neq \emptyset\}$ est fini pour tout $t \in \mathbf{SA}(p)$. En effet, pour $\nu_n \in \mathbf{E}^n(t)$, il existe une suite de coupures $c_1, \ldots, c_{n-1}$ telle que, si l'on écrit $c_i = \langle\mu_i|\nu_i\rangle$, l'unique chemin de ν_n à t est $\chi = \nu_n, \rho_{n-1}, \ldots, \rho_1, t$ où $S_p(\nu_1) = t$, et pour tout $i \in \{1, \ldots, n-1\}$, soit $S_p(\nu_{i+1}) = \mu_i$, et $\rho_i = c_i$, ou $S_p(\nu_{i+1}) = \nu_i$ et $\rho_i = \nu_i$. Voir la figure 2.23.

On observe immédiatement que $n \leq \mathbf{ln}(\chi)$. On déduit donc immédiatement $\max\{n \in \mathbf{N} \mid \mathbf{E}^n(t) \neq \emptyset\} \leq \mathbf{ln}(p)$. C'est dire qu'une chaîne telle qu'illustrée en figure 2.23 correspond à un chemin, et ne peut donc pas visiter plus de $\mathbf{ln}(p)$ coupures evanescentes. La longueur de χ est en fait comprise entre n et $2n$, car chaque coupure évanescente correspond soit à un arbre de χ (quand le chemin

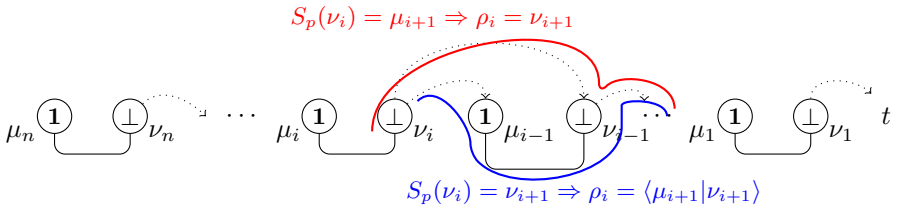


FIGURE 2.23 – Un chemin de n coupures évanescentes vers t, avec deux possibilités de jonction pour chaque coupure.

ne passe que par un seul arbre $\perp_\nu$ de la coupure), soit à deux (quand la coupure est traversée).

Maintenant, nous cherchons à borner la taille des ensembles $\mathbf{E}_p^n(t)$: nous allons montrer par induction sur n que $\mathbf{card}(\mathbf{E}_p^n(t)) \leq (2\mathbf{deg}(p))^n$. Nous avons déjà remarqué que $\mathbf{card}(\mathbf{E}^0(t)) = \mathbf{deg}(t) \leq \mathbf{deg}(p)$. Supposons donc maintenant que l'hypothèse est vraie pour $n \geq 0$. Pour toute coupure $c = \langle \mu | \nu \rangle \in \mathbf{C}(p)$ telle que $\nu \in \mathbf{E}_p^n(t)$, le nombre de ν' tels que $S_p(\nu') \in c$ est plus petit que $2\mathbf{deg}(p) \geq \mathbf{deg}(\nu) + \mathbf{deg}(\lambda)$. En effet, les arbres dont le saut pointe vers la coupure correspondent aux arbres pointant vers l'une ou l'autre de ses prémisses. On obtient donc $\mathbf{card}(\mathbf{E}_p^{n+1}(t)) \leq 2\mathbf{deg}(p)\mathbf{card}(\mathbf{E}^n(p))$. Le cas critique de cette configuration est illustré en figure 2.24.

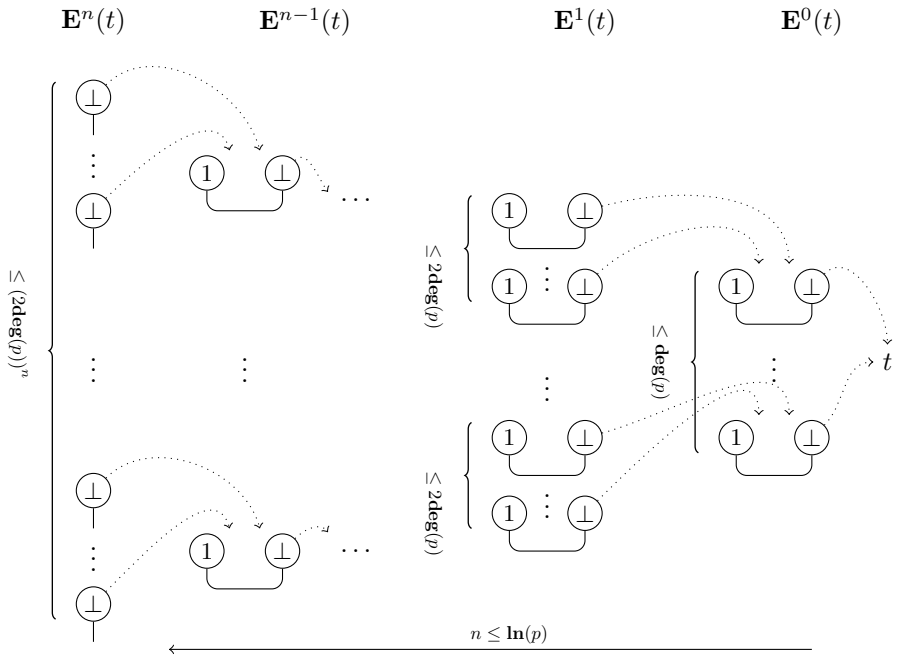


FIGURE 2.24 – Réductions évanescentes : cas critique

On a donc $\mathbf{card}\left(\bigcup_{i=0}^{\mathbf{ln}(p)} \mathbf{E}_p^i(t)\right) \leq \sum_{i=0}^{\mathbf{ln}(p)} (2\mathbf{deg}(p))^i$, ce qui achève la preuve

du premier point de ce lemme. Pour en déduire le second, il suffit de montrer que pour tout $\nu \in \mathbf{U}_{\perp}(p)$, il existe $t \in \mathbf{SA}(q)$ tel que $\nu \in \mathbf{E}_p^k(t)$ pour un certain $k \in \mathbf{N}$: En effet, le nombre de coupures évanescentes dans p est évidemment borné par $\mathbf{card}(\mathbf{U}_{\perp}(p))$.

Pour cela, posons $\nu_0 = \nu$, et $(c_i)_{i\in\{1,\dots,k\}}$ comme la plus longue suite de coupures $\langle \mu_i | \nu_i \rangle \in \mathbf{C}(p)$ telle que pour tout $i \in \{1, \dots, k-1\}$, $S_p(\nu_i) \in \{\nu_{i+1}, \mu_{i+1}\}$. Une suite maximale de la sorte existe par acyclicité. Nécessairement, $S_p(\nu_i)$ n'appartient pas à une coupure évanescente de $\mathbf{C}(p)$, donc $S_p \in \mathbf{SA}(q)$. On peut conclure, car $\nu_0 \in \mathbf{E}^k(S_p(\nu_k))$. □

Lemme 18. Soient p, q deux structures telles que $p \rightrightarrows_\epsilon q$. $\mathbf{taille}(p)$ est borné par une fonction sur $\mathbf{taille}(q), \mathbf{ln}(p), \mathbf{deg}(p)$. Plus précisément, $\mathbf{taille}(p) \leq \mathbf{taille}(q) + 2\mathbf{taille}(q)\,(2\mathbf{deg}(p))^{\mathbf{ln}(p)}$.

Démonstration. Par le lemme 17, on a $p = (\langle \mu_1 | \nu_1 \rangle, \dots, \langle \mu_n | \nu_n \rangle, \overrightarrow{c}; \overrightarrow{t})$, $q = (\overrightarrow{c}; \overrightarrow{t})$, et $n \leq \mathbf{taille}(q) \times (2\mathbf{deg}(p))^{\mathbf{ln}(p)}$. Comme pour toute coupure évanescente $c_i = \langle \mu_i | \nu_i \rangle$, $\mathbf{taille}(c_i) = 2$, la taille de p correspond à la taille de q plus $2n$. En particulier, $\mathbf{taille}(p) \leq \mathbf{taille}(q) + 2\left(\mathbf{taille}(q)\,(2\mathbf{deg}(p))^{\mathbf{ln}(p)}\right)$. □

2.3.3.2 Évolution des chemins sous réductions évanescentes

On aura besoin pour nos conclusions de s'assurer que quand $p \rightrightarrows_\epsilon q$, $\mathbf{ln}(q)$ peut être borné en fonction de $\mathbf{ln}(p)$. Il suffit ici de s'assurer que les réductions évanescentes ne peuvent que raccourcir les chemins.

Lemme 19. Soient $p \rightrightarrows_\epsilon q$. $\mathbf{ln}(q) \leq \mathbf{ln}(p)$.

Démonstration. Soit $t \sim_q s$ deux arbres adjacents dans q. Si $t \not\sim_p s$, alors $t \in \mathbf{U}_{\perp} \cup \mathbf{U}_?$ et la réduction évanescente a redirigé le saut de t vers s. C'est en effet la seule configuration dans laquelle une relation d'adjacence peut être créée par la réduction. Cela implique notamment que $t \in \mathbf{E}_p^n(s)$ pour un certain $n \in \mathbf{N}$. Or, on a déjà vu qu'ici, il existe un chemin $\chi \in \mathbf{Ch}(p)$ entre t et s tel que $\mathbf{ln}(\chi) \geq n$.

Donc on peut, à tout chemin $\xi \in \mathbf{Ch}(q)$ associer un chemin dans p qui est au moins aussi long que ξ, en remplaçant t, s par un chemin de p, comme ci-dessus, chaque fois que l'on a $t \sim_q s$ mais $t \not\sim_p s$. Ce qui conclut. □

2.3.3.3 Augmentations du degré de saut

Nous avons établi que le degré de saut permettait de traiter l'évolution de la taille des structures à travers une étape de réduction évanescente parallèle. Mais cherchant à étendre l'argument pour une itération de cette réduction, nous devons nous assurer qu'il est possible de contrôler la valeur de ce degré après une étape.

Nous allons donc mesurer $\mathbf{E}^0(t)$ pour tous les arbres t d'une structure réduite. On écrira simplement $\mathbf{E}(t)$.

Lemme 20. Soient p, q tels que $p \rightrightarrows_\epsilon q$. $\mathbf{deg}(q) \leq (2\mathbf{deg}(p))^{\mathbf{ln}(p)}$.

Démonstration. Soit $t \in \mathbf{SA}(q)$. Pour tout $\gamma \in \mathbf{U}_{\perp}(q) \cup \mathbf{U}_?(q)$, si $S_q(\gamma) = t$, alors il existe $n \in \mathbf{N}$ tel que $\gamma \in \mathbf{E}_p^n(t)$.

Par le lemme 17 (1), $\mathbf{card}(\bigcup_{n\in\mathbf{N}} \mathbf{E}^n(t)) \leq (2\mathbf{deg}(p))^{\mathbf{ln}(p)}$. Donc on a bien $\mathbf{deg}_q(t) \leq (2\mathbf{deg}(p))^{\mathbf{ln}(p)}$ pour tout $t \in \mathbf{SA}(q)$, et on conclut. □

Lemme 21. Soient p, q tel que $p \gg q$ par réductions persistantes ou multiplicatives. $\mathbf{deg}(q) \leq 2\mathbf{deg}(p)$.

Démonstration. Examinons les réductions par type de coupure. Soit $\langle \otimes(t_1, t_2) | \parr (s_1, s_2)\rangle \in \mathbf{C}(p)$ une coupure réduite. On se rappelle que l'élimination des coupures redirige les sauts qui pointaient sur les arbres coupés, en leur attribuant la première prémisse. En particulier, $\mathbf{E}_q(t_1) = \mathbf{E}_p(t_1) \cup \mathbf{E}_p(\otimes(t_1, t_2))$, et $\mathbf{E}_q(s_1) = \mathbf{E}_p(s_1) \cup \mathbf{E}_p(\parr(s_1, s_2))$. Le degré des autres arbres est conservé. Comme $\mathbf{E}_p(t) \leq \mathbf{deg}(p)$ pour tout t par définition, $\mathbf{deg}_q(t_1)$ et $\mathbf{deg}_q(s_1)$ sont inférieurs à $2\mathbf{deg}(p)$.

Si $\langle !(t) | d(s)\rangle$ est réduite, on raisonne de la même façon. $\mathbf{E}_q(t) = \mathbf{E}_p(t) \cup \mathbf{E}_p(!(t))$, et $\mathbf{E}_q(s) = \mathbf{E}_p(s) \cup \mathbf{E}_p(d(s))$. Donc $\mathbf{deg}_q(t) \leq 2\mathbf{deg}(p)$ et $\mathbf{deg}_q(s) \leq 2\mathbf{deg}(p)$.

Si $\langle c(t_1, \ldots, t_n) | \overline{\mathbf{a}}_\lambda\rangle$ est réduite en $\langle t_1 | \overline{\mathbf{a}}_{\lambda_1}\rangle, \ldots, \langle t_n | \overline{\mathbf{a}}_{\lambda_n}\rangle$, alors on a $\mathbf{E}_q(t_1) = \mathbf{E}_p(t_1) \cup \mathbf{E}_p(c(t_1, \ldots, t_n))$, $\mathbf{E}_q(\overline{\mathbf{a}}_{\lambda_1}) = \mathbf{E}_p(\overline{\mathbf{a}}_\lambda)$, $\bigcup_{i=2}^n \mathbf{E}_q(\overline{\mathbf{a}}_{\lambda_i}) = \emptyset$, et $\mathbf{E}_q(t_j) = \mathbf{E}_p(t_j)$ pour $j \neq 1$. Donc $\mathbf{deg}(t_1) \leq 2\mathbf{deg}(p)$, et le degré des autres arbres reste inférieur ou égal à $\mathbf{deg}(p)$.

Si $\langle !(t_1, \ldots, t_n)) | c(s_1, \ldots, s_m)\rangle \gg_{\langle !|c\rangle} \langle !(\overrightarrow{\delta}_1) | s_1\rangle, \ldots, \langle !(\overrightarrow{\delta}_m) | s_m\rangle$, alors on a $\mathbf{E}_q(s_1) = \mathbf{E}_p(s_1) \cup \mathbf{E}_p(c(s_1, \ldots, s_m))$, $\mathbf{E}_q(s_i) = \mathbf{E}_p(s_i)$ pour $i \neq 1$. Et $\mathbf{E}_q(!(\overrightarrow{\delta}_1)) = \mathbf{E}_p(!(t_1, \ldots, t_n))$, $\mathbf{E}_q(!(\overrightarrow{\delta}_i)) = \emptyset$ pour $i \neq 1$. On a donc $\mathbf{deg}(s_1) \leq 2\mathbf{deg}(p)$, et les autres arbres de degré inférieur ou égal à celui de p.

Si p est de la forme $(c_1, \ldots, c_n, \overrightarrow{d}; \overrightarrow{t})$ et q est de la forme $(\overrightarrow{c}'_1, \ldots, \overrightarrow{c}'_n, \overrightarrow{d}; \overrightarrow{t})$, avec $c_i \gg \overrightarrow{c}'_i$ par l'une des réductions vues ci-dessus, alors on a établi que pour tout $s \in \mathbf{SA}(c_i)$, et tout $s' \in \mathbf{SA}(\overrightarrow{c}'_i)$, $\mathbf{deg}(s') \leq 2\mathbf{deg}(s)$. On peut donc conclure que $\mathbf{deg}(q) \leq 2\mathbf{deg}(p)$, car le degré des arbres de $\overrightarrow{d}$ et de $\overrightarrow{t}$ reste inchangé. □

Lemme 22. Soient $p \rightrightarrows_{\mathsf{ax}} q$. $\mathbf{deg}(q) \leq \mathbf{ln}(p)\mathbf{deg}(p)$.

Démonstration. Pour considérer la redirection des sauts pendant les contractions de chaînes d'axiomes, on caractérise ces dernières explicitement comme pour le lemme 10, en posant $p = (\langle x_1 | t_1\rangle, \ldots, \langle x_n | t_n\rangle, \overrightarrow{c}; \overrightarrow{s})$ et en écrivant $\langle x_1 | t_1\rangle, \ldots, \langle x_n | t_n\rangle = \overrightarrow{c}_1, \ldots, \overrightarrow{c}_k$, où

$$\overrightarrow{c}_i = \langle x_0^i | \overline{x}_1^i\rangle, \ldots, \langle x_{n_{i-1}}^i | \overline{x}_{n_i}^i\rangle, \langle x_{n_i}^i | t_i\rangle$$

et où $q = (\overrightarrow{c}; \overrightarrow{s})[t_1/\overline{x}_0^1, \ldots, t_k/\overline{x}_0^k]$. Par la définition de l'élimination des coupures, pour tout $\gamma \in \mathbf{U}_\perp(p) \cup \mathbf{U}_?(p)$ tel que $S_p(\gamma) \in \{\overline{x}_0^i, x_0^i, \ldots, x_{n_i}^i, t^i\}$, on a $S_q(\gamma) = t^i$. Plus précisément,

$$\mathbf{E}_q(t) = \mathbf{E}_p(t) \cup \bigcup_{x \in \{\overline{x}_0^i, x_0^i, \ldots, x_{n_i}^i\}} \mathbf{E}_p(x)$$

Pour toute variable $x \in \mathcal{V}(p)$, on a $\mathbf{deg}(x) \leq \mathbf{deg}(p)$, et $2n_i + 3 \leq \mathbf{ln}(p)$, car chaque chaîne considérée ici induit un chemin visitant les $2n_i + 3$ arbres

$\{\overline{x}_0^i, x_0^i, \ldots, x_{n_i}^i, t^i\}$. On peut conclure que pour tout $i \in \{1, \ldots, n\}$,

$$\begin{aligned}\mathbf{deg}_q(t^i) &\leq \mathbf{deg}_p(t_i) + \sum_{x \in \{x_0^i, \vec{i}_0^i, \ldots, \overline{x}_{n_i}^i\}} \mathbf{deg}_p(x) \\ &\leq (2n_i + 3)\mathbf{deg}(p) \\ &\leq \mathbf{ln}(p)\mathbf{deg}(p)\end{aligned}$$

Il suffit alors d'observer qu'aucun saut autre que ceux qui pointaient sur les x_j^i n'est redirigé pendant la réduction axiomatique. Cela implique en particulier que pour tout $s \in \mathbf{SA}(q)/\{t_1, \ldots, t_n\}$, on a $\mathbf{deg}_q(s) = \mathbf{deg}_p(s)$. On peut conclure que $\mathbf{deg}(q) \leq \mathbf{ln}(p)\mathbf{deg}(p)$. □

On peut maintenant donner une borne générale sur l'évolution du degré de saut des structures, en rassemblant les résultats précédents.

Théorème 4. Soient p, q tels que $p \gg q$. Il existe une fonction croissante $\theta : \mathbf{N} \times \mathbf{N} \to \mathbf{N}$ telle que $\mathbf{deg}(q) \leq \theta(\mathbf{ln}(p), \mathbf{deg}(p))$.

Démonstration. On utilise la standardisation pour poser $p \rightrightarrows_{\mathsf{ax}} p_1 \rightrightarrows_{\epsilon} p_2 \gg q$, où $p_2 \gg q$ est une réduction persistante et multiplicative.

Par le lemme 21, $\mathbf{deg}(q) \leq 2\mathbf{deg}(p_2)$. Par le lemme 20, on a $\mathbf{deg}(p_2) \leq (2\mathbf{deg}(p_1))^{\mathbf{ln}(p_1)}$. Par le lemme 22, $\mathbf{deg}(p_1) \leq (\mathbf{ln}(p))\mathbf{deg}(p)$. Or, $\mathbf{ln}(p2) \leq \mathbf{ln}(p_1) \leq \mathbf{ln}(p)$ (on a déjà vu que les réductions axiomatique et évanescente ne font que contracter les chemins, au Corollaire 3 et au lemme 19).

Donc on peut poser

$$\mathbf{deg}(q) \leq 2\left((2(\mathbf{ln}(p))\mathbf{deg}(p))^{\mathbf{ln}(p)}\right)$$

qui convient pour la fonction θ recherchée. □

2.3.4 Cas général et conclusions sur $\mathbf{DLL}_0$

Pour conclure cette section, on peut donner les résultats concernant la réduction parallèle générale sur les réseaux à ressources. On donne une borne sur la taille des antiréduits en fonction de nos mesures $\mathbf{deg}()$ et $\mathbf{ln}()$ et de la taille du réduit, puis on montre que nos mesures restent bornées, de façon à ce que notre argument puisse être itéré.

Théorème 5. Soient $p \rightrightarrows \sum_{i \in I} q_i$. Il existe une fonction φ telle que pour tout $i \in I$, $\mathbf{taille}(p) \leq \varphi(\mathbf{taille}(q_i), \mathbf{ln}(p), \mathbf{deg}(p))$.

Démonstration. Par standardisation, pour tout q_i donné, on peut poser $p \rightrightarrows_{\epsilon} p_1 \rightrightarrows_{\mathsf{ax}} p_2 \rightrightarrows_m p_3 \gg q$.

Par le lemme 16, $\mathbf{taille}(p_3) \leq 6\mathbf{taille}(q)$. Par le lemme 9, $\mathbf{taille}(p_2) \leq \frac{3\mathbf{taille}(p_3)}{2}$. Par le lemme 10, $\mathbf{taille}(p_1) \leq (\mathbf{ln}(p_1))\mathbf{taille}(p_2)$. Par le lemme 18, $\mathbf{taille}(p) \leq \mathbf{taille}(p_1) + 2\mathbf{taille}(p_1)\,(2\mathbf{deg}(p))^{\mathbf{ln}(p)}$. On peut donc poser $\mathbf{taille}(p)$ inférieur ou égal à :

$$(\mathbf{ln}(p_1))\frac{3(6\mathbf{taille}(q))}{2} + 2\left((\mathbf{ln}(p_1))\frac{3(6\mathbf{taille}(q))}{2}\right)(2\mathbf{deg}(p))^{\mathbf{ln}(p)}$$

Par le lemme 19, $\mathbf{ln}(p_1) \leq \mathbf{ln}(p)$. On peut donc écrire, en simplifiant en même temps l'équation ci-dessus :

$$\begin{aligned}\mathbf{taille}(p) &\leq (\mathbf{ln}(p))9\mathbf{taille}(q) + ((\mathbf{ln}(p))18\mathbf{taille}(q))\,(2\mathbf{deg}(p))^{\mathbf{ln}(p)} \\ &= (\mathbf{ln}(p)18\mathbf{taille}(q))\left((2\mathbf{deg}(p))^{\mathbf{ln}(p)} + \frac{1}{2}\right)\end{aligned}$$

qui est bien une fonction sur $\mathbf{taille}(q), \mathbf{deg}(p)$ et $\mathbf{ln}(p)$, et convient donc pour le φ recherché. □

Lemme 23. Soient $p \rightrightarrows \sum_{i\in I} q_i$. Pour tout $i \in I$, $\mathbf{ln}(q_i) \leq f_{\mathbf{ln}}(\mathbf{ln}(p))$.

Démonstration. Pour un q_i donné, on peut écrire $p \rightrightarrows_m p_1 \gg p_2 \rightrightarrows_\epsilon p_3 \rightrightarrows_{\mathsf{ax}} q$, avec $p_1 \gg p_2$ par réductions persistantes. Par le théorème 3, $\mathbf{ln}(p_2) \leq f_{\mathbf{ln}}(\mathbf{ln}(p))$. Par le Corollaire 3 et le lemme 19, $\mathbf{ln}(q) \leq \mathbf{ln}(p_3) \leq \mathbf{ln}(p_2)$, ce qui conclut. □

Théorème 6. Soient $k, l, m \in \mathbf{N}$, et q une structure de $\mathbf{DLL}_0$. $\{p \in \mathbf{DLL}_0 \mid \mathbf{ln}(p) \leq l \wedge \mathbf{deg}(p) \leq m \wedge p \gg^k q\}$ est un ensemble fini.

Démonstration. La preuve est par induction sur k, et identique à celle du théorème 2, en utilisant cette fois le lemme 23 et les théorèmes 4 et 5. □

2.4 MELL

Nous présentons maintenant les constructions permettant de définir les boîtes exponentielles, en utilisant une approche syntaxique que nous avons décrite en section 2.1.1.4.

2.4.1 Syntaxe, contextes de boîtes

Définition 20.
On considère dans un premier temps cinq ensembles dénombrables, deux-à-deux disjoints, $\mathcal{V}, \mathbf{U_1}, \mathbf{U}_\perp, \mathbf{U}_?, \mathbf{B}$, où $\mathbf{B} = \{a_i^b \mid i, b \in \mathbf{N}\}$ est une énumération de *portes de boîtes*. On dira que a_0^b est la porte principale de la boîte b, et que a_{i+1}^b en est une porte auxiliaire. La réunion de ces cinq ensembles constitue les *atomes* de **MELL**, notés $\mathbf{A}$.

Les pré-arbres de **MELL** sont donnés par la syntaxe suivante, où $x \in \mathcal{V}, \mu \in \mathbf{U_1}, \nu \in \mathbf{U}_\perp, \kappa \in \mathbf{U}_?, a_i^b \in \mathbf{B}$, et $n \neq 0$:

$$T, S ::= x \mid a_i^b \mid \mathbf{1}_\mu \mid \perp_\nu \mid \mathbf{a}_\kappa \mid \otimes(T,S) \mid \parr(T,S) \mid c(T,S) \mid d(T)$$

À nouveau, on identifiera les (co)unités et les affaiblissements avec les indices de leurs occurrences.

On définit $\mathbf{SA}(T)$ l'ensemble des sous-arbres de T comme précédemment, et notons $\mathbf{X}(T) = \mathbf{SA}(T) \cap \mathbf{X}$, pour tout $\mathbf{X} \in \{\mathbf{B}, \mathcal{V}, \mathbf{U_1}, \mathbf{U}_\perp, \mathbf{U}_?, \mathbf{A}\}$. Un *arbre* est un pré-arbre dans lequel chaque atome a au plus une occurrence. Une *coupure* est une paire d'arbres $C = \langle T|S\rangle$ dont les atomes sont disjoints.

Nous avons décrit les structures de surface, il nous faut maintenant définir les boîtes, de façon à introduire un peu de profondeur dans nos réseaux.

On définit par induction mutuelle les pré-structures et structures de **MELL**, et les contextes de boîte. Un *contexte de boîte* Θ est la donnée d'un ensemble fini

$B_\Theta \subset \mathbf{N}$, et d'une structure de la forme $\Theta(b) = i((\overrightarrow{C}_b; \overrightarrow{S}_b, T_b), S_{\Theta(b)}, \Theta_b)$, pour tout $b \in B_\Theta$. On note $\mathbf{ar}_\Theta(b)$, ou simplement $\mathbf{ar}(b)$ la longueur de $\overrightarrow{S}_b$, que l'on appelle *arité* de la boîte b, et qui correspond au nombre de portes auxiliaires qu'elle comporte.

Une pré-structure est un triplet $P = ((\overrightarrow{C}, \overrightarrow{S}), S_P, \Theta)$ tel que Θ est un contexte de boîte, chaque atome a au plus une occurrence dans $(\overrightarrow{C}; \overrightarrow{S})$, S_P est une fonction de saut de domaine $\mathbf{U}_\perp((\overrightarrow{C}; \overrightarrow{S})) \cup \mathbf{U}_?((\overrightarrow{C}; \overrightarrow{S}))$ et de codomaine $\mathbf{SA}((\overrightarrow{C}; \overrightarrow{S}))$, et tel que si $a_i^b \in \mathbf{B}((\overrightarrow{C}; \overrightarrow{S}))$, alors $b \in B_\Theta$ et $i \leq \mathbf{ar}(b)$. Une structure est une pré-structure $P = ((\overrightarrow{C}; \overrightarrow{T}), S_P, \Theta)$ telle que $x \in \mathbf{SA}((\overrightarrow{C}; \overrightarrow{T}))$ si et seulement si $\overline{x} \in \mathbf{SA}((\overrightarrow{C}; \overrightarrow{T}))$, et si de plus, si $b \in B_\Theta$, alors $a_i^b \in \mathbf{SA}((\overrightarrow{C}; \overrightarrow{T}))$ pour tout $i \in \{0, \ldots, \mathbf{ar}(b)\}$. On écrira parfois Θ_P pour désigner le contexte de boîte de P.

La profondeur d'une structure P, notée $\mathbf{prof}(P)$, correspond à la profondeur inductive de la définition ci-dessus, c'est le niveau maximal d'imbrication des boîtes de P.

Nous dénoterons souvent abusivement les structures en omettant les informations de sauts ou de contexte de boîte, quand elles ne seront pas pertinentes, et les écrirons $((\overrightarrow{C}; \overrightarrow{T}), \Theta)$ ou encore $(\overrightarrow{C}; \overrightarrow{T})$.

Les structures sont considérées à nouveau à α-équivalence près, et à renommage des atomes près (voir la définition 9).

Il faudra dans la suite prendre le soin de voir si nous raisonnons sur une structure de surface, ou sur une structure en profondeur. Par exemple, $\mathbf{taille}(P) = \mathbf{card}(\mathbf{SA}(P))$ correspond au nombre de sous-arbres de P à la profondeur 0, c'est-à-dire sans tenir compte du contenu des boîtes. On doit définir une deuxième notion de taille, qui correspond au nombre de sous-arbres à toutes les profondeurs. Ainsi on définit la *taille à toute profondeur* $\mathbf{Ptaille}(P) = \mathbf{taille}(P) + \sum_{b \in B_\Theta} \mathbf{Ptaille}(\Theta(b))$ (cette définition est par induction sur la profondeur).

Remarquons que par définition, pour tout $\gamma \in \mathbf{U}_\perp(P) \cup \mathbf{U}_?(P)$, $S_P(\gamma)$ est un sous-arbre qui est à la même profondeur que γ.

Les chemins sont définis de la même façon que dans $\mathbf{DLL}_0$, en ajoutant une nouvelle relation d'adjacence $\sim_b$ pour $b \in B_\Theta$, en posant $a_i^b \sim_b a_j^b$ si $0 \leq i \leq j \leq \mathbf{ar}(b)$. Les boîtes se comportent du point de vue des chemins comme des axiomes d'arité $\mathbf{ar}(b)+1$. Les chemins sont donc des constructions de surface, le contenu des boîtes n'est pas considéré. On écrit $\mathbf{Ch}(P)$ l'ensemble des chemins de P, et l'on dit que P est acyclique si aucun chemin de P n'est un cycle, et si inductivement $\Theta(b)$ est acyclique pour tout $b \in B_\Theta$. Nous ne considérons à nouveau que des structures acycliques.

2.4.2 Réductions exponentielles

À propos des *copies* de structures : Dans les constructions qui suivent, nous serons amenés à considérer des *copies de structures*. Nos structures étant considérées à renommage des feuilles près, une copie d'une structure P est simplement une instance de P. Mais si l'on construit une structure faite de plusieurs copies, on considère que les noms des atomes sont renommés à la volée, de façon à ce que les définitions soient respecteés. Par exemple, on a $(; x, \overline{x}) = (; y, \overline{y})$, mais $(; x, \overline{x}, y, \overline{y}) \neq (; x, \overline{x}, x, \overline{x})$. Le dernier objet n'étant même pas une structure.

Définition 21.
On définit une famille de réductions qui correspond à l'élimination des coupures de **MELL**.

Si $P = ((\langle a_0^b | d(S)\rangle, \overrightarrow{C}; \overrightarrow{U}), S_P, \Theta)$ est une structure de **MELL** telle que $\Theta(b) = ((\overrightarrow{C}_b; S_1, \ldots, S_{\mathbf{ar}(b)}, T_b), \Theta_b, S_{\Theta(b)})$, alors, $P \to_{\langle a_0|d\rangle}$

$$((\langle a_0^b|S\rangle, \overrightarrow{C}, \overrightarrow{C}_b; \overrightarrow{U})[T_b/a_0^b, S_1/a_1^b, \ldots, S_{\mathbf{ar}(b)}/a_{\mathbf{ar}(b)}^b], S'_P \cup S_{\Theta(b)}, \Theta \cup \Theta_{\Theta_b})$$

Dans cette réduction, on remplace toutes les portes de la boîte b par les conclusions de la structure emboîtée, on « ouvre la boîte ». En particulier, $\langle T_b|S\rangle$ est une nouvelle coupure dans la structure réduite. S'_P coïncide avec S_P, à l'exception des $\gamma \in \mathbf{U}_\perp \cup \mathbf{U}_?$ tels que $S_P(\gamma) = d(S)$ ou $S_P(\gamma) \in \{a_i^b \mid 0 \leq i \leq \mathbf{ar}(b)\}$. Dans le premier cas, on pose $S'_P(\gamma) = S$, et dans le second, $S'_P(\gamma) = S_P(\gamma)[T_b/a_0^b, S_1/a_1^b, \ldots, S_{\mathbf{ar}(b)}/a_{\mathbf{ar}(b)}^b]$.

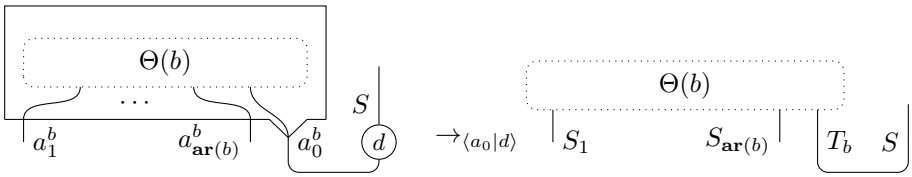


FIGURE 2.25 – Réduction promotion/déréliction, la boîte est ouverte

Si $P = ((\langle a_0^b | c(T_1, T_2)\rangle, \overrightarrow{C}; \overrightarrow{U}), S_P, \Theta)$ et si le contenu de la boîte b est égal à $\Theta(b) = ((\overrightarrow{C}_b; S_1, \ldots, S_{\mathbf{ar}(b)}, T_b), \Theta_b, S_{\Theta(b)})$, alors on doit considérer deux copies distinctes de $\Theta(b)$ (avec des ensembles d'atomes disjoints). Pour $i \in \{1, 2\}$, on pose $P_i = ((\overrightarrow{C}_{b_i}; S_{i,1}, \ldots, S_{i,\mathbf{ar}(b_i)}, T_{b_i}), \Theta_{b_i}, S_{\Theta(b_i)})$ deux copies de $\Theta(b)$. On pose alors

$$P \to_{\langle a_0|c\rangle} \left(\left(\langle a^{b_1}|T_1\rangle, \langle a^{b_2}|T_2\rangle, \overrightarrow{C}; \overrightarrow{U}\right)([c(a_i^{b_1}, a_i^{b_3})/a_i^b])_{i\in\{1,\ldots,\mathbf{ar}(b)\}}, S'_P, \Theta'\right)$$

où S'_P coïncide avec S_P, sauf pour les atomes γ tels que $S_P(\gamma) = (T_1, T_2)$ ou $S_P(\gamma) \in \{a_i^b \mid 0 \leq i \leq \mathbf{ar}(b)\}$, dans le premier cas, on pose $S'_P(\gamma) = T_1$, et dans le second, $S'_P(\gamma) = S_P(\gamma)([a_i^b/c(a_i^{b_1}, a_i^{b_3})])_{i\in\{1,\ldots,\mathbf{ar}(b)\}}[a_0^{b_1}/a_0^b]$. Quant à Θ', il est défini comme $(\Theta/\{b \to P\}) \cup \{b_1 \to P_1, b_2 \to P_2\}$. On a remplacé la boîte par ses deux copies dans le contexte.

Si $P = ((\langle a_0^b|\mathbf{a}_\kappa\rangle, \overrightarrow{C}; \overrightarrow{U}), S_P, \Theta)$ et si le contenu de la boîte b est égal à $\Theta(b) = ((\overrightarrow{C}_b; S_1, \ldots, S_{\mathbf{ar}(b)}, T_b), \Theta_b, S_{\Theta(b)})$, alors on pose :

$$P \to_{\langle a_0|\mathbf{a}\rangle} ((\overrightarrow{C}; \overrightarrow{U})([\mathbf{a}_{\kappa_i}/a_i^b])_{i\in\{1,\ldots,\mathbf{ar}(b)\}}, S'_P, \Theta')$$

où les $\mathbf{a}_{\kappa_i}$ sont des nouveaux noms d'affaiblissements, et où S'_P coïncide avec S_P, sauf pour les atomes γ tels que $S_P(\gamma) \in \{\mathbf{a}_\kappa, a_0^b\}$ ou $S_P(\gamma) \in \{a_i^b \mid 1 \leq i \leq \mathbf{ar}(b)\}$. Dans le premier cas, on pose $S'_P(\gamma) = \mathbf{a}_{\kappa_1}$, et dans le second, on pose $S'_P(\gamma) = S_P(\gamma)([\mathbf{a}_{\kappa_i}/a_i^b])_{i\in\{1,\ldots,\mathbf{ar}(b)\}}$. De plus, il faut que S'_P soit défini pour les nouveaux affaiblissements, sans que ne soit créé un cycle. On pose donc pour tout $i \in \{1, \ldots, \mathbf{ar}(b) - 1\}$, $S'_P(\mathbf{a}_{\kappa_i}) = \mathbf{a}_{\kappa_{i+1}}$, et $S'_P(\mathbf{a}_{\kappa_{\mathbf{ar}(b)}}) = S_P(\mathbf{a}(\kappa))$. Et bien sûr, $S'_P(\mathbf{a}_\kappa)$ n'est plus défini, car $\mathbf{a}_\kappa$ n'apparaît plus. Comme toutes les portes

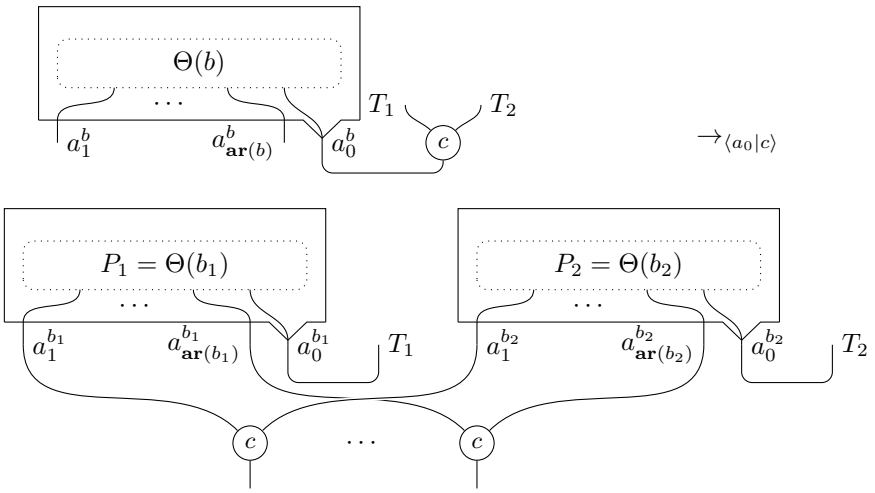


FIGURE 2.26 – Réduction promotion/contraction, la boîte est dupliquée

d'une même boîte sont adjacentes, on vérifie facilement que l'on n'a pas créé de chemins dans le réduit entre deux arbres qui n'étaient pas reliés dans P, on conserve donc l'acyclicité.

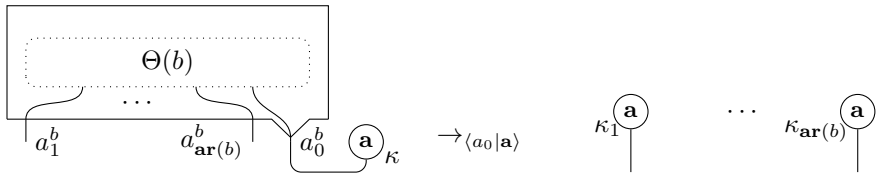


FIGURE 2.27 – Réduction promotion/affaiblissement, le contenu de la boîte est effacé

Si $P = ((\langle a_0^b | a_{l+1}^c \rangle, \overrightarrow{C}; \overrightarrow{U}), S_P, \Theta)$ et si le contenu de c est égal à $\Theta(c) = ((\overrightarrow{C}_c; S_1, \ldots, S_{\mathbf{ar}(c)}, T_c), \Theta_c, S_{\Theta(c)})$, alors il faut construire une nouvelle boîte. On pose

$$Q = ((\langle a_0^b | S_{l+1} \rangle, \overrightarrow{C}_c; a_1^b, \ldots, a_{\mathbf{ar}(b)}^b, (S_j)_{j \in \{1, \ldots, \mathbf{ar}(b)\}/\{S_{l+1}\}}, T_c), \Theta_c, S_\emptyset)$$

Q correspond au réseau où l'on a ouvert la boîte c, pour couper l'arbre correspondant à a_{l+1}^c avec a_0^b. Ici, $\Theta'_c = \Theta_c \cup \{b \to \Theta(b)\}$, et $S_\emptyset$ est la fonction de saut vide.

On pose alors

$$P \to_{\langle a_0 | a \rangle} ((\overrightarrow{C}; \overrightarrow{U})([a_i^d / a_i^b])_{1 \leq i \leq \mathbf{ar}(b)}([a_{\mathbf{ar}(b)+j}^d / a_j^c])_{1 \leq j \leq \mathbf{ar}(c)}[a_0^d / a_0^c], S'_P, \Theta')$$

où $\Theta' = (\Theta / \{b \to \Theta(b), c \to \Theta(c)\}) \cup \{d \to Q\}$ et où S'_P coïncide avec S_P, sauf pour les atomes γ tels que $S_P(\gamma) \in \{a_i^b \mid i \in \{1, \ldots, \mathbf{ar}(b)\} \cup \{a_j^c \mid c \in \{0, \ldots, \mathbf{ar}(c)\}/\{l+1\}\}$, auquel cas on pose :

$$S'_P(\gamma) = S_P(\gamma)([a_i^d / a_i^b])_{1 \leq i \leq \mathbf{ar}(b)}([a_{\mathbf{ar}(b)+j}^d / a_j^c])_{1 \leq j \leq \mathbf{ar}(c)}[a_0^d / a_0^c]$$

ou tels que $S_P(\gamma) \in \{a_0^b, a_{l+1}^c\}$, auquel cas on pose $\mathbf{S}'_P(\gamma) = a_0^d$. On fait ce choix car il est possible que $\mathbf{ar}(d) = 0$, et car ainsi on ne risque pas non plus de créer de cycle (il aurait par ailleurs été possible de rediriger ces sauts dans Q, en conservant les mêmes propriétés). On peut vérifier que la structure réduite respecte les contraintes de la définition des structures et contextes de boîte, notamment en s'assurant que le contexte $d \to Q$ est cohérent, respectivement à l'arité des boîtes. On remarque par ailleurs que l'arité de d est égale à $(\mathbf{ar}(b) + \mathbf{ar}(c)) - 1$.

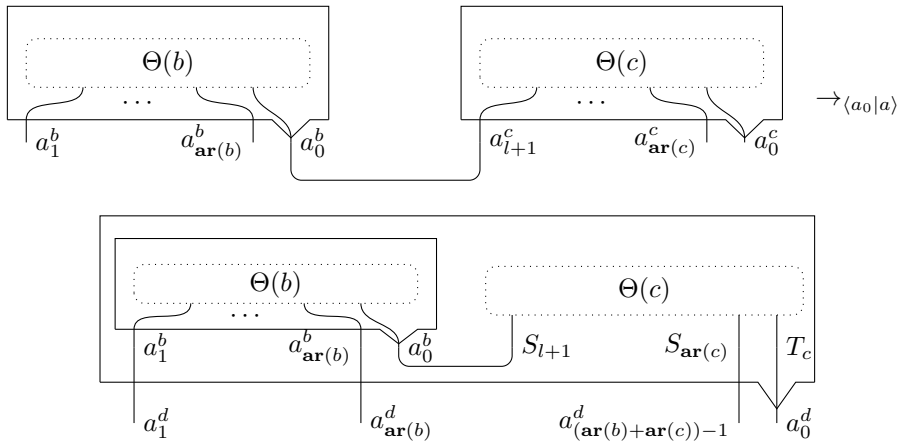


FIGURE 2.28 – Réduction promotion/promotion (pour $l = 0$)

De plus, on conserve les réductions $\to_m$ et $\to_{\mathsf{ax}}$ de **MLL**.

2.5 Développement de Taylor

2.5.1 Définition

On définit maintenant le développement de Taylor des réseaux de **MELL**. On pourra alors montrer que les considérations des sections précédentes nous permettront d'utiliser la réduction des structures de $\mathbf{DLL}_0$ pour simuler l'élimination des coupures exponentielles qui vient d'être définie.

Le développement de Taylor est une fonction qui à chaque réseau de **MELL** associe une somme infinie de structures de $\mathbf{DLL}_0$ avec coefficients. Formellement, soit $\mathbf{S}$ un semi-anneau, $\mathcal{T} : \mathbf{MELL} \to \mathbf{S}^{\mathbf{DLL}_0}$ (voir la section 1.4 pour les notations et constructions relatives aux séries et coefficients).

Définition 22 (Développement de Taylor des réseaux)**.** Soit $P = ((\overrightarrow{C}; \overrightarrow{T}), \Theta)$ une structure de **MELL**. La définition du *développement de Taylor* de P (que nous appellerons parfois simplement le *développement* de P) est par induction sur la profondeur. Si P est de profondeur nulle, alors $\mathcal{T}(P) = P$. Dans ce cas, P est en effet déjà une structure de $\mathbf{DLL}_0$.

Sinon, on considère les boîtes de P, dont les noms sont dans B_Θ, et on suppose que pour tout $b \in B_\Theta$, $\mathcal{T}(\Theta(b)) = \sum_{p^b \in \mathbf{DLL}_0} \lambda_{p^b} \cdot p^b$ est défini (pour tout

b, $\lambda_{p^b} \in \mathbf{S}$ est un scalaire), et que pour tout i, p_i contient $\mathbf{ar}(b)+1$ conclusions (ce qui se vérifie à profondeur 0, et dont on peut vérifier que ça reste vrai dans la construction inductive du développement). On pose alors, pour tout $b \in B_\Theta$, et pour tout i, $p_i^b = ((\overrightarrow{c}_i^b; t_{i,1}^b, \ldots, t_{i,\mathbf{ar}(b)}^b, t_{i,0}^b), S_{p_i^b})$. Si l'on pose alors $B_\Theta = \{b_1, \ldots, b_r\}$, alors on peut définir le développement de Taylor de P comme suit :

$$\mathcal{T}(P) = \sum_{n_{b_1}\ldots n_{b_r}\in\mathbf{N}} \frac{1}{n_{b_1}!\ldots n_{b_r}!} \sum_{\substack{(p_1^{b_1},\ldots,p_{n_{b_1}}^{b_1}) \\ \in|\mathcal{T}(\Theta(b_1))|^{n_{b_1}}}} \cdots \sum_{\substack{(p_1^{b_r},\ldots,p_{n_{b_r}}^{b_r}) \\ \in|\mathcal{T}(\Theta(b_r))|^{n_{b_r}}}} \prod_{i=1}^{r}\prod_{j=1}^{n_{b_i}} \lambda_{p_j^{b_i}}$$
$$\left(\left(\overrightarrow{C}, (\overrightarrow{c}_i^b)_{b\in B_\Theta, 1\le i\le n_b}; \overrightarrow{T}\right), S'\right) \left(\left[c(t_{1,l}^b,\ldots,t_{n_b,l}^b)/a_l^b\right]\right)_{b\in B_\Theta, 1\le l\le \mathbf{ar}(b)}$$
$$\left(\left[!(t_{1,0}^b,\ldots,t_{n_b,0}^b)/a_0^b\right]\right)_{b\in B\Theta}$$

où S' est défini dans ce cas comme $\bigcup_{b\in B_\Theta, 1\le i\le k} S_{p_i^b} \cup S'_P$ et S'_P coïncide avec S_P, sauf pour les atomes $\gamma \in \mathbf{U}_\perp(P) \cup \mathbf{U}_?(P)$ tels que $S_P(\in \mathbf{B}(P))$. Dans ce cas, on pose $S'_P(\gamma) = S_P(\gamma)\sigma$, où σ est la substitution ci-dessus (qui associe $c(t_{1,l}^b,\ldots,t_{n_b,l}^b)$ à a_1^b, etc...). De plus, si $n_b = 0$, $c(t_{1,l}^b,\ldots,t_{n_b,l}^b) = \mathbf{a}_\kappa$ pour un $\kappa \in \mathbf{U}_?$ frais, et $!(t_{1,0}^b,\ldots,t_{n_b,0}^b) = \overline{\mathbf{a}}_\lambda$ pour $\lambda \in \mathbf{U}_!$ frais. Dans ce cas, il faut que S' soit défini pour les nouveaux affaiblissements, donc on pose $S'((a_{i+1}^b)\sigma) = (a_0^b\sigma)$ le cas échéant.

De plus, pour que le développement soit bien défini, les p_i^j sont considérés à renommage des atomes près, de façon à ce que l'on ait construit des structures bien définies. En particulier, $\mathbf{SA}(p_i^b) \cap \mathbf{SA}(p_j^b) = \emptyset$ quand $i \neq j$.

On peut remarquer que tous les réseaux apparaissant dans $\mathcal{T}(P)$ ont autant de conclusions que P, car ils sont obtenus depuis P en opérant des substitutions et en rajoutant des coupures, ce qui ne change pas le nombre de conclusions. On remarque également qu'ils ne contiennent plus de porte de boîte, et que leurs cocontractions ont toujours pour prémisses des codérélictions, ce qui en fait bien des structures de $\mathbf{DLL}_0$.

Pour éclaircir la définition 22, déroulons-la sur un exemple très simple. Considérons la structure $P = ((; a_1^b, a_0^b), \{b \to (; x, \overline{x})\})$ représentée ci-dessous : Ici,

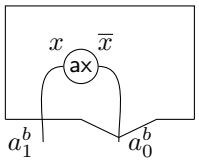


FIGURE 2.29 – Un axiome emboîté

$\mathcal{T}(\Theta(b))$ est un singleton, car la structure dans la boîte b est de profondeur nulle, il s'agit d'un axiome. Donc $\mathcal{T}(\Theta(b)) = \{(; x, \overline{x})\}$ et $\{(p_1,\ldots,p_{n_b}) \mid p_i \in |\mathcal{T}(\Theta(b))|\} = \{((; x_1, \overline{x}_1), \ldots, (; x_{n_b}, \overline{x}_{n_b}))\}$. Ici, les renommages nécessaires sont faits de façon à ce que les p_i aient des atomes disjoints.

On a donc $\mathcal{T}(P) =$

$$\sum_{n_b \in \mathbf{N}} \frac{1}{n_b!} \sum_{\substack{(p_1,\ldots,p_{n_b}) \\ \in |\mathcal{T}(\Theta(b))|^{n_b}}} \prod_{j=1}^{n_b} \lambda_{p_j^b}(; a_1^b, a_0^b) \left[c(t_{1,1}^b, \ldots, t_{1,n_b}^b)/a_1^b\right] \left[!(t_{0,1}^b, \ldots, t_{0,n_b}^b)/a_1^b\right]$$

Or, ici $\lambda_{p_j^b} = 1$ pour tout j, et de plus, pour tout j, $t_{1,j}^b = x_j$ et $t_{0,j}^b = \overline{x}_j$, par construction. Donc on a

$$\mathcal{T}(P) = \sum_{n_b \in \mathbf{N}} \frac{1}{n_b!} (; c(x_1, \ldots, x_{n_b}), !(\overline{x}_1, \ldots, \overline{x}_{n_b}))$$

ce qui peut être illustré de la façon suivante :

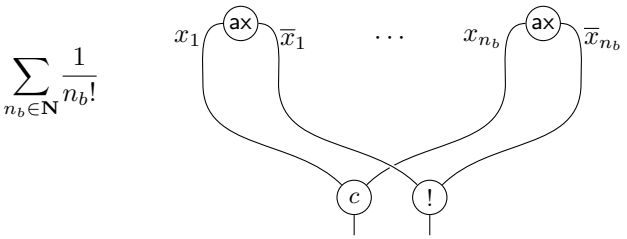
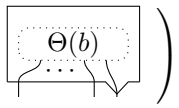


FIGURE 2.30 – Le développement d'un axiome emboîté

Arrêtons-nous également un instant sur un cas particulier du développement : le cas où une boîte comporte un nombre nul de copies. Nous illustrons ce cas en figure 2.31 de façon à illustrer la nécessité que l'on a de bien comprendre les mécanismes liés aux sauts et affaiblissements pour traiter le développement dans son ensemble.



contient toujours le réseau suivant :

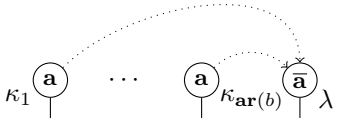


FIGURE 2.31 – Développement 0-aire d'une boîte

2.5.2 Initialisations

Maintenant que nous avons défini le développement de Taylor, nous devons nous assurer que les définitions des sections précédentes s'appliquent aux réseaux à ressources qui y figurent.

En effet, nous cherchons à appliquer les résultats précédents, en particulier le théorème 6. Pour cela, il nous faut procéder à une initialisation de nos preuves, dans le sens où l'on doit montrer que les conditions du théorème 6 sont vérifiées par toute structure p, dès lors qu'il existe $P \in \mathbf{MELL}$ tel que $p \in |\mathcal{T}(P)|$. C'est l'objet de cette section.

On fixe donc pour la suite un réseau $P = ((\overrightarrow{C}; \overrightarrow{T}), \Theta, S_P) \in \mathbf{MELL}$, et $p \in |\mathcal{T}(P)|$. On suppose que $B_\Theta = \{b_1, \ldots, b_r\}$, et que pour $n_{b_1}, \ldots, n_{b_r}$ donnés, $(p_1^{b_i}, \ldots, b_{n_{b_i}}^{b_i}) \in |\mathcal{T}(\Theta(b_i))|^{n_{b_i}}$ pour tout $b_i \in B_\Theta$. On nomme aussi σ la substitution qu'induit le développement, telle que $p = ((\overrightarrow{C}, (\overrightarrow{c}_i^b)_{b \in B_\Theta, 1 \leq i \leq n_b}; \overrightarrow{T})\sigma, S_p)$.

2.5.2.1 Chemins dans le développement

Ici, nous allons montrer que, dès lors que $p \in |\mathcal{T}(P)|$, $\mathbf{ln}(p)$ est borné en fonction de P. Pour cela, nous allons caractériser la forme des chemins dans p relativement à ceux de P.

Définition 23.
Si $t \in \mathbf{SA}(p)$, on remarque que :

- soit t est dans une copie de boîte, c'est-à-dire qu'il existe b, j tels que $t \in \mathbf{SA}(p_j^b)$. Dans ce cas on dit que t est un arbre *intérieur*.
- Soit $t = T\sigma$ pour $T \in \mathbf{SA}(P)$. Dans ce cas, on dit que t est un arbre *extérieur*, et on définit $t^* = T$, qui correspond à l'antécédent de t pour le développement.

On définit ensuite les *chemins intérieurs* comme les chemins de p constitués d'arbres intérieurs uniquement, et les *chemins extérieurs* comme les chemins de p constitués d'arbres extérieurs uniquement.

Remarquons déjà qu'il existe des chemins de p qui ne sont ni extérieurs ni intérieurs. Par exemple, si $s = c(t_1, \ldots, t_n)$ apparaît dans le développement, (s, t_i) est un chemin, mais s est extérieur et t_i est intérieur.

Nous allons nous efforcer d'associer à chaque chemin de p un chemin dans P. Pour cela, on commence par remarquer que l'on peut associer à tout interrupteur I de p un interrupteur I^* de P. On rappelle que les positions d'interrupteurs ne sont définies qu'à la profondeur courante. Si $I(\alpha(T_1\sigma, T_2\sigma)) = T_i\sigma$, alors on pose $I^*(T_1, T_2) = T_i$, pour $\alpha \in \{c, \parr\}$. Remarquons qu'il n'est pas besoin de considérer les contractions n-aires générées par le développement, car leurs antécédents sont des portes de boîtes, et les interrupteurs ne les concernent donc pas.

Si $c = \langle t | s \rangle$ est une coupure entre arbres extérieurs, on dit que c est extérieure, et on pose $c^* = \langle t^* | s^* \rangle$. De même avec les étiquettes d'adjacence l telles que $t \sim_l s$ si t et s sont extérieurs, et en posant $(\mathbf{a}_\kappa, \overline{\mathbf{a}}_\lambda)^* = b$ si $\mathbf{a}_\kappa^* = a_{i+1}^b$ et $\overline{\mathbf{a}}_\lambda^* = a_0 + 1$: on obtient $t^* \sim_{l^*} s^*$.

Lemme 24. Si ξ est un chemin extérieur dans p pour l'interrupteur I, alors ξ^* est un chemin dans P pour I^*.

Démonstration. On vérifie que si $t \sim_l^I s$, alors $t^* \sim_{l^*}^{I^*} s^*$. C'est évident si t est un sous-arbre de s, car l'inverse de la substitution σ préserve cette propriété, et la définition de I^* assure que les interrupteurs I^* suivent les adjacences de ξ. C'est également évident si t et s sont reliés par un axiome ou une coupure.

Pour les sauts, il reste à remarquer que si $S_p(\mathbf{a}_\kappa) = t$, et bien soit $\mathbf{a}_\kappa \in \mathbf{SA}(P)$, et dans ce cas $S_P(\mathbf{a}_\kappa) = t\sigma$, par définition, soit il existe $b \in B_\Theta$ tel que $n_b = 0$, et $\mathbf{a}^*_\kappa = a^b_{i+1}$. Dans ce cas, $S_p(\mathbf{a}_\kappa) = \overline{\mathbf{a}}_\lambda = a^b_0\sigma$, par la définition 22, et on a bien $\mathbf{a}^*_\kappa \sim_{l^*} \overline{\mathbf{a}}^*_\lambda$, car les portes de boîtes sont adjacentes entre elles : $a^b_{i+1} \sim_b a^b_0$. □

On s'intéresse maintenant aux chemins intérieurs, et on montre ci-dessous qu'ils ne peuvent apparaître que dans une seule copie de boîte.

Lemme 25. Si $\xi = t_1, \ldots, t_n$ est un chemin intérieur de p, alors il existe $k \in \{1, \ldots, r\}$ et $j \in \{1, \ldots, n_{b_k}\}$ tels que pour tout $i \in \{1, \ldots, n\}$, $t_i \in \mathbf{SA}(p^{b_k}_j)$.

Démonstration. Il suffit de remarquer que si $t \sim s$, pour t et s deux arbres intérieurs, alors t et s sont nécessairement sous-arbres du même $p^{b_k}_j$, ce qui s'étend à ξ. □

En particulier, on en déduit qu'un chemin intérieur dont les arbres sont dans un certain p^b_j est également un chemin de $\mathbf{Ch}(p^b_j)$.

On cherche maintenant à caractériser la forme générale des chemins de p. Ils sont constitués d'alternances entre chemins intérieurs et extérieurs, que nous allons mesurer. Regardons déjà par quels points un chemin extérieur peut se connecter à un chemin intérieur.

Lemme 26. Si $t \sim^{p,I} s$, avec t intérieur et s extérieur, alors il existe $b \in B_\Theta$ et $i \in \{0, \ldots, \mathbf{ar}(b)\}$ tels que $s = a^b_i\sigma = \alpha(t_{i,1}, \ldots, t_{i,n_b})$ pour $\alpha \in \{c, !\}$, et $t \in \{t_{i,1}, \ldots, t_{i,n_b}\}$.

Démonstration. Cette propriété se vérifie par cas, en se rappelant que les sauts ne peuvent pas passer d'un arbre intérieur à un arbre extérieur, par construction. □

On appelle *chemin de boîte* un chemin $t_1, \ldots, t_n$ tel que pour tout i, t_i est soit intérieur, soit un connecteur !.

Lemme 27. Soit ξ un chemin de boîte de p. Si ξ n'est pas intérieur, $\xi = \chi_1, t, \chi_2$, où il existe $b \in B_\Theta, i, j \in \{1, \ldots, n_b\}$ tels que $\chi_1 \in \mathbf{Ch}(p^b_i), \chi_1 \in \mathbf{Ch}(p^b_j)$ et $t = a^b_0\sigma$. De plus, $i \neq j$.

Démonstration. Par le lemme 26, si χ, t ou t, χ est un chemin de boîte, et que t est un arbre extérieur, alors il existe $b \in B_\Theta$ tel que $t = a^b_0\sigma$, et $i \in \{1, \ldots, n_b\}$ tel que $\chi \in \mathbf{Ch}(p^b_i)$.

Donc, il n'existe pas de chemin de boîte de la forme χ_1, t_1, χ_2, t_2 avec t_1, t_2 extérieurs. En effet comme t_1 et t_2 sont des connecteurs !, ils seraient égaux, et l'on aurait un cycle. Donc tout chemin de boîte est soit intérieur, soit de la forme χ_1, t, χ_2 avec $\chi_1 \in \mathbf{Ch}(p^b_i), \chi_1 \in \mathbf{Ch}(p^b_j)$ et $t = a^b_0\sigma$ (notons que χ_1 ou χ_2 peut être vide).

Dans ce cas, on peut également voir que $i \neq j$, car si l'on regarde $s \sim^p a^b_0\sigma = !(t^b_{0,1}, \ldots, t^b_{0,n_b})$, pour s un arbre intérieur, alors $s \in \{t^b_{0,1}, \ldots, t^b_{0,n_b}\}$. En effet, s ne peut pas être un atome dont le saut pointerait vers $a^b_0\sigma$, car par définition du développement, les seuls sauts de la sorte proviennent soit d'atomes $\gamma \in \mathbf{A}(P)$ tels que $S_P(\gamma) = a^b_0$, et alors γ est un arbre extérieur de p, ce qui contredit nos hypothèses ; soit il existe $\gamma = a^b_{i+1}\sigma$ tel que $S_p(\gamma) = a^b_0\sigma$, ce qui se produit si $n_b = 0$. Mais alors $a^b_0\sigma$ est un coaffaiblissement, ce qui contredit également nos

hypothèses. La seule possibilité restante est donc que s soit une prémisse du connecteur !.

Comme pour tout $i \in \{1, \dots, n_b\}$, il n'existe qu'une seule prémisse de $a_0^b\sigma$ dans $\mathbf{SA}(p_i^b)$, on en conclut que si $\xi = \chi_1, t, \chi_2$, alors $\xi = \chi_1' s_1, t, s_2', \chi_2$ pour $s_1 \in S(p_i^b), s_2 \in S(p_j^b)$ et pour $i \neq j$, par acyclicité. Comme χ_1 et χ_2 sont intérieurs, et par le lemme 25, on conclut. □

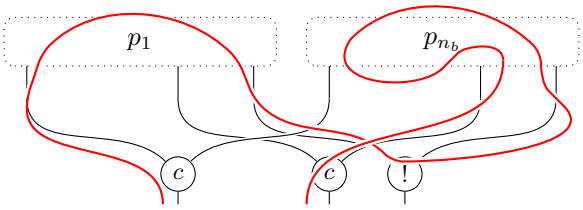


FIGURE 2.32 – Chemins de boîte dans le développement : cas critique

Corollaire 6. Soient $\xi \in \mathbf{Ch}(p) = \chi_1, \zeta, \chi_2$ pour χ_1, χ_2 extérieurs, où ζ est un chemin intérieur non vide pour une boîte b, alors $\xi = \chi_1' t_1, \zeta, t_2, \chi_2'$, où il existe $i, j \in \{0, \dots, \mathbf{ar}(b)\}$ tels que $t_1 = a_i^b\sigma$ et $t_2 = a_j^b\sigma$. Dans ce cas, on définit $\xi^* = \chi_1^*, \chi_2^* \in \mathbf{Ch}(P)$.

Démonstration. On vérifie que ξ^* est bien défini. $i \neq j$, par acyclicité de ξ, et $a_i^b \sim_b^P a_j^b$. □

On étend la construction ξ^* aux chemins de boîte. Si χ_1, ζ, χ_2 sont tels que χ_1 et χ_2 soient extérieurs et que ζ soit *maximal*, c'est-à-dire qu'il n'existe pas de chemin de boîte contenant ζ, alors soit ζ est intérieur, et on construit ξ^* comme dans le Corollaire 6, soit $\xi = \chi_1', t_1, \zeta_1, a_0^b\sigma, \zeta_2, t_2, \chi_2'$, où ζ_1 et ζ_2 sont intérieurs pour deux copies de boîtes distinctes. Dans ce cas, on pose $\xi^* = \chi_1^*, \chi_2^*$, en remarquant à nouveau que c'est bien défini car $t_1 = a_i^b\sigma$ et $t_2 = a_j^b\sigma$ pour $i \neq j$.

Ce point, relatif à la maximalité de ζ, est essentiel, car si l'on avait étendu simplement la construction du Corollaire 6, on aurait $\xi^* = \chi_1^*, a_0^b, \chi_2^*$, ce qui n'est pas un chemin de p, car contenant trois portes d'une même boîte consécutivement, la contrainte selon laquelle deux étiquettes d'adjacence identiques ne sont pas consécutives n'est pas respectée, et ξ^* ne satisfait alors pas à la définition des chemins.

On peut donc maintenant étendre aux chemins composés de plusieurs chemins de boîte. Si $\xi = \chi_1, \zeta_1, \dots, \zeta_n, \chi_{n+1}$ où les χ_i sont extérieurs et les ζ_j sont des chemins de boîte maximaux, alors $\xi^* = \chi_1^*, \dots, \chi_{n+1}^*$ est bien un chemin dans P. De plus, tout chemin d'extrémités extérieures admet naturellement une décomposition de la sorte, par conséquence des définitions.

Le lemme suivant établit que les chemins qui passent d'une copie de boîte p_i^b à une autre, p_j^b pour $i \neq j$, passent nécessairement par la cocontraction qui les relie, et sont donc des chemins de boîte. Pour cela, on montre que n'importe quel autre chemin induirait un chemin dans P entre deux portes d'une même boîte (différent du chemin trivial qui les relie par $\sim_b^P$), ce qui entraîne naturellement l'existence d'un cycle.

Lemme 28. Si $\xi \in \mathbf{Ch}(p)$ est d'extrémités $t_i^b \in \mathbf{SA}(p_i^b)$ et $s_j^b \in \mathbf{SA}(p_j^b)$ pour $i \neq j \in \{1, \ldots, n_b\}$, alors ξ est un chemin de boîte.

Démonstration. Par définition, $\xi = \chi_1, \zeta, \chi_2$ pour χ_1 et χ_2 des chemins intérieurs, respectivement dans p_i^b et dans p_j^b. Il s'agit donc de montrer ici que ζ ne peut être différent de $a_0^b\sigma$. Supposons qu'il en soit autrement.

On remarque dans un premier temps que ζ n'est pas un chemin intérieur, sinon il devrait être dans p_i^b et dans p_j^b à la fois. On peut donc poser $\xi = \chi_1', t_1, s_1, \zeta', s_2, t_2, \chi_2'$, avec s_1 et s_2 extérieurs. Par le lemme 26, il existe $k, l \in \{0, \ldots, \mathbf{ar}(b)\}$ tels que $s_1 = a_k^b\sigma$ et $s_2 = a_l^b\sigma$. Il est impossible que $k = l$, car alors, ζ' serait vide, par acyclicité, et $a_k^b\sigma = c(t_{k,1}^b, \ldots, t_{k,n_b}^b)$ (car par hypothèse on a alors $k \neq 0$) : cela implique en particulier, par le lemme 26 que $\xi = \chi_1', t_1, c(t_{k,1}^b, \ldots, t_{k,n_b}^b), t_2, \chi_2$ pour $t_1, t_2 \in \{t_{k,1}^b, \ldots, t_{k,n_b}^b\}$. Or, il n'existe aucune position d'interrupteur qui soit correcte pour ce chemin, manifestement.

Donc, $k \neq l$. On a donc un chemin ζ de $a_k^b\sigma$ à $a_l^b\sigma$. Comme les extrémités de ζ sont extérieures, ζ doit être de la forme $\zeta_1, \eta_1, \ldots, \eta_n, \zeta_{n+1}$, où les ζ_i sont extérieurs et les η_i sont des chemins de boîte maximaux. On peut donc appliquer les constructions qui suivent le Corollaire 6 pour poser $\zeta^* = \zeta_1^*, \ldots, \zeta_n^*$, qui est un chemin de P entre a_l^b et a_k^b. Or, ζ^* ne peut pas commencer par $a_l^b \sim_b^P a_k^b$, ni se terminer par $a_k^b \sim_b^P a_l^b$, car les étiquettes d'adjacence générées par la construction $()^*$ ne sont égales à b que si $n_b = 0$, par définition. Donc on peut construire le chemin ζ^*, a_k^b, car ζ se termine par a_l^b, et $a_l^b \sim_b^P a_k^b$, ce qui constitue donc un cycle dans P (idem avec le chemin a_l^b, ζ^*), et contredit alors nos hypothèses. Voir la figure 2.33, avec en rouge un chemin η qui entraîne l'existence d'un cycle, et en bleu la seule possibilité pour un chemin ζ de passage d'une copie de boîte à une autre. □

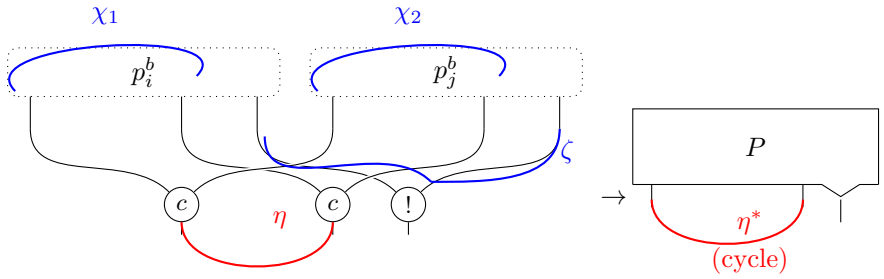


FIGURE 2.33 – Passage d'un chemin entre deux copies d'une même boîte

On s'approche d'une caractérisation générale des chemins dans p. Il reste à s'assurer qu'un chemin ne peut passer par plus de deux copies d'une même boîte.

Lemme 29. Soit $\xi \in \mathbf{Ch}(p)$. ξ peut s'écrire $\chi_1, \zeta_1, \ldots, \zeta_n, \chi_{n+1}$, où pour tous $i \neq j$, ζ_i et ζ_j sont des chemins de boîte maximaux pour deux boîtes distinctes.

Démonstration. Supposons par l'absurde qu'on ait ζ_i et ζ_j chemins de boîte pour la même boîte b. Ils ne peuvent être consécutifs par maximalité. Par ailleurs, si ζ_i, ξ', ζ_j est un sous-chemin de ξ avec ξ' non vide, alors par le lemme 28, on

obtient une contradiction, car ζ_i, ξ', ζ_j serait d'extrémités t_i^b et s_j^b, et serait donc un chemin de boîte. □

On peut enfin démontrer le théorème principal de cette section, qui correspond à l'initialisation de notre mesure sur les chemins. On rappelle que $\mathbf{taille}(P)$ désigne le nombre de sous-arbres de P à la profondeur courante, et $\mathbf{Ptaille}(P)$ désigne le nombre de sous-arbres à toutes profondeurs.

Théorème 7. $\mathbf{ln}(p) \leq 2^{\mathbf{prof}(P)}\mathbf{Ptaille}(P)$.

Démonstration. On montre par induction sur $\mathbf{prof}(P)$ que si $\xi \in \mathbf{Ch}(p)$, $\mathbf{ln}(\xi) \leq 2^{\mathbf{prof}(P)}\mathbf{Ptaille}(P)$. Si la profondeur de P est nulle, $p = P$ donc la propriété est vérifiée.

Sinon, soit $\pi + 1 = \mathbf{prof}(P)$. Pour tout $b \in B_\Theta$, $\Theta(b)$ est une structure de profondeur π. Donc si $\xi \in \mathbf{Ch}(p^b)$ pour $p^b \in |\mathcal{T}(\Theta(b))|$, on conclut par hypothèse d'induction.

S'il en est autrement, alors, grâce au lemme 29, on peut fixer la forme de ξ :

$$\xi = \chi_1, \zeta_1, \ldots, \zeta_n, \chi_{n+1}$$

avec pour tout i, (sans perte de généralité) ζ_i étant un chemin de boîte pour la boîte $b_i \in B_\Theta$, et χ_j un chemin extérieur pour tout j.

Mesurons d'abord les ζ_i. Par le lemme 27, soit ζ_i est interieur, et il existe donc $p_l^{b_i} \in |\mathcal{T}(\Theta(b_i))|$ tel que $\zeta_i \in \mathbf{Ch}(p_l^{b_i})$, et par hypothèse de récurrence, $\mathbf{ln}(\zeta_i) \leq 2^\pi\mathbf{Ptaille}(\Theta(b_i))$; soit il existe $k, l \in \{1, \ldots, n_{b_i}\}$ tels que $\zeta_i = \zeta_k, a_0^{b_i}\sigma, \zeta_l$ et $\zeta_k \in \mathbf{Ch}(p_k^{b_i}), \zeta_l \in \mathbf{Ch}(p_l^{b_i})$. Or, $p_k^{b_i}$ et $p_l^{b_i}$ appartiennent à $|\mathcal{T}(\Theta(b_i))|$. Donc par hypothèse d'induction, $\mathbf{ln}(\zeta_k)$ et $\mathbf{ln}(\zeta_l)$ sont inférieurs ou égaux à $2^\pi\mathbf{Ptaille}(\Theta(b_i))$, et on établit alors $\mathbf{ln}(\zeta_i) \leq 2(2^\pi\mathbf{Ptaille}(\Theta(b_i))) + 1$ ($+1$ car on rajoute l'arbre $a_0^{b_i}\sigma$ au compte de $\zeta_k + \zeta_l$). Posons alors $X \subseteq \{1, \ldots, n\} = \{i \mid \zeta_i \text{ est un chemin de boîte}\}$ et $Y = \{1, \ldots, n\} \backslash X$.

On peut poser par les considérations qui précèdent, en notant que $\mathbf{card}(X) + \mathbf{card}(Y) = n$, la propriété (A) :

$$\begin{aligned}
\sum_{i=1}^{n} \mathbf{ln}(\zeta_i) &= \sum_{j \in X} (2(2^\pi\mathbf{Ptaille}(\Theta(b_i))) + 1) + \sum_{j' \in Y} 2^\pi\mathbf{Ptaille}(\Theta(b_i)) \\
&= \mathbf{card}(X) + \sum_{j \in X} 2(2^\pi\mathbf{Ptaille}(\Theta(b_i))) + \sum_{j' \in Y} 2^\pi\mathbf{Ptaille}(\Theta(b_i)) \\
&\leq \mathbf{card}(X) + \sum_{i=1}^{n} 2(2^\pi\mathbf{Ptaille}(\Theta(b_i)))
\end{aligned}$$

Remarquons ensuite que (B) : $\sum_{j=1}^{n+1} \mathbf{ln}(\chi_j) \leq \mathbf{taille}(P) - \mathbf{card}(X)$. En effet, par construction de χ_j^* sur les chemins extérieurs, χ_j comporte exactement autant d'arbres que χ_j^* (en particulier car $()^*$ est injective pour les arbres extérieurs). Or, tous les arbres des χ_j sont distincts deux-à-deux, par acyclicité. De plus, pour tout chemin de boîte ζ_i quand $i \in X$, il existe un arbre $a_0^{b_i}\sigma \in \zeta_i$, donc ce dernier ne peut pas apparaître également dans les χ_j, également par acyclicité, d'où le fait que l'on omette $\mathbf{card}(X)$ des possibles arbres apparaissant dans les chemins extérieurs χ_j. Donc, la réunion des χ_j ne peut pas contenir plus d'arbres que n'en contient $\mathbf{SA}(P)$ (on se rappelle que $()^*$ est définie sur $\mathbf{SA}(P)$), ce qui prouve (B).

On peut donc poser :

$$\begin{aligned}
\mathbf{ln}(\xi) &\leq \sum_{j=1}^{n+1} \mathbf{ln}(\chi_i) + \sum_{i=1}^{n} \mathbf{ln}(\zeta_i) \\
&\leq \mathbf{taille}(P) - \mathbf{card}(X) + \sum_{i=1}^{n} \mathbf{ln}(\zeta_i) && \text{par (B)} \\
&\leq \mathbf{taille}(P) + \sum_{i=1}^{n} 2(2^{\pi}\mathbf{Ptaille}(\Theta(b_i))) && \text{par (A)}
\end{aligned}$$

Or, comme tous les b_i sont deux-à-deux distincts, par le lemme 29, on a $\{b_1, \ldots, b_n\} \subseteq B_\Theta$. De plus, $\mathbf{Ptaille}(P) = \mathbf{taille}(P) + \sum_{b \in B_\Theta} \mathbf{Ptaille}(\Theta(b))$, ce qui entraîne (C) : $\mathbf{Ptaille}(P) \leq \mathbf{taille}(P) + \sum_{i=1}^{n} \mathbf{Ptaille}(\Theta(b))$.

On peut donc conclure en continuant avec l'équation précédente :

$$\begin{aligned}
\mathbf{ln}(\xi) &\leq \mathbf{taille}(P) + \sum_{i=1}^{n} 2(2^{\pi}\mathbf{Ptaille}(\Theta(b_i))) \\
&\leq \mathbf{taille}(P) + (2^{\pi+1}) \sum_{i=1}^{n} \mathbf{Ptaille}(\Theta(b_i)) \\
&\leq 2^{\pi+1} \left(\mathbf{taille}(P) + \sum_{i=1}^{n} \mathbf{Ptaille}(\Theta(b_i)) \right) \\
&\leq 2^{\pi+1}\mathbf{Ptaille}(P) && \text{par (C)}
\end{aligned}$$

□

2.5.2.2 Degré de saut dans le développement

Nous essayons dans cette section de borner $\mathbf{deg}(p)$ en fonction de P, comme nous l'avons fait avec $\mathbf{ln}(p)$. La définition du développement de Taylor permet de montrer que le degré dans p est environ le même que celui de P. Environ, car le développement 0-aire d'une boîte b fait sauter $\mathbf{ar}(b)$ affaiblissements sur un coaffaiblissement, ce qui peut faire augmenter le degré.

Pour les sauts qui appartiennent à un p_i^b, ils pointent toujours vers un arbre de p_i^b de façon à ce que le degré soit préservé. Le nombre d'affaiblissements ou de co-unités de p n'est pas borné par P, car il est multiplié par n_b pour tout $b \in B_\Theta$. On doit montrer que S_p envoie chaque affaiblissement ou co-unité issu d'une duplication de boîte sur l'arbre qui correspond dans cette copie à celui vers lequel il sautait dans P. Par exemple, la boîte illustrée en figure 2.34

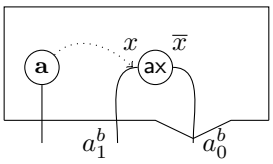


FIGURE 2.34 – Un axiome avec un saut emboîté

contient un saut. Les éléments de son développement de Taylor contiennent un nombre arbitraire d'affaiblissements, mais le degré est le même que celui du réseau de départ, comme l'illustre la figure 2.35. Le lemme suivant formalise ce raisonnement, en utilisant à nouveau la taille de P (à toute profondeur) comme borne supérieure.

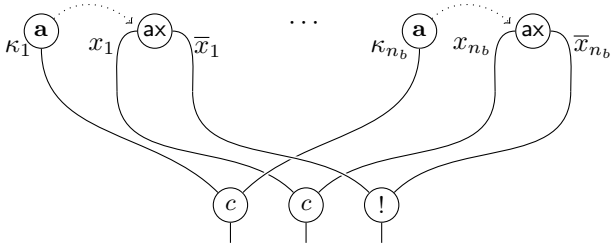


FIGURE 2.35 – Sauts dans le développement de Taylor

Lemme 30. $\mathbf{deg}(p) \leq \mathbf{Ptaille}(P)$.

Démonstration. On montre par induction sur la profondeur de P que $\mathbf{deg}(t) \leq \mathbf{Ptaille}(P)$ pour tout $t \in \mathbf{SA}(p)$. Si t est un arbre intérieur, alors on utilise l'hypothése d'induction. En effet, dans ce cas, $t \in p_i^b$, pour $b \in B_\Theta$ et $i \in \{1, \ldots, n_b\}$. Par définition du développement de Taylor, $S_p^{-1}(t) = S_{p_i^b}^{-1}$. Or, $\mathbf{Ptaille}(\Theta(b)) \leq \mathbf{Ptaille}(P)$.

Si t est extérieur, et que $S_p(\gamma) = t$, alors soit $\gamma \in \mathbf{SA}(P)$ et par définition, S_p coïncide avec S_P, soit $\gamma = a_{i+1}^b\sigma$ et $t = a_0^b\sigma$ (si $n_b = 0$). Donc $\mathbf{card}(S_p^-1(t)) \leq \mathbf{card}(\mathbf{U}_\perp(P) \cup \mathbf{U}_?(P) \cup \sum_{b\in B_\Theta} \mathbf{ar}(b)) \leq \mathbf{Ptaille}(P)$. □

2.6 Plongements

2.6.1 Introduction

Nous avons montré que le développement de Taylor des réseaux de **MELL** contenait des réseaux dont nos mesures sur les coupures et sur les sauts sont bornées en fonction du réseau de départ. Ce travail nous permet donc enfin d'appliquer les résultats de la section 2.3 au développement de Taylor.

Dans cette section, nous cherchons à faire essentiellement deux choses :

1. Définir une réduction entre combinaisons linéaires infinies, qui respecte certaines contraintes algébriques quand elle s'applique au développement de Taylor.
2. Montrer que cette réduction est adéquate pour simuler l'élimination des coupures de **MELL**.

Le premier point est rendu possible par les théorèmes principaux des sections précédentes. On verra que nous avons déjà tous les ingrédients pour définir une notion convenable de réduction. La contrainte algébrique que l'on impose est que les combinaisons obtenues par réduction ne comportent jamais que des coefficients finis. Nous appliquerons à cet effet le théorème 6 sur la finitude se l'antiréduction.

Le second point demandera une étude combinatoire de la réduction des réseaux à ressources, pour vérifier que si $P \to Q$ dans **MELL**, $\mathcal{T}(P)$ se réduira dans $\mathcal{T}(Q)$, avec les bons coefficients.

Mais il reste toutefois quelques réglages à apporter à nos constructions avant de définir la simulation. Notamment, la commutation entre élimination des coupures et développement de Taylor est vraie modulo un quotient sur les contractions et les cocontractions. Nous allons donc enrichir notre notion de réduction en y ajoutant des règles structurelles pour les connecteurs c et !. Ces réductions n'auront pas d'incidence capitale sur nos considérations, mais il faudra néanmoins s'assurer qu'elles n'interfèrent pas avec nos mesures de façon dommageable.

2.6.2 Expansions Rétoré

Les réductions des réseaux à ressources, si elles contiennent l'essentiel de la dynamique nécessaire, ne sont pas suffisantes à simuler l'élimination des coupures de **MELL** dans le développement de Taylor. En effet, il nous faut introduire quelques règles de réduction supplémentaires. Une version de ces règles est connue sous le nom de *réductions Rétoré*, et a été motivée par la simulation des règles de réduction du λ-calcul dans les réseaux qui, comme dans notre cas, ne pouvait être établie avec les seules réductions propres aux réseaux de preuves. Nous considérons des réductions allant (pour plusieurs règles), dans le sens opposé à celui habituel, d'où le terme d'*expansions Rétoré* (nous parlerons tout de même de règles de réduction). Nous contraignons ces expansions de façon à obtenir une notion de réduction suffisante pour notre objectif, mais qui préserve de bonnes propriétés combinatoires et géométriques.

Une particularité de ces nouvelles réductions est que, à la différence des réseaux à ressources, elles ne présentent pas de forme normale. En particulier, l'une des règles (et c'est déjà le cas pour les réductions Rétoré usuelles) consiste en une permutation, ce qui peut être répété à l'infini. Il faudra donc garder en tête que la forme normale de notre notion globale de réduction ne sera pas définie. Il faudra prendre soigneusement ce point en compte lors de la définition de la forme normale du développement de Taylor d'un réseau (nous discutons ce point en conclusion de ce chapitre).

La difficulté qui nous contraint à introduire cette nouvelle dynamique est que si l'on utilise les réductions à ressources, on parvient à montrer que quand $P \to Q$ dans **MELL**, les réseaux à ressources de $\mathcal{T}(P)$ se réduisent en des réseaux qui sont *presque* des réseaux de $\mathcal{T}(Q)$. C'est-à-dire que les structures obtenues sont équivalentes à celles de $\mathcal{T}(Q)$ modulo une relation standard définie sur la contraction, de façon à ce qu'elle soit associative, commutative, et qu'elle ait pour élément neutre l'affaiblissement (idem avec la cocontraction, mais ce n'est plus vrai avec le connecteur ! plus général). Nous ne pouvons pas traiter de telles classe d'équivalence, car elles comportent des arbre de taille non bornée, avec un nombre d'affaiblissements également non borné. C'est pour cette raison qu'on fait le choix de ne pas utiliser cette équivalence, mais de définir une réduction qui est comprise dans cette équivalence, et qui suffit à capturer les égalités qui nous intéressent.

Néanmoins, une tension persiste, car parmi les réductions qui nous permettent de compléter la simulation, l'une génère des affaiblissements en nombre arbitraire. Or, il faut nous assurer que notre nouvelle notion de réduction ne

rende pas incontrôlables les bornes sur le degré des sauts et la longueur des chemins. Nous avons alors besoin de contrôler la création de ces nouveaux affaiblissements.

La réduction qui pose problème permet de remplacer un affaiblissement $\mathbf{a}$ dans une structure par $c(\mathbf{a}_1, \ldots, \mathbf{a}_n)$ pour n'importe quel n. Pour éviter que cette réduction (que l'on appellera $\rightarrow_{\mathsf{r}_{\mathsf{neut}}}$, pour la neutralité de l'affaiblissement respectivement à la contraction) ne fasse augmenter le degré de saut (si tous les $\mathbf{a}_i$ sautent vers le même arbre) ou la longueur des chemins (si $\mathbf{a}_i$ saute vers $\mathbf{a}_{i+1}$ pour tout i, par exemple), nous conditionnons cette réécriture d'arbre à une élimination des coupures dans le réseau. C'est-à-dire que quand nous aurons besoin de créer ces affaiblissements, nous devrons nous assurer que nous aurons auparavant effectué une réduction qui permette de rediriger les sauts de façon satisfaisante. Cette assurance, elle nous sera donnée par l'existence préalable d'une réduction $\rightarrow_{\langle c|\overline{\mathbf{a}}\rangle}$ ailleurs dans le réseau.

Cette restriction, et la définition de cette réduction un peu particulière proviennent du fait que lorsque le développement de Taylor génère un nombre nul de copies de boîtes dans un réseau, la simulation que nous allons construire doit prendre en compte ce cas particulier où les affaiblissements apparaissent dans un agencement particulier. On peut voir plus loin la figure 2.42 pour comprendre en quoi la définition 24 est nécessaire et adaptée pour cette configuration.

Nous allons donc redéfinir $\rightarrow_{\langle c|\overline{\mathbf{a}}\rangle}$ en une réduction $\rightrightarrows^{\mathsf{neut}}_{\langle \overline{\mathbf{a}}|c\rangle}$ qui intègre dans une certaine mesure la neutralité de l'affaiblissement :

Définition 24 (Réduction $\rightrightarrows^{\mathsf{neut}}_{\langle \overline{\mathbf{a}}|c\rangle}$)**.** Soient $p \rightrightarrows_{\langle \overline{\mathbf{a}}|c\rangle} q$,telles que les coupures réduites de p soient $(\langle \overline{\mathbf{a}}_{\lambda_i}|c(t_{i,1}, \ldots, t_{i,n_i})\rangle)_{1\leq i\leq k}$. Soient également des ensembles $X_i \subseteq \{\mathbf{a}_{\kappa_{i,1}}, \ldots, \mathbf{a}_{\kappa_{i,l_i}} \mid \forall j : S_p(\mathbf{a}_{\kappa_{i,j}}) = \overline{\mathbf{a}}_{\lambda_i}\}$ pour tout $i \in \{1, \ldots, k\}$. On pose alors $p \rightrightarrows^{\mathsf{neut}}_{\langle \overline{\mathbf{a}}|c\rangle} q\sigma$ où σ est la substitution suivante :

$$\left(\left([c(\mathbf{a}_{\kappa_{i,j,1}}, \ldots, \mathbf{a}_{\kappa_{i,j,n_i}})]/\mathbf{a}_{\kappa_{i,j}}\right)_{1\leq i\leq k}\right)_{1\leq j\leq l_i}$$

où les $\mathbf{a}_{\kappa_{i,j,i'}}$ sont des nouveaux noms d'affaiblissements. Il faut également définir la fonction de saut dans $q\sigma$. Pour toute unité γ telle que $S_q(\gamma) = \mathbf{a}_{\kappa_{i,j}}$, on pose $S_{q\sigma}(\gamma) = \mathbf{a}_{\kappa_{i,j,1}}$.

Par définition de la réduction $\rightrightarrows_{\langle \overline{\mathbf{a}}|c\rangle}$, q contient pour tout $i \leq k$ les coupures $\langle \overline{\mathbf{a}}_{\lambda_{i,1}}|t_{i,1}\rangle, \ldots, \langle \overline{\mathbf{a}}_{\lambda_{i,n_i}}|t_{i,n_i}\rangle$ où les $\overline{\mathbf{a}}_{\lambda_{i,j}}$ sont des nouveaux coaffaiblissements. On pose donc pour tout affaiblissement créé $S_{q\sigma}(\mathbf{a}_{\kappa_{i,j,i'}}) = \overline{\mathbf{a}}_{\lambda_{i,i'}}$.

La situation est illustrée en figure 2.36.

On note que chaque X_i correspondent à un sous-ensemble de $\mathbf{E}(\overline{\mathbf{a}}_{\kappa_i})$, éventuellement vide (dans ce cas, la réduction est identique à $\rightrightarrows_{\langle \overline{\mathbf{a}}|c\rangle}$).

Les expansions Rétoré suivantes sont moins problématiques, mais il faut prendre tout de même à nouveau un soin particulier pour que l'on n'ait pas d'explosion des bornes ou d'effondrement de la taille. Cela se fait en n'autorisant la réduction que sur des arbres disjoints. Cela sera suffisant pour compléter la simulation du développement de Taylor.

Définition 25 (Expansions Rétoré)**.** On donne ensuite trois règles supplémentaires : $\rightarrow_{\mathsf{r}_{\mathsf{perm}}}$ pour la permutativité de la contraction et de !, $\rightarrow_{\mathsf{r}_{\mathsf{ass}}}$ pour l'associativité de la contraction, et $\rightarrow_{\mathsf{r}_{\mathsf{c}}}$ pour la contraction unaire redondante. Les réductions sont d'abord définies sur les arbres.

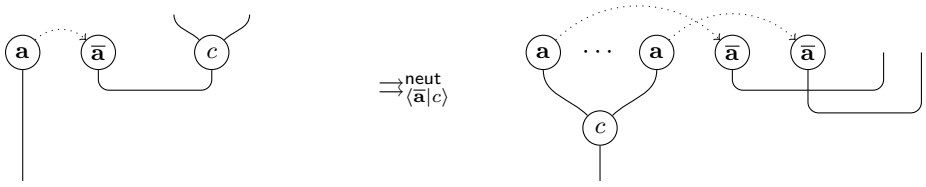


FIGURE 2.36 – Réduction $\rightrightarrows^{\mathsf{neut}}_{\langle\overline{\mathbf{a}}|c\rangle}$

— $\alpha(t_1,\dots,t_n) \to_{\mathsf{r_{perm}}} \alpha(t_{f(1)},\dots,t_{f(n)})$ pour $\alpha \in \{c,!\}$ et pour tout $f \in \mathfrak{S}_n$.
— $c(t_1,\dots,t_n) \to_{\mathsf{r_{ass}}} c(c(t_{1,1},\dots,t_{1,k_1}),\dots,c(t_{n,1},\dots,t_{n,k_n}))$ pour toutes les suites $(t_{1,1},\dots,t_{1,k_1}) :: \dots :: (t_{n,1},\dots,t_{n,k_n}) = (t_1,\dots,t_n)$.
— $c(t) \to_{\mathsf{r_c}} t$.

La réduction s'étend aux réseaux, en parallèle, mais d'une façon particulière : il est interdit d'opérer deux réductions dans le même arbre. Cette restriction permet, on le verra, de s'assurer que la taille des structures et le degré de sauts restent sous contrôle, tout en conservant la capacité de ces réductions à simuler l'élimination des coupures exponentielles. Ainsi, on définit la réduction $\rightrightarrows_{\mathsf{r}_\alpha}$ pour tout $\alpha \in \{\mathsf{perm},\mathsf{ass},\mathsf{c}\}$ ainsi : Soit p un réseau à ressources, et $t_1,\dots,t_n \in \mathbf{SA}(p)$ tels que pour tous $i \neq j \in \{1,\dots,n\}$, l'on ait $\mathbf{A}(t_i) \cap \mathbf{A}(t_j) = \emptyset$ (*i. e.* aucun arbre n'est sous-arbre d'un autre), et tels que pour tout $i \in \{1,\dots,n\}$, $t_i \to_{\mathsf{r}_\alpha} s_i$, alors, si $p = p'[t_1/x_1,\dots,t_n/x_n]$, on pose $p \rightrightarrows_{\mathsf{r}_\alpha} p'[s_1/x_1,\dots,s_n/x_n]$.

Il nous faut également déterminer l'évolution des sauts lors de ces réductions. Si $p = p'[c(t)/x] \rightrightarrows_{\mathsf{r_c}} q = q'[t/x]$, pour tout $\gamma \in \mathbf{SA}(p)$ tel que $S_p(\gamma) = c(t)$, on pose $S_{p'}(\gamma) = t$.

Pour les autres réductions, $\rightrightarrows_{\mathsf{r_{perm}}}$ et $\rightrightarrows_{\mathsf{r_{ass}}}$, les sauts sont redirigés de la façon triviale vers les arbres modifiés (S n'a plus le même codomaine car les arbres sont modifiés, donc quand $t \to_{\mathsf{r_{perm}}} t'$ ou $t \to_{\mathsf{r_{ass}}} t'$, les sauts qui pointaient vers t pointent dans le réduit vers t')

On définit enfin la réduction générale $p \rightrightarrows_{\mathsf{r}} q$ s'il existe p_1,p_2,p_3 tels que $p \rightrightarrows^{\mathsf{neut}}_{\langle\overline{\mathbf{a}}|c\rangle} p_1 \rightrightarrows_{\mathsf{r_{ass}}} p_2 \rightrightarrows_{\mathsf{r_{perm}}} p_3 \rightrightarrows_{\mathsf{r_c}} q$.

Nous devons maintenant nous assurer que ces nouvelles réductions n'altèrent pas les propriétés des structures qui nous intéressent, c'est-à-dire que les résultats concernant les bornes sur **ln** et **deg** doivent pouvoir être adaptées, ainsi que le résultat de finitude de l'antiréduction. Les trois lemmes suivants attestent de l'aspect inoffensif des expansions Rétoré sur les mesures qui nous concernent.

Lemme 31. Soient p et q des réseaux à ressources tels que $p \rightrightarrows_{\mathsf{r}} q$. $\mathbf{ln}(q) \leq 4\mathbf{ln}(p)$.

Démonstration. Il nous faut ici considérer les réductions $\rightrightarrows_{\mathsf{r_{ass}}}, \rightrightarrows_{\mathsf{r_c}}$, et $\rightrightarrows^{\mathsf{neut}}_{\langle\overline{\mathbf{a}}|c\rangle}$ (il est évident que $\rightrightarrows_{\mathsf{r_{perm}}}$ ne modifie pas la longueur des chemins). Nous devons vérifier qu'aucune de ces réductions n'a pu générer de chemins passant par de nouveaux fragments dans le réseau. On montre que chacune d'elles peut, au plus, doubler la longueur des chemins.

Pour $\rightrightarrows_{\mathsf{r_{ass}}}$, il nous suffit d'observer que dans le cas d'une réduction $s = c(t_1,\dots,t_n) \to_{\mathsf{r_{ass}}} s' = c(c(t_{1,1},\dots,t_{1,k_1}),\dots,c(t_{n,1},\dots,t_{n,k_n}))$, pour tout chemin passant par $s', c(t_{i,1},\dots,t_{i,k_i}), t_{i,j}$ dans le réduit, il existe une position d'interrupteurs I dans l'antiréduit telle que $s \sim^I t_{i,j}$. En particulier, à tout

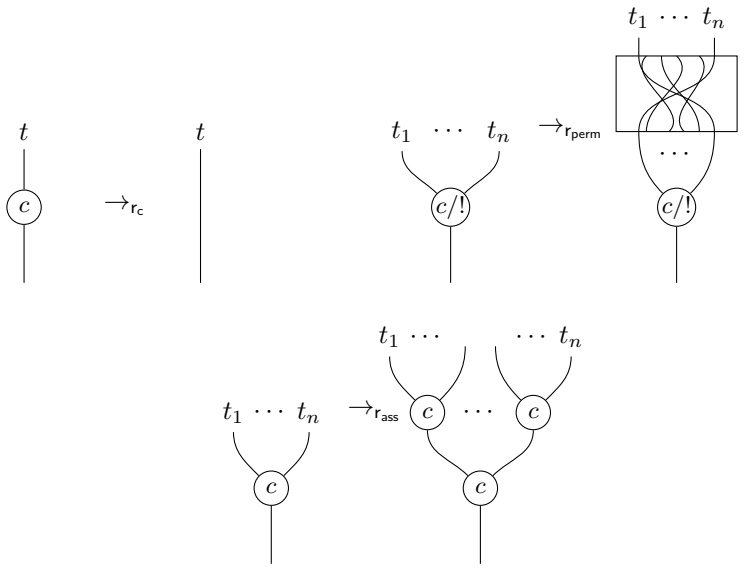


FIGURE 2.37 – Expansions Rétoré

chemin $\chi_1, s, t_{i,j}, \chi_2$ dans l'antiréduit, on peut associer un chemin de la forme $\chi'_1, s', c(t_{i,1}, \ldots, t_{i,k_i}), t_{i,j}, \chi'_2$ dans le réduit, où χ'_1 et χ'_2 sont obtenus de la même façon, récursivement. Par hypothèse de récurrence, $\mathbf{ln}(\chi'_i) \leq 2\mathbf{ln}(\chi_i)$, ce qui permet de conclure pour cette réduction.

Pour $\rightrightarrows_{\mathsf{r_c}}$, il suffit également d'observer que dans le cas d'une réduction $c(t) \to_{\mathsf{r_c}} t$, les nouveaux chemins dans le réduit sont ceux qui passent par un affaiblissements ou une co-unité dont le saut pointait vers $c(t)$ dans l'antiréduit, et pointe vers t dans le réduit. On constate à nouveau que tous les chemins de la sorte passant par γ, t dans le réduit correspondent dans l'antiréduit aux chemins passant par $\gamma, c(t)$. On construit donc aisément à partir de tout chemin dans le réduit un chemin dans l'antiréduit passant par un arbre de moins, tout au plus, ce qui conclut également pour cette réduction.

Pour $\rightrightarrows^{\mathsf{neut}}_{\langle \overline{\mathbf{a}}|c\rangle}$, rappelons-nous que si $p \rightrightarrows^{\mathsf{neut}}_{\langle \overline{\mathbf{a}}|c\rangle} q$, il existe par définition p' tel que $p \rightrightarrows_{\langle \overline{\mathbf{a}}|c\rangle} p'$ et $q = p'\sigma$, pour une substitution d'arbres σ. On a déjà vu au théorème 3 que dans ce cas $\mathbf{ln}(p') \leq \mathbf{ln}(p)$. En inspectant la définition de la réduction $\rightrightarrows^{\mathsf{neut}}_{\langle \overline{\mathbf{a}}|c\rangle}$ (voir aussi figure 2.36), on observe que le seul endroit ou les chemins sont modifiés sont les chemins de la forme $\xi = \chi_1, \mathbf{a}_\kappa, \overline{\mathbf{a}}_\lambda, c(t_1, \ldots, t_n), \chi_2$ dans p, quand $S_p(\mathbf{a}_\kappa) = \overline{\mathbf{a}}_\lambda$, et $\langle \overline{\mathbf{a}}_\lambda | c(t_1, \ldots, t_n)\rangle$ est une coupure réduite de p. Dans ce cas, p' a pour nouvelles coupures $\langle \overline{\mathbf{a}}_{\lambda_i} | t_i\rangle$ $(1 \leq i \leq n)$, et σ comporte la substitution $[c(\mathbf{a}_{\kappa_1}, \ldots, \mathbf{a}_{\kappa_n})/\mathbf{a}_\kappa]$, où $S_{q\sigma}(\mathbf{a}_{\kappa_i}) = \overline{\mathbf{a}}_{\lambda_i}$. On peut alors associer à ξ le chemin $\chi'_1, c(\mathbf{a}_{\kappa_1}, \ldots, \mathbf{a}_{\kappa_n}), \mathbf{a}_{\kappa_i}, \overline{\mathbf{a}}_{\kappa_i}, t_i, \chi'_2$, où χ'_1 et χ'_2 sont inductivement obtenus de la même façon à partir de χ_1 et de χ_2, et où i est choisi en fonction de χ_2 (si χ_2 commence par un t_j, on prend $i = j$, si χ_2 commence par un atome dont le saut pointait vers $c(t_1, \ldots, t_n)$, alors on prend $i = 1$ car le saut a été redirigé vers t_1). Les chemins ont été légèrement modifiés, mais on observe qu'ils font la même longueur qu'auparavant. On en conclut que si $p \rightrightarrows^{\mathsf{neut}}_{\langle \overline{\mathbf{a}}|c\rangle} q$, $\mathbf{ln}(q) = \mathbf{ln}(p)$.

On conclut donc en composant les résultats ci-dessus que les chemins peuvent au plus quadrupler en longueur pendant les expansions Rétoré. Nous devons maintenant vérifier que le degré de saut n'explose pas pendant ces réductions. □

Lemme 32. Soient p et q des réseaux à ressources tels que $p \rightrightarrows_{\mathsf{r}} q$. $\mathbf{deg}(q) \leq 4\mathbf{deg}(p)$.

Démonstration. Les réductions qui modifient les sauts de façon non triviale sont $\rightrightarrows^{\mathsf{neut}}_{\langle \overline{\mathbf{a}}|c\rangle}$ et $\rightrightarrows_{\mathsf{r_c}}$. Pour la première, soit $p \rightrightarrows^{\mathsf{neut}}_{\langle \overline{\mathbf{a}}|c\rangle} q$.considérons la structure p' telle que $p \rightrightarrows_{\langle \overline{\mathbf{a}}|c\rangle} p'$ et $q = p'\sigma$ pour une substitution σ. Par le lemme 21, $\mathbf{deg}(p') \leq 2\mathbf{deg}(p)$. Il reste à vérifier que le degré n'augmente pas entre p' et q, alors même que des affaiblissements sont créés. Il suffit pour cela d'inspecter la définition 24 pour observer que le nombre d'affaiblissements qui sautent sur un coaffaiblissement dans q est au plus égal au nombre d'affaiblissements qui sautaient sur un coaffaiblissement dans p (voir figure 2.36).

La seconde double le degré. Soient $p = p'[c(t_1)/x_1, \ldots, c(t_n)/x_n] \rightrightarrows_{\mathsf{r_c}} q = p'[t_1/x_1, \ldots, t_n/x_n]$. Par la contrainte demandant que les t_i soient disjoints, il n'existe aucune paire $i, j \in \{1, \ldots, n\}$ tels que $t_i = c(t_j)$. On en conclut que pour tout $i \in \{1, \ldots, n\}, \mathbf{deg}_q(t_i) = \mathbf{deg}_p(t_i) + \mathbf{deg}_p(c(t_i)) \leq 2\mathbf{deg}(p)$. C'est cette contrainte qui permet d'éviter l'explosion du degré de saut, car sans elle, on pourrait avoir des réductions de la forme $c(c(c(\ldots(c(t))))) \rightarrow_{\mathsf{r_{neut}}} t$, ce qui serait catastrophique □

C'est la contrainte sur les arbres disjoints qui permet d'éviter l'explosion du degré de saut, car sans elle, on pourrait avoir des réductions de la forme $c(c(c(\ldots(c(t))))) \rightarrow_{\mathsf{r_{neut}}} t$, ce qui serait catastrophique, car cela nous empêcherait de borner à la fois l'évolution du degré de saut, et la diminution de la taille dans le lemme suivant.

Lemme 33. Soient p et q des réseaux à ressources tels que $p \rightrightarrows_{\mathsf{r}} q$. $\mathbf{taille}(p) \leq 3\mathbf{taille}(q)$.

Démonstration. 15

Les expansions Rétoré qui font diminuer la taille du réseau sont $\rightrightarrows_{\mathsf{r_c}}$ (voir paragraphe précédent) et éventuellement $\rightrightarrows^{\mathsf{neut}}_{\langle \overline{\mathbf{a}}|c\rangle}$: dans ce cas, si la taille diminue lors d'une réduction $p \rightrightarrows^{\mathsf{neut}}_{\langle \overline{\mathbf{a}}|c\rangle} q$, on vérifie immédiatement que $p \rightrightarrows_{\langle \overline{\mathbf{a}}|c\rangle} q$, c'est-à-dire que l'on ne crée pas de nouveaux affaiblissements. Alors, par le lemme 15, on a $\mathbf{taille}(p) \leq \frac{3}{2}\mathbf{taille}(q)$.

En ce qui concerne la réduction $\rightrightarrows_{\mathsf{r_c}}$, comme nous l'avons vu lors de la preuve du lemme précédent, si $p = p'[c(t_1)/x_1, \ldots, c(t_n)/x_n] \rightrightarrows_{\mathsf{r_c}} q = p'[t_1/x_1, \ldots, t_n/x_n]$, il n'existe aucune paire $i, j \in \{1, \ldots, n\}$ tels que $t_i = c(t_j)$. Donc la taille de p est égale à la taille de q plus n, ce qui permet d'établir que l'on a alors $\mathbf{taille}(p) \leq 2\mathbf{taille}(q)$. On peut donc conclure, car les autres réductions Rétoré augmentent la taille (pour $\rightarrow_{\mathsf{r_{ass}}}$) ou la laissent inchangée (pour $\rightarrow_{\mathsf{r_{perm}}}$). □

On peut utiliser ce résultat pour adapter le théorème 6 à une notion de réduction plus large que celle des réseaux à ressources.

Définition 26.
Soient p, q deux réseaux à ressources tels qu'il existe p', et tels que $p \rightrightarrows p' \rightrightarrows_{\mathsf{r}} q$. On écrit alors $p \rightrightarrows_{(r)} q$. De même, si $p \gg p' \rightrightarrows_{\mathsf{r}} q$, on écrit $p \gg_{(r)}$.

Corollaire 7. Soit $X = \{p \in \mathbf{DLL}_0 \mid \mathbf{deg}(p) \leq n, \mathbf{ln}(p) \leq m\}$ un ensemble de réseaux à ressources, et q un réseau à ressources. $\{p \in X \mid p \gg_{(r)} q\}$ est un ensemble fini.

Démonstration. La preuve s'obtient par une adaptation simple du théorème 6 à l'aide du lemme 33. □

Corollaire 8. Soient $p \gg_{(r)} q$. Il existe deux fonctions f, g telles que $\mathbf{ln}(q) = f(\mathbf{ln}(p))$ et $\mathbf{deg}(q) = g(\mathbf{deg}(p))$.

Démonstration. Composition des lemmes 23, 31, 32 et du théorème 4. □

2.6.3 Une réduction entre séries infinies

Nous allons pouvoir définir la notion de réduction qui nous permettra de simuler l'élimination des coupures de **MELL** dans le développement de Taylor. Cette réduction concerne des séries infinies de réseaux, et nous devons nous assurer qu'elle est toujours bien définie. Cette assurance est la raison d'être du théorème 6, et avec lui d'une grande partie des travaux du présent chapitre.

Définition 27.
Soit $\tau \in \mathbf{S}_0^{\mathbf{DLL}}$. Si les ensembles $\{\mathbf{ln}(p) \mid p \in |\tau|\}$ et $\{\mathbf{deg}(p) \mid p \in |\tau|\}$ sont bornés, alors on dit que τ est une combinaison *décente*.

Proposition 1. Soient $\tau = \sum_{i\in I} \lambda_i p_i \in \mathbf{S}^{\mathbf{DLL}_0}$ une combinaison décente, et $(\phi_i)_{i\in I}$ une famille de réseaux de $\mathbf{DLL}_0^+$ telle que $p_i \rightrightarrows_{(r)} \phi_i$ pour tout $i \in I$.

1. Pour tout $q \in \cup_{i\in I}|\phi_i|$, $\{p \in |\tau| \mid p \gg_{(r)} q\}$ est fini.
2. $\sum_{i\in I} \lambda_i \phi_i \in \mathbf{S}_0^{\mathbf{DLL}}$.
3. $\sum_{i\in I} \lambda_i \phi_i$ est une combinaison décente.

Démonstration. Le premier point est une conséquence directe du corollaire 7.

Le deuxième point est une conséquence du premier, le semi anneau **S** étant stable par sommes finies.

Le troisième point est une conséquence du corollaire 8. □

Définition 28.
Soit $\tau = \sum_{i\in I} \lambda_i p_i \in \mathbf{S}^{\mathbf{DLL}_0}$ une combinaison décente. On écrit $\tau \Rightarrow \sum_{i\in I} \lambda_i \phi_i$ si $p_i \rightrightarrows \phi_i$ pour tout $i \in I$. De même, on écrit $\tau \Rightarrow_r \sum_{i\in I} \lambda_i q_i$ si $p_i \rightrightarrows_{\mathsf{r}} q_i$ pour tout i, et $\tau \Rightarrow_{(r)} \sum_{i\in I} \lambda_i \phi_i$ si $p_i \rightrightarrows_{(r)} \phi_i$ pour tout i.

Par la proposition 1, ces réductions sont bien définies sur $\mathbf{S}^{\mathbf{DLL}_0}$.

Remarque 5. $\sigma \Rightarrow_{(r)} \tau$ si et seulement s'il existe ρ telle que $\sigma \Rightarrow \rho \Rightarrow_r \tau$.

Proposition 2. Soit P un réseau de **MELL**. $\mathcal{T}(P)$ est une combinaison décente. En particulier, $\Rightarrow_{(r)}$ est toujours bien définie sur le développement de Taylor.

Démonstration. Il suffit de rappeler les résultats de la section 2.5.2 qui établissent que tous les éléments du développement de Taylor d'un réseau voient leurs mesures **deg** et **ln** bornées uniformément en fonction de P. □

Le lemme suivant établit que la réduction passe au contexte, dans le sens où si une combinaison se réduit en une autre, on peut considérer cette réduction au sein de structures qui englobent les premières. C'est une version étendue de la propriété de localité des réductions dans les réseaux à ressources.

On rappelle qu'une variable x est libre dans une structure p si $\overline{x} \notin \mathbf{SA}(p)$.

Lemme 34 (Modularité de la réduction). Soient $(p_i)_{i\in I}$ une famille de réseaux à ressources telle que pour tout $i \in I, p_i = (\overrightarrow{c}_i; t_{i,1}, \ldots, t_{i,k})$ (tous les p_i ont le même nombre k de conclusions), et $(p'_j)_{j\in J}$ une famille telle que pour tout $i \in I, p'_i = (\overrightarrow{c}'_i; t'_{i,1}, \ldots, t'_{i,k})$.

Soient également $\tau = \sum_{i\in I} \lambda_i \cdot p_i$ et $\sigma = \sum_{j\in J} \lambda_j \cdot p'_j$ dans $\mathbf{S}_0^{\mathbf{DLL}}$ telles que $\tau \Rightarrow \sigma$, et soit $\psi = \sum_{l\in L} \lambda_l \cdot q_l \in \mathbf{S}_0^{\mathbf{DLL}}$ tel que pour tout $l \in L$, q_l soit une pré-structure comportant les k variables libres $x_{l,1}, \ldots, x_{l,k}$. Si $q_l = (\overrightarrow{d}; \overrightarrow{s})$, on écrit $q_l[p_i] = (\overrightarrow{c}_i, \overrightarrow{d}; \overrightarrow{s})[t_{i,1}/x_{l_1}, \ldots, t_{i,k}/x_{l,k}]$. De même pour $q_l[p'_j]$. On note que $q_l[p_i]$ est une structure, car elle ne comporte plus de variables libres.

On a :

$$\sum_{l\in L} \lambda_l \sum_{i\in I} \lambda_i \cdot q_l[p_i] \Rightarrow_{(r)} \sum_{l\in L} \lambda_l \sum_{j\in J} \lambda_j \cdot q_l[p'_j]$$

Démonstration. Par définition de la réduction parallèle entre séries $\Rightarrow_{(r)}$, pour tout $i \in I$, il existe $X_i \subseteq J$ tel que $p_i \rightrightarrows_{(r)} \sum_{j\in X_i} p'_j$. En particulier, $\sum_{j\in J} \lambda_j \cdot p'_j = \sum_{i\in I} \lambda_i \sum_{j\in X_j} p'_j$ (voir la définition 28).

Il suffit donc, par linéarité, de montrer que pour tout $l \in L$, $\sum_{i\in I} \lambda_i \cdot q_l[p_i] \Rightarrow_{(r)} \sum_{i\in I} \lambda_i \sum_{j\in X_i} \cdot q_l[p'_j]$. Pour cela, il suffit également de montrer que pour tout $l \in L$ et pour tout $i \in I$, $q_l[p_i] \rightrightarrows_{(r)} \sum_{j\in X_i} \cdot q_l[p'_j]$. On peut se contenter de montrer que c'est vrai pour la réduction simple $\rightarrow$, car le raisonnement pourra être itéré.

Ce résultat découle alors de la localité de la réduction. Par exemple, pour le cas où $p_i \rightarrow_m p'_j$, il est immédiat que $q_l[p_i] \rightarrow_m q_l[p'_j]$. En effet, $q_l[p_i]$ contient la coupure multiplicative dont les deux réduits sont dans $q_l[p'_j]$, par construction. On vérifie de la même façon que si $p_i \rightarrow \sum_{j\in X_i} p_j$, par une coupure quelconque ou par une expansion Rétoré, alors $q_l[p_i] \rightarrow \sum_{j\in X_i} q_l[p_j]$, par un raisonnement simple par cas sur chaque type de coupures que nous ne détaillerons pas. L'extension à la réduction parallèle est immédiate, elle découle de l'associativité de la somme (qui est valide dans tout semi-anneau).

□

2.6.4 Simulation de l'élimination des coupures

Au sujet de l'oubli des sauts Dans les propriétés de simulation que nous nous apprêtons à établir, toutes les informations relatives aux sauts sont perdues. En effet, nous allons montrer que lorsque dans **MELL**, $P \rightarrow Q$, on a $\mathcal{T}(P) \Rightarrow_{(r)} \mathcal{T}(Q)$, mais cette commutation entre développement de Taylor et élimination des coupures ne prend pas en compte les sauts. C'est-à-dire que $\mathcal{T}(P)$ se réduit en une combinaison équivalente à $\mathcal{T}(Q)$ modulo une réaffectation des sauts (une modification de l'image de la fonction de saut). Pour comprendre que cet oubli est inoffensif, il faut se rappeler que les fonctions de sauts sont des informations *externes* aux structures. On entend par là que l'existence d'une fonction de saut possédant certaines propriétés nous permet de déduire des

résultats combinatoires sur les structures (les résultats de finitude de l'antiréduction), mais que leurs définitions n'ont aucun impact sur l'aspect géométrique ou logique des réseaux : ces fonctions et leur redirection sont d'ailleurs définies de façon très arbitraire. Savoir que la combinaison de départ $\mathcal{T}(P)$ peut être munie d'une fonction de saut avec une borne sur le degré nous permet de nous assurer que la réduction $\Rightarrow$ est bien définie. Une fois cette assurance établie, nous pouvons oublier ce qu'il advient des sauts, et la commutation concerne la géométrie des réseaux, leur structure d'arbres, et non les sauts qu'il faut voir désormais comme un outil de preuve qui nous à permis de parvenir aux définitions nécessaires, à savoir une notion de réduction satisfaisante entre combinaisons infinies de réseaux à ressources.

Dans les preuves qui suivent, on notera $(;s_1,\dots,s_n,s)_k \in |\mathcal{T}(\Theta(b))|^k$ pour désigner $((;s_1^1,\dots,s_n^1,s^1),\dots,(;s_1^k,\dots,s_n^k,s^k)) \in |\mathcal{T}(\Theta(b))|^k$, de façon à rendre plus lisibles les équations à venir.

Lemme 35. Soient P et Q des réseaux de **MELL** tels que $P \to_{\langle a_0^b|d\rangle} Q$. $\mathcal{T}(P) \Rightarrow_{(r)} \mathcal{T}(Q)$.

Démonstration. On pose dans un premier temps $P = ((\langle a_0^b|d(S)\rangle, \overrightarrow{C}; \overrightarrow{U}), \Theta)$, et $Q = ((\langle T|S\rangle, \overrightarrow{C}, \overrightarrow{D}; \overrightarrow{U}), \Xi \cup \Theta')([T_i/a_i^b])_{1\le i\le n}$, avec le contexte de la boîte b $\Theta(b) = ((\overrightarrow{D}; T_1,\dots,T_n,T),\Xi)$ et $\Theta' = \Theta/\{b \mapsto \Theta(b)\}$. Soit $B_\Theta = \{b_1,\dots,b_k,b\}$, pour tout $i \in \{1,\dots,k\}$, on pose

$$\mathcal{T}(\Theta(b_i)) = \sum_{j\in J_i} \lambda_{i,j}\cdot p_{i,j} = (\overrightarrow{d}_{i,j}; t_{j,1},\dots,t_{j,n_i},t_j)$$

et la substitution $\sigma_{i,j} = [t_{j,1}/a_1^{b_i},\dots,t_{j,n_i}/a_{n_i}^{b_i}, t_j/a_0^{b_i}]$. Par définition du développement de Taylor, on a $\mathcal{T}(P) =$

$$\sum_{\substack{(j_1,\dots,j_k)\\ \in J_1\times\dots\times J_k}} \prod_{i=0}^{k} \lambda_{i,j_i} \sum_{l\in\mathbf{N}} \frac{1}{l!} \sum_{\substack{(\overrightarrow{c};t_1,\dots,t_n,t)_l\\ \in|\mathcal{T}(\Theta(b))|^l}} \Big(\overrightarrow{c}^1,\dots,\overrightarrow{c}^l,\overrightarrow{d}_{1,j_1},\dots,\overrightarrow{d}_{k,j_k},\langle !(t^1,\dots,t^l)|d(S)\rangle,\overrightarrow{C};\overrightarrow{U}\Big) \big[c(t_1^1,\dots,t_1^l)/a_1^b,\dots,c(t_n^1,\dots,t_n^l)/a_n^b\big]\,\sigma_{1,j_1},\dots,\sigma_{k,j_k}$$

Or, dans $\mathbf{DLL}_0$, $\langle d(t)|!(t^1,\dots,t^l)\rangle \to 0$ si $l \neq 1$. Donc les arguments de la somme ci-dessus qui ont un réduit sont ceux pour lesquels $l = 1$, c'est-à-dire ceux où le contenu de la boîte b n'a été copié qu'une seule fois. En particulier, seuls les arguments de la combinaison ci-dessous produisent une réduction non nulle :

$$\sum_{\substack{(j_1,\dots,j_k)\\ \in J_1\times\dots\times J_k}} \prod_{i=0}^{k} \lambda_{i,j_i} \sum_{\substack{(\overrightarrow{c};t_1,\dots,t_n,t)\\ \in|\mathcal{T}(\Theta(b))|}} \Big(\overrightarrow{c},\overrightarrow{d}_{1,j_1},\dots,\overrightarrow{d}_{k,j_k},\langle !(t)|d(S)\rangle,\overrightarrow{C};\overrightarrow{U}\Big) \big[c(t_1)/a_1^b,\dots,c(t_n)/a_n^b\big]\,\sigma_{1,j_1},\dots,\sigma_{k,j_k}$$

En appliquant donc les réductions de la forme $\langle d(t)|!(s)\rangle \to \langle t|s\rangle$, et en se rappelant que $c(t) \to_{\mathsf{r_c}} t$, on peut écrire :

$$\begin{aligned}
\mathcal{T}(P) \Rightarrow & \sum_{\substack{(j_1,\ldots,j_k)\\ \in J_1\times\ldots\times J_k}} \prod_{i=0}^{k} \lambda_{i,j_i} \sum_{\substack{(\overrightarrow{c};t_1,\ldots,t_n,t)\\ \in|\mathcal{T}(\Theta(b))|}} \left(\overrightarrow{c}, \overrightarrow{d}_{\,1,j_1}, \ldots, \overrightarrow{d}_{\,k,j_k}, \langle t|S\rangle, \overrightarrow{C}; \overrightarrow{U}\right) \\
& \qquad \left[c(t_1)/a_1^b, \ldots, c(t_n)/a_n^b\right] \sigma_{1,j_1}, \ldots, \sigma_{k,j_k} \\
\Rightarrow_r & \sum_{\substack{(j_1,\ldots,j_k)\\ \in J_1\times\ldots\times J_k}} \prod_{i=0}^{k} \lambda_{i,j_i} \sum_{\substack{(\overrightarrow{c};t_1,\ldots,t_n,t)\\ \in|\mathcal{T}(\Theta(b))|}} \left(\overrightarrow{c}, \overrightarrow{d}_{\,1,j_1}, \ldots, \overrightarrow{d}_{\,k,j_k}, \langle t|S\rangle, \overrightarrow{C}; \overrightarrow{U}\right) \\
& \qquad \left[t_1/a_1^b, \ldots, t_n/a_n^b\right] \sigma_{1,j_1}, \ldots, \sigma_{k,j_k} \\
= & \mathcal{T}(Q)
\end{aligned}$$

En effet, les boîtes de Q sont précisément celles de $\Theta(b)$, du contexte Ξ, et $\{b_1, \ldots, b_k\}$. Le mécanisme général à l'œuvre dans cette réduction est illustré en figure 2.38 □

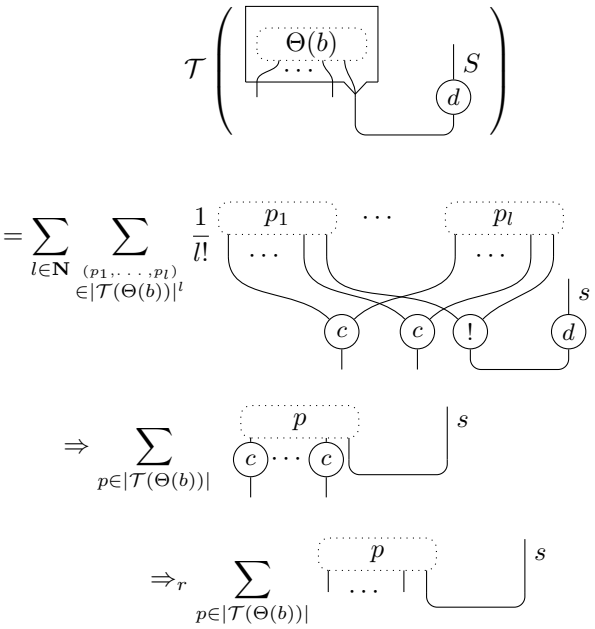


FIGURE 2.38 – Lemme 35 : simulation de la réduction promotion/déréliction

Lemme 36. Soient P et Q des réseaux de **MELL** tels que $P \to_{\langle a_0|\mathbf{a}\rangle} Q$. $\mathcal{T}(P) \Rightarrow \mathcal{T}(Q)$.

Démonstration. Posons dans un premier temps $P = (\langle a_0^b|\mathbf{a}_\kappa\rangle, \overrightarrow{C}; \overrightarrow{U}), \Theta)$, et $Q = ((\overrightarrow{C}; \overrightarrow{U}), \Theta')([\mathbf{a}_{\kappa_i}/a_i^b])_{1\le i\le n}$ où $\mathbf{ar}(b) = n$. Soit $B_\Theta = \{b_1, \ldots, b_k, b\}$, pour tout

$i \in \{1, \ldots, k\}$, on pose

$$\mathcal{T}(\Theta(b_i)) = \sum_{j \in J_i} \lambda_{i,j} \cdot p_{i,j} = (\overrightarrow{d}_{i,j}; t_{j,1}, \ldots, t_{j,n_i}, t_j)$$

et la substitution $\sigma_{i,j} = [t_{j,1}/a_1^{b_i}, \ldots, t_{j,n_i}/a_{n_i}^{b_i}, t_j/a_0^{b_i}]$. Par définition du développement de Taylor, on a $\mathcal{T}(P) =$

$$\sum_{\substack{(j_1,\ldots,j_k) \\ \in J_1 \times \ldots \times J_k}} \prod_{i=0}^{k} \lambda_{i,j_i} \sum_{l \in \mathbf{N}} \frac{1}{l!} \sum_{\substack{(\overrightarrow{c}; t_1, \ldots, t_n, t)_l \\ \in |\mathcal{T}(\Theta(b))|^l}} \left(\overrightarrow{c}^1, \ldots, \overrightarrow{c}^l, \overrightarrow{d}_{1,j_1}, \ldots, \overrightarrow{d}_{k,j_k}, \langle !(t^1, \ldots, t^l) | \mathbf{a}_\kappa \rangle, \overrightarrow{C}; \overrightarrow{U} \right)$$
$$\left[c(t_1^1, \ldots, t_1^l)/a_1^b, \ldots, c(t_n^1, \ldots, t_n^l)/a_n^b \right] \sigma_{1,j_1}, \ldots, \sigma_{k,j_k}$$

Or, dans $\mathbf{DLL}_0$, $\langle \mathbf{a}_\kappa | !(t^1, \ldots, t^l) \rangle \to 0$ si $l \neq 0$. Donc les arguments de la somme ci-dessus qui ont un réduit sont ceux pour lesquels $l = 0$, c'est-à-dire ceux où le contenu de la boîte b a été copié zéro fois. Dans ce cas, des affaiblissements et un coaffaiblissement sont créés pour remplacer les portes de la boîte, et les coupures $\overrightarrow{c}^i$ n'apparaissent plus. De façon similaire à la démonstration du lemme 35, seuls les arguments de la combinaison ci-dessous produisent une réduction non nulle :

$$\sum_{\substack{(j_1,\ldots,j_k) \\ \in J_1 \times \ldots \times J_k}} \prod_{i=0}^{k} \lambda_{i,j_i} \left(\overrightarrow{d}_{1,j_1}, \ldots, \overrightarrow{d}_{k,j_k}, \langle \overline{\mathbf{a}}_\lambda | \mathbf{a}_\kappa \rangle, \overrightarrow{C}; \overrightarrow{U} \right)$$
$$\left[\mathbf{a}_{\kappa_1}/a_1^b, \ldots, \mathbf{a}_{\kappa_n}/a_n^b \right] \sigma_{1,j_1}, \ldots, \sigma_{k,j_k}$$

En appliquant donc les réductions de la forme $\langle \overline{\mathbf{a}} | \mathbf{a} \rangle \to \epsilon$, on peut écrire : $\mathcal{T}(P) \Rightarrow$

$$\sum_{\substack{(j_1,\ldots,j_k) \\ \in J_1 \times \ldots \times J_k}} \prod_{i=0}^{k} \lambda_{i,j_i} \left(\overrightarrow{d}_{1,j_1}, \ldots, \overrightarrow{d}_{k,j_k}, \overrightarrow{C}; \overrightarrow{U} \right) \left[\mathbf{a}_{\kappa_1}/a_1^b, \ldots, \mathbf{a}_{\kappa_n}/a_n^b \right] \sigma_{1,j_1}, \ldots, \sigma_{k,j_k}$$
$$= \mathcal{T}(Q)$$

Le mécanisme général à l'œuvre dans cette réduction est illustré en figure 2.39 □

Lemme 37. Soient X un ensemble, et $\overrightarrow{a}_1, \ldots, \overrightarrow{a}_k$ des suites d'éléments de x telles que $\overrightarrow{a}_i = (a_{i,1}, \ldots, a_{i,n_i})$ et telles que pour tous $(i,j) \neq (i',j')$, $a_{i,j} \neq a_{i',j'}$, et soit $n = \sum_{i=1}^{k} n_i$.

$$\mathbf{card}\{\overrightarrow{a} \in X^n \mid (\overrightarrow{a}_1, \ldots, \overrightarrow{a}_k) \in \Delta^k(\overrightarrow{a})\} = \frac{n!}{n_1! \cdots n_k!}$$

Démonstration. On se rappelle que si $(\overrightarrow{a}_1, \ldots, \overrightarrow{a}_k) \in \Delta^k(\overrightarrow{a})$, alors chaque $\overrightarrow{a}_i$ est une sous-suite de $\overrightarrow{a}$. En particulier, pour $\overrightarrow{a} = (a_1, \ldots, a_n)$ et $(\overrightarrow{a}_1, \ldots, \overrightarrow{a}_k)$ dans $\Delta(\overrightarrow{a})$ fixés, et sans répétitions d'éléments, il existe une unique fonction

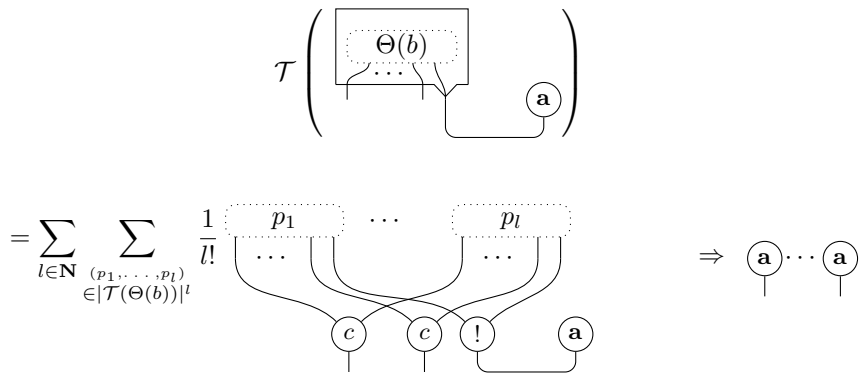


FIGURE 2.39 – Lemme 36 : simulation de la réduction promotion/affaiblissement

$f : \{1,\dots,n\} \to \{1,\dots,k\}$ telle que pour tout $i \in \{1,\dots,n\}$, a_i apparaisse dans $\overrightarrow{a}_{f(i)}$. On en déduit que $\mathbf{card}\{\overrightarrow{a} \in X^n \mid (\overrightarrow{a}_1,\dots,\overrightarrow{a}_k) \in \Delta^k(\overrightarrow{a})\}$ est en bijection avec l'ensemble des fonctions de $\{1,\dots,n\}$ dans $\{1,\dots,k\}$. Le nombre de ces fonctions étant $\frac{n!}{n_1!\dots n_k!}$, on conclut. □

Lemme 38. Soient P, Q des réseaux de **MELL** tels que $P \to_{\langle a_0|c\rangle} Q$. $T(P) \Rightarrow_{(r)} \mathcal{T}(Q)$.

Démonstration. La démonstration de ce lemme est illustrée (de façon simplifiée) en figure 2.40. Posons $P = ((\langle a_0^b | c(T_1,T_2)\rangle, \overrightarrow{C}; \overrightarrow{U}), \Theta)$ et

$$Q = ((\langle a_0^{b^1}|T_1\rangle, \langle a_0^{b^2}|T_2\rangle, \overrightarrow{C}; \overrightarrow{U})[c(a_1^{b^1}, a_1^{b^2})/a_1^b, \dots, c(a_n^{b^1}, a_n^{b^2})/a_n^b], \Theta')$$

pour $\mathbf{ar}(b) = \mathbf{ar}(b^1) = \mathbf{ar}(b^2) = n$, pour $\Theta(b) = ((\overrightarrow{D}; S_1, \dots, S_n, S), \Theta')$, et $\Theta'(b^i) = (\overrightarrow{D}_i; S_1^i, \dots, S_n^i, S^i)$ pour $i \in \{1,2\}$.

Soit également $B_\Theta = \{b_1,\dots,b_k,b\}$. Notons que les b_i sont des noms de boîtes distinctes, alors que b^1 et b^2 correspondent à deux copies de la boîte b issues de la réduction. On pose pour tout $i \in \{1,\dots,k\}$:

$$\mathcal{T}(\Theta(b_i)) = \sum_{j\in J_i} \lambda_{i,j} \cdot p_{i,j} = (\overrightarrow{d}_{i,j}; t_{j,1}, \dots, t_{j,n_i}, t_j)$$

et la substitution $\sigma_{i,j} = [t_{j,1}/a_1^{b_i}, \dots, t_{j,n_i}/a_{n_i}^{b_i}, t_j/a_0^{b_i}]$

Le développement de Taylor de P s'écrit comme suit :

$$\mathcal{T}(P) = \sum_{\substack{(j_1,\dots,j_k)\\ \in J_1\times\dots\times J_k}} \prod_{i=0}^{k} \lambda_{i,j_i} \sum_{l\in\mathbf{N}} \frac{1}{l!} \sum_{\substack{(\overrightarrow{c};t_1,\dots,t_n,t)_l\\ \in|\mathcal{T}(\Theta(b))|^l}}$$
$$\left(\overrightarrow{c}^1,\dots,\overrightarrow{c}^l, \overrightarrow{d}_{1,j_1},\dots,\overrightarrow{d}_{k,j_k}, \langle !(t^1,\dots,t^l)|c(T_1,T_2)\rangle, \overrightarrow{C}; \overrightarrow{U}\right)$$
$$\left[c(t_1^1,\dots,t_1^l)/a_1^b,\dots,c(t_n^1,\dots,t_n^l)/a_n^b\right] \sigma_{1,j_1},\dots,\sigma_{k,j_k}$$

Et il se réduit en la combinaison suivante, par réduction $\rightarrow_{\langle !|c\rangle}$ (ou $\rightarrow_{\langle c|\overline{\mathbf{a}}\rangle}$ quand $l = 0$) sur chaque argument de la somme, donc en une étape de réduction $\Rightarrow$:

$$\sum_{\substack{(j_1,\ldots,j_k)\\ \in J_1\times\ldots\times J_k}} \prod_{i=0}^{k} \lambda_{i,j_i} \sum_{l\in\mathbf{N}} \frac{1}{l!} \sum_{\substack{(\overrightarrow{c};t_1,\ldots,t_n,t)_l\\ \in|\mathcal{T}(\Theta(b))|^l}} \sum_{\substack{(\overrightarrow{t}_1,\overrightarrow{t}_2)\\ \in\Delta^2((t^1,\ldots,t^l))}}$$
$$\left(\overrightarrow{c}^1,\ldots,\overrightarrow{c}^l,\overrightarrow{d}_{1,j_1},\ldots,\overrightarrow{d}_{k,j_k},\langle !(\overrightarrow{t}_1)|T_1\rangle,\langle !(\overrightarrow{t}_2)|T_2\rangle,\overrightarrow{C};\overrightarrow{U}\right)$$
$$\left[c(t_1^1,\ldots,t_1^l)/a_1^b,\ldots,c(t_n^1,\ldots,t_n^l)/a_n^b\right]\sigma_{1,j_1},\ldots,\sigma_{k,j_k}$$

Avant de continuer, remarquons que pour un l-uplet quelconque $\overrightarrow{a}$, la notation $\sum_{(\overrightarrow{a}_1,\overrightarrow{a}_2)\in\Delta^2(\overrightarrow{a})}(\overrightarrow{a}_1,\overrightarrow{a}_2)$ est équivalente à la notation :

$$\sum_{l_1+l_2=l} \sum_{\substack{((a^{1,1},\ldots,a^{1,l_1}),\\ (a^{2,1},\ldots,a^{2,l_2}))\\ \in\Delta^2(\overrightarrow{a})}} ((a^{1,1},\ldots,a^{1,l_1}),(a^{2,1},\ldots,a^{2,l_2}))$$

Nous allons maintenant pouvoir procéder à des expansions Rétoré, de façon à répartir les arbres du développement de la boîte b le long des contractions. On utilise les règles pour l'associativité et la commutativité de la contraction pour obtenir la combinaison suivante :

$$\sum_{\substack{(j_1,\ldots,j_k)\\ \in J_1\times\ldots\times J_k}} \prod_{i=0}^{k} \lambda_{i,j_i} \sum_{l\in\mathbf{N}} \frac{1}{l!} \sum_{\substack{(\overrightarrow{c};t_1,\ldots,t_n,t)_l\\ \in|\mathcal{T}(\Theta(b))|^l}} \sum_{l_1+l_2=l} \sum_{\substack{((\overrightarrow{c}^1;t_1^1,\ldots,t_n^1,t^1)_{l_1},\\ (\overrightarrow{c}^2;t_1^2,\ldots,t_n^2,t^2)_{l_2})\\ \in\Delta^2((\overrightarrow{c};t_1,\ldots,t_n,t)_l)}}$$
$$\left(\overrightarrow{c}^{1,1},\ldots,\overrightarrow{c}^{1,l_1},\overrightarrow{c}^{2,1},\ldots,\overrightarrow{c}^{2,l_2},\overrightarrow{d}_{1,j_1},\ldots,\overrightarrow{d}_{k,j_k},\right.$$
$$\left.\langle !(t^{1,1},\ldots,t^{1,l_1})|T_1\rangle,\langle !(t^{2,1},\ldots,t^{2,l_2})|T_2\rangle,\overrightarrow{C};\overrightarrow{U}\right)$$
$$\left(\left[c(c(t_i^{1,1},\ldots,t_i^{1,l_1})c(t_i^{2,1},\ldots,t_i^{2,l_2}))/a_i^b\right]\right)_{1\leq i\leq n}\sigma_{1,j_1},\ldots,\sigma_{k,j_k}$$

Notons que dans le cas des ! 0-aires, l'expansion est toujours valide, mais en utilisant la neutralité de l'affaiblissement. Dans les cas où $l = 0$, les arbres de la forme $c(t_i^1,\ldots,t_i^l)$ sont en fait des affaiblissements $\mathbf{a}_i$: la réduction est alors de $\mathbf{a}_i$ vers $c(\mathbf{a}_i^1,\mathbf{a}_i^2)$, ce qui correspond à $c(c(t_i^{1,1},\ldots,t_i^{1,l_1})c(t_i^{2,1},\ldots,t_i^{2,l_2}))$ quand $l_1 = l_2 = 0$.

Nous pouvons alors appliquer le lemme 37, car les suites $(\overrightarrow{c};t_1,\ldots,t_n,t)_l \in |\mathcal{T}(\Theta(b))|^l$ sont sans répétition. En effet, il ne peut – par définition – y avoir deux arbres syntaxiquement identiques dans une même structure. Nous cherchons à déterminer pour toute suite $((\overrightarrow{c}^1;t_1^1,\ldots,t_n^1)_{l_1},(c^2;t_1^2,\ldots,t_n^2)_{l_2}) \in |\Theta(b)|^{l_1}\times|\Theta(b)|^{l_2}$, quel est son coefficient dans la combinaison ci-dessus. On pourra ainsi la comparer avec les coefficients de $\mathcal{T}(Q)$. Par le lemme 37, donc, on sait que le nombre de suites $(\overrightarrow{c};t_1,\ldots,t_n,t)_l \in |\Theta(b)|^l$ telles que

$$((\overrightarrow{c}^1;t_1^1,\ldots,t_n^1)_{l_1},(c^2;t_1^2,\ldots,t_n^2)_{l_2}) \in \Delta^2((\overrightarrow{c};t_1,\ldots,t_n,t)_l)$$

est de $\frac{l!}{l_1!l_2!}$. Dans la mesure où tout élément de la série donnée plus haut est pondéré, pour tout $l \in \mathbf{N}$, d'un coefficient $\frac{1}{l!}$, on peut simplifier, et établir qu'elle est égale à la combinaison suivante :

$$\sum_{\substack{(j_1,\ldots,j_k)\\ \in J_1\times\ldots\times J_k}} \prod_{i=0}^{k} \lambda_{i,j_i} \sum_{l_2,l_2\in\mathbf{N}} \frac{1}{l_1!l_2!} \sum_{\substack{(\overrightarrow{c}^1;t_1^1,\ldots,t_n^1,t^1)_{l_1}\\ \in|\mathcal{T}(\Theta(b))|^{l_1}}} \sum_{\substack{(\overrightarrow{c}^2;t_1^2,\ldots,t_n^2,t^2)_{l_2}\\ \in|\mathcal{T}(\Theta(b))|^{l_2}}}$$
$$\left(\overrightarrow{c}^{1,1},\ldots,\overrightarrow{c}^{1,l_1},\overrightarrow{c}^{2,1},\ldots,\overrightarrow{c}^{2,l_2},\overrightarrow{d}_{1,j_1},\ldots,\overrightarrow{d}_{k,j_k},\right.$$
$$\langle !(t^{1,1},\ldots,t^{1,l_1})|T_1\rangle,\langle !(t^{2,1},\ldots,t^{2,l_2})|T_2\rangle,\overrightarrow{C};\overrightarrow{U}\Big)$$
$$\left(\left[c(c(t_i^{1,1},\ldots,t_i^{1,l_1})c(t_i^{2,1},\ldots,t_i^{2,l_2}))/a_i^b\right]\right)_{1\le i\le n}\sigma_{1,j_1},\ldots,\sigma_{k,j_k}$$

Or, le développement de Taylor de Q, quant à lui, est par définition égal à la série ci-dessous :

$$\sum_{\substack{(j_1,\ldots,j_k)\\ \in J_1\times\ldots\times J_k}} \prod_{i=0}^{k} \lambda_{i,j_i} \sum_{l_2,l_2\in\mathbf{N}} \frac{1}{l_1!l_2!} \sum_{\substack{(\overrightarrow{c}^1;t_1^1,\ldots,t_n^1,t^1)_{l_1}\\ \in|\mathcal{T}(\Theta'(b^1))|^{l_1}}} \sum_{\substack{(\overrightarrow{c}^2;t_1^2,\ldots,t_n^2,t^2)_{l_2}\\ \in|\mathcal{T}(\Theta'(b^2))|^{l_2}}}$$
$$\left(\overrightarrow{c}^{1,1},\ldots,\overrightarrow{c}^{1,l_1},\overrightarrow{c}^{2,1},\ldots,\overrightarrow{c}^{2,l_2},\overrightarrow{d}_{1,j_1},\ldots,\overrightarrow{d}_{k,j_k},\right.$$
$$\langle !(t^{1,1},\ldots,t^{1,l_1})|T_1\rangle,\langle !(t^{2,1},\ldots,t^{2,l_2})|T_2\rangle,\overrightarrow{C};\overrightarrow{U}\Big)$$
$$\left(\left[c(c(t_i^{1,1},\ldots,t_i^{1,l_1})c(t_i^{2,1},\ldots,t_i^{2,l_2}))/a_i^b\right]\right)_{1\le i\le n}\sigma_{1,j_1},\ldots,\sigma_{k,j_k}$$

Où l'on se souvient que Θ' associe à b_1 et à b_2 deux structures identiques (à renommage des atomes près), également identiques à $\Theta(b)$. On peut donc enfin conclure que $\mathcal{T}(P) \Rightarrow_{(r)} \mathcal{T}(Q)$ (voir la figure 2.40 pour une illustration simplifiée de la preuve).

□

Lemme 39. Soient P, Q des réseaux de **MELL** tels que $P \to_{\langle a_0|a\rangle} Q$. $T(P) \Rightarrow_{(r)} \mathcal{T}(Q)$.

Démonstration. On ne détaille pas ici la preuve, car elle résulte de raisonnements et d'équations similaires à celle du lemme précédent, mais avec une coupure contre une contraction d'arité arbitraire. On utilise donc le lemme 37 dans toute sa généralité et pas seulement dans le cas binaire comme précédemment. La simulation est toutefois illustrée en figure 2.41 dans sa version simplifiée.

Nous illustrons également en figure 2.42 la simulation de la réduction quand une des boîtes est développée avec un nombre zéro de copies. Ce cas est la motivation de l'introduction de la rédcuion $\rightrightarrows^{\mathsf{neut}}_{\langle \overline{\mathbf{a}}|c\rangle}$.

□

Théorème 8. Soient P, Q deux réseaux de **MELL**. Si $P \to Q$, $\mathcal{T}(P) \Rightarrow_{(r)} \mathcal{T}(Q)$.

Démonstration. La preuve est par récurrence sur la profondeur de P.

Si P est de profondeur 0, alors P ne contient pas de boîte, et $\mathcal{T}(P) = P$, il en va de même pour Q : $\mathcal{T}(Q) = Q$, ce qui conclut ce cas.

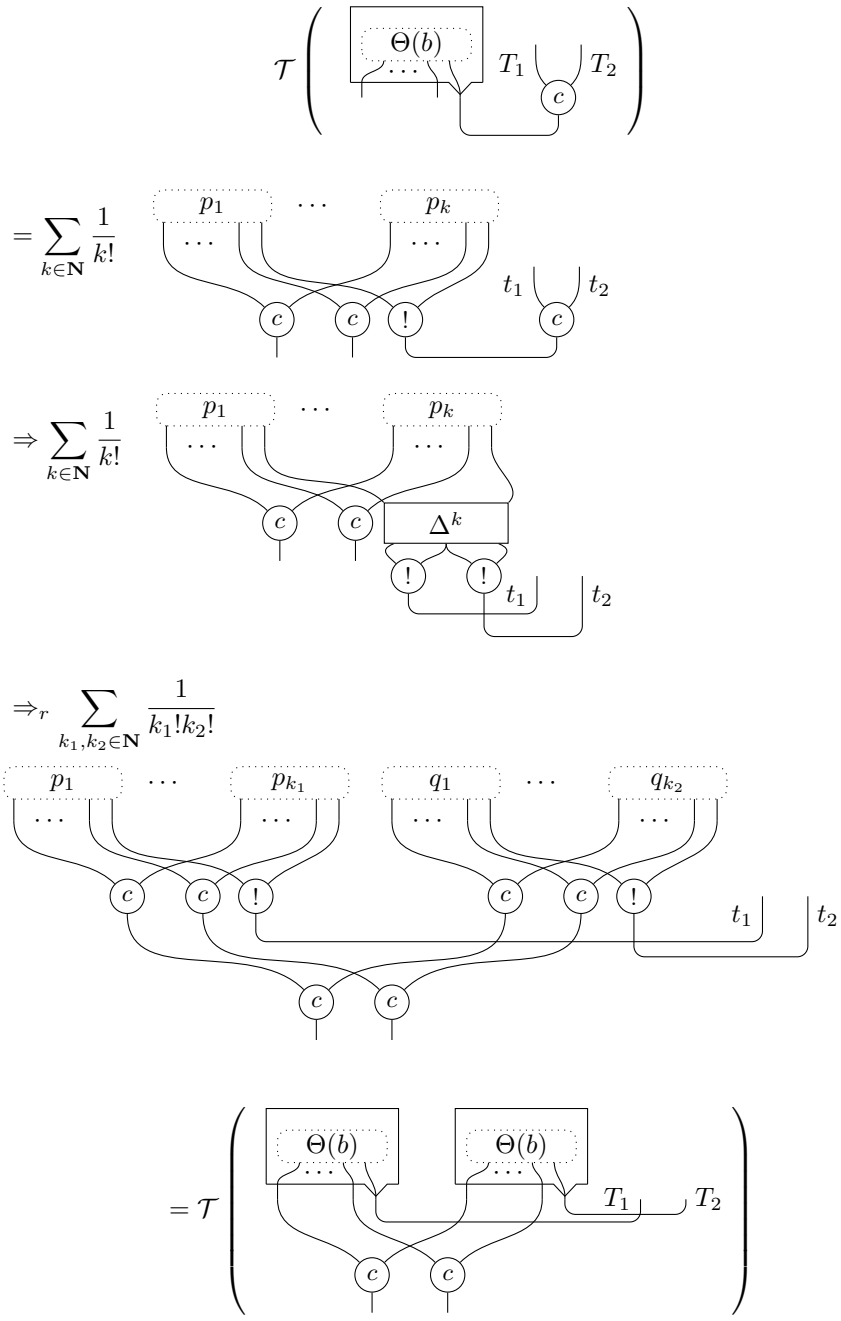


FIGURE 2.40 – Simulation de la réduction promotion/contraction

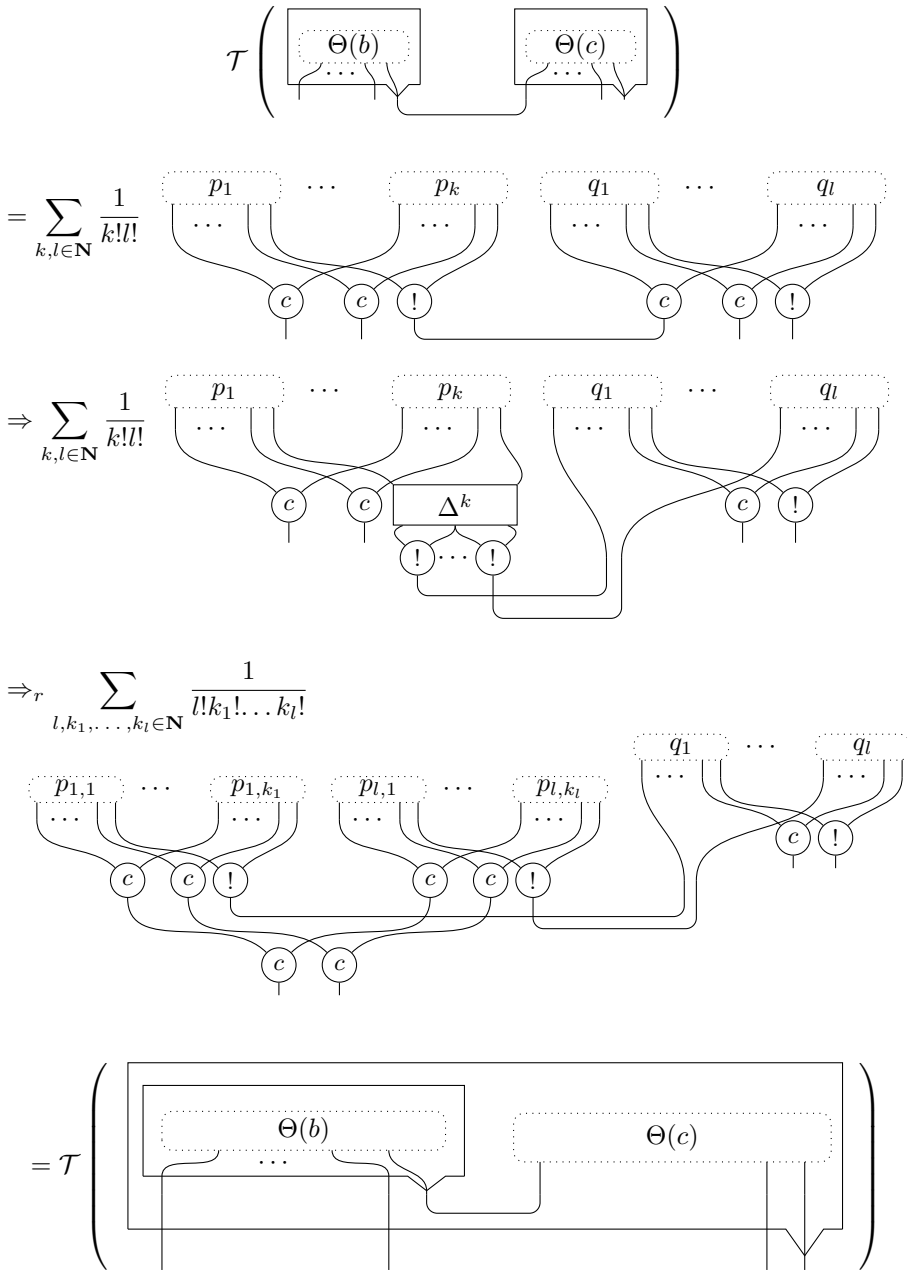


FIGURE 2.41 – Simulation de la réduction promotion/promotion

Si P est de profondeur $\pi+1$, alors il faut distinguer plusieurs cas en fonction du type de la réduction de P à Q.

Si cette dernière est multiplicative ou axiomatique, il suffit de remarquer que, dans le cas d'une coupure multiplicative, $\mathcal{T}(P) = \sum i \in I\lambda_i \cdot (\langle \otimes(t_i, t'_i) |\ \parr (s_i, s'_i)\rangle, \overrightarrow{c}_i; \overrightarrow{t}_i)$ et $\mathcal{T}(Q) = \sum_{i\in I} \lambda_i \cdot (\langle t_i|s_i\rangle, \langle t'_i|s'_i\rangle, \overrightarrow{c}_i; \overrightarrow{t}_i)$. Et dans le cas d'une coupure axiomatique, $\mathcal{T}(P) = \sum_{i\in I} \lambda_i \cdot (\langle t_i|x_i\rangle, \overrightarrow{c}_i; \overrightarrow{t}_i)$ et $\mathcal{T}(Q) = \sum_{i\in I} \lambda_i \cdot (\overrightarrow{c}_i; \overrightarrow{t}_i)[t_i/\overline{x}_i]$. Dans les deux cas, la réduction $\Rightarrow$ s'applique par définition.

Si P se réduit en Q par réduction exponentielle, alors l'un des lemmes 35, 36, 38, et 39 s'applique.

La dernière possibilité est que (P, Θ) se réduise en (Q, Θ') par réduction interne. C'est-à-dire qu'il existe une boîte $b \in B_\Theta$ et une structure $R = \Theta(b)$ telle que $R \to R'$, avec $\Theta' = \Theta[\{b \mapsto R\}/\{b' \mapsto R'\}]$ et $Q = P([a_i^{b'}/a_i^b])_{0\leq i\leq \mathbf{ar}(b)}$. Dans ce cas, nous pouvons utiliser l'hypothèse de récurrence, car R est à profondeur π : $\mathcal{T}(R) \Rightarrow_{(r)} \mathcal{T}(R')$

Soit r_i égal à $(\overrightarrow{c}_i; t_{i,1}, \ldots, t_{i,\mathbf{ar}(b)}, t_{i,0})$ pour tout $i \in \{1, \ldots, n\}$. On écrit $\mathcal{T}(R) =$

$$\sum_{n\in\mathbf{N}} \frac{1}{n!} \sum_{\substack{(r_1,\ldots,r_n)\\ \in|\mathcal{T}(R)|^n}} \prod_{i=1}^{n} \mu_i$$
$$(\overrightarrow{c}_1, \ldots, \overrightarrow{c}_n; c(t_{1,1}, \ldots, t_{n,1}), \ldots, c(t_{1,\mathbf{ar}(b)}, \ldots, t_{n,\mathbf{ar}(b)}), !(t_{1,0}, \ldots, t_{n,0}))$$

Les μ_i sont les coefficients obtenus inductivement par le développement de Taylor de R. De la même façon, on écrit $\mathcal{T}(R') =$

$$\sum_{n\in\mathbf{N}} \frac{1}{n!} \sum_{\substack{(r'_1,\ldots,r'_n)\\ \in|\mathcal{T}(R')|^n}} \prod_{i=1}^{n} \mu'_i$$
$$(\overrightarrow{c}'_1, \ldots, \overrightarrow{c}'_n; c(t'_{1,1}, \ldots, t'_{n,1}), \ldots, c(t'_{1,\mathbf{ar}(b')}, \ldots, t'_{n,\mathbf{ar}(b')}), !(t'_{1,0}, \ldots, t'_{n,0}))$$

Soulignons que $\mathbf{ar}(b) = \mathbf{ar}(b')$, car le nombre de conclusions de R' est identique au nombre de conclusions de R. Ce nombre est invariant par réduction.

En isolant la boîte b contenant R, on peut écrire $\mathcal{T}(P) =$

$$\sum_{j\in J} \lambda_J \cdot \sum_{n\in\mathbf{N}} \frac{1}{n!} \sum_{\substack{(r_1,\ldots,r_n)\\ \in|\mathcal{T}(R)|^n}} \prod_{i=1}^{n} \mu_i$$
$$p_i\left([c(t_{1,l}, \ldots, t_{n,l})/x_l]\right)_{1\leq l\leq \mathbf{ar}(b)} [!(t_{1,0}, \ldots, t_{n,0})/x_0]$$

et $\mathcal{T}(Q) =$

$$\sum_{j\in J} \lambda_j \cdot \sum_{n\in\mathbf{N}} \frac{1}{n!} \sum_{\substack{(r'_1,\ldots,r'_n)\\ \in|\mathcal{T}(R'))|^n}} \prod_{i=1}^{n} \mu'_i$$
$$p_i\left([c(t_{1,l}, \ldots, t_{n,l})/x_l]\right)_{1\leq l\leq \mathbf{ar}(b')} [!(t_{1,0}, \ldots, t_{n,0})/x_0]$$

Or, notre hypothèse de récurrence nous assure de la propriété suivante :

$$\sum_{n\in\mathbf{N}} \frac{1}{n!} \sum_{\substack{(r_1,\dots,r_n)\\ \in|\mathcal{T}(R)|^n}} \prod_{i=1}^{n} \mu_i$$
$$(\overrightarrow{c}_1,\dots,\overrightarrow{c}_n; c(t_{1,1},\dots,t_{n,1}),\dots,c(t_{1,\mathbf{ar}(b)},\dots,t_{n,\mathbf{ar}(b)}),!(t_{1,0},\dots,t_{n,0}))$$

$$\Rightarrow_{(r)} \sum_{n\in\mathbf{N}} \frac{1}{n!} \sum_{\substack{(r'_1,\dots,r'_n)\\ \in|\mathcal{T}(R')|^n}} \prod_{i=1}^{n} \mu'_i$$
$$(\overrightarrow{c}'_1,\dots,\overrightarrow{c}'_n; c(t'_{1,1},\dots,t'_{n,1}),\dots,c(t'_{1,\mathbf{ar}(b')},\dots,t'_{n,\mathbf{ar}(b')}),!(t'_{1,0},\dots,t'_{n,0}))$$

On peut donc conclure par le lemme 34, en intégrant cette réduction en contexte, c'est-à-dire que ce lemme nous assure que la réduction de $\mathcal{T}(R)$ à $\mathcal{T}(R')$ peut être considérée à l'intérieur d'une combinaison plus large, à savoir $\mathcal{T}(P)$.

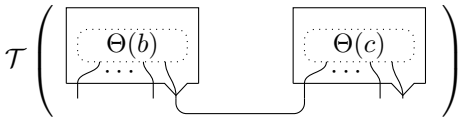
□

2.7 Conclusions et perspectives

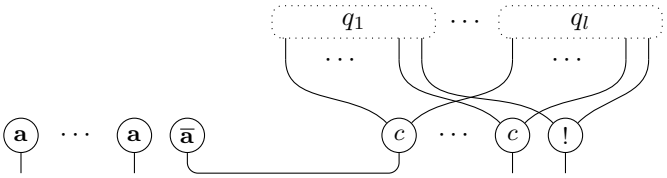
Nous avons montré que l'élimination des coupures pouvait être simulée dans le développement de Taylor, à l'aide notamment des expansions Rétoré. Ce résultat peut être étendu à plusieurs étapes d'élimination des coupures, car les propriétés nécessaires à la simulation sont préservées par la réduction. Pour autant, il faut être plus précis si nous souhaitons simuler la normalisation en général. En effet, pour un réseau P de **MELL** ayant une forme normale $\mathsf{fn}(P)$, nous ne pouvons pas donner simplement la formule : $\mathcal{T}(\mathsf{fn}(P)) = \mathsf{fn}(\mathcal{T}(P))$. En effet, les expansions Rétoré ne sont pas normalisantes, donc la forme normale du développement de Taylor ne peut être définie que pour la réduction à ressources. On peut définir alors $\mathsf{fn}(\sum_{i\in I} \lambda_i \cdot p_i) = \sum_{i\in I} \lambda_i \cdot \mathsf{fn}(p_i)$, où $\mathsf{fn}(p_i)$ correspond à la normalisation de la réduction de $\mathbf{DLL}_0$. Dans ce cas, le résultat de simulation est un peu plus faible, car ce que l'on pourra montrer est que $\mathsf{fn}(\mathcal{T}(P)) \Rightarrow_r^k \mathcal{T}(\mathsf{fn}(P))$, pour un certain $k \in \mathbf{N}$. Une fois normalisés les termes à ressources, on peut compléter la simulation par un nombre fini d'expansions Rétoré, car ces expansions sont compatibles avec la normalisation de $\mathbf{DLL}_0$.

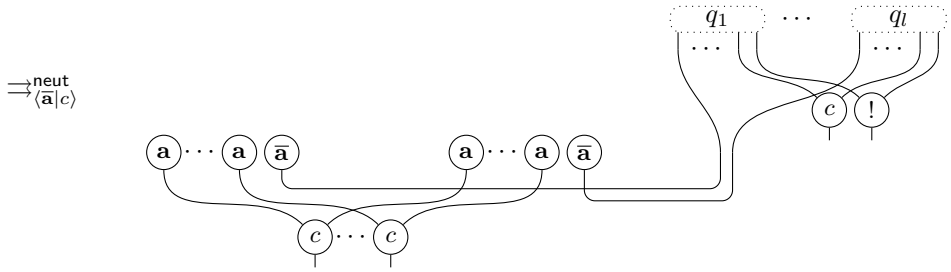
Une des extensions envisageable de nos travaux est d'étendre ce résultat de simulation à la Logique Linéaire différentielle, c'est à dire à **MELL** enrichie avec la cocontraction, le coaffaiblissement et la codéréliction. Cela demanderait de faire un autre choix de présentation du développement de Taylor. En particulier, pour analyser les réductions à un niveau de précision similaire à celui que nous avons choisi, il faudrait séparer la cocontraction et la codéréliction, et ne plus utiliser de connecteur généralisé !. Cela n'apporte pas de difficulté conceptuelle supplémentaire, mais demande une gestion des réductions beaucoup plus lourde dans la présentation (d'où notre choix, suffisant pour **MELL**).

Une autre direction qui semble intéressante serait d'examiner l'interaction entre nos résultats et les réseaux de preuves infinis, présentés récemment par De et Saurin [DS19], dans le cadre de la conception d'outils pour les preuves circulaires.



contient le réseau suivant pour tout $l \in \mathbf{N}$:





qui apparaît bien dans le développement suivant, où la boîte la plus profonde a un nombre nul de copies :

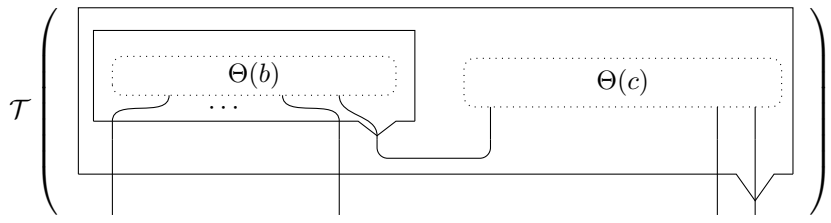


FIGURE 2.42 – Illustration de la commutation de la réduction boîte/boîte dans le cas d'un développements 0-aire.

Chapitre 3

Structures calculatoires

J'aimerais connaître votre réaction devant le contenu mathématique de ma dernière lettre, à supposer qu'elle soit parvenue entre vos mains, ce dont je doute.

Georg Cantor [a]

a. Lettre à Richard Dedekind du 15 avril 1882. Dans *Jean Cavaillès, Philosophie mathématique*, Hermann, 1962.

3.1 Introduction

3.1.1 Stratégies et théories d'évaluation

Le λ-calcul, avec ou sans types, apparaît comme une *théorie*, dans laquelle est définie une syntaxe qui caractérise un ensemble de termes, ceux que l'on a *le droit* d'écrire. Par dessus cette définition, la théorie est munie de règles de réduction, qui déterminent pour un terme M bien formé l'ensemble des termes N tels que $M \to N$, c'est-à-dire les règles de réduction que l'on a le droit d'appliquer, en fonction de la structure de M. La variante du λ-calcul la moins contrainte à cet égard est celle dans laquelle n'importe quel redex apparaissant comme sous-terme de M peut être soumis à la réduction correspondante.

Une façon de raffiner le calcul, et rendre plus concrète sa sémantique opérationnelle, est d'établir une *stratégie* d'évaluation. Une stratégie consiste à imposer des contraintes sur les réductions que l'on s'autorise à effectuer. Ces contraintes ne concernent pas la forme des redex, qui est fixée par la théorie, mais concerne leur place dans le terme considéré.

La stratégie peut être présentée de façon algorithmique, en explicitant l'endroit où l'on cherche à effectuer une réduction. On peut par exemple donner l'instruction suivante : réduire le redex le plus à droite dans les applications. Cette instruction interdit les réductions de la forme $MN \to M'N$, où $M \to M'$, dès que N est sujet à réduction. Elle rend la procédure déterministe, bien que les contextes d'évaluation permettent en général l'application de règles à différents endroits d'un même terme.

Une autre façon d'obtenir un calcul plus fin est de modifier directement la théorie, sans passer nécessairement par une stratégie particulière. Là, par exemple, où l'on parle parfois de stratégie *par valeur* pour désigner la procédure consistant à ne pas réduire l'argument d'une application, si celui-là est lui-même réductible, on peut également définir une théorie qui enrichit le λ-calcul ordinaire et correspond aux comportements d'*Appel-Par-Valeur* dans ses définitions mêmes.

Une telle théorie demande de redéfinir la syntaxe des termes et les règles de réduction. On peut alors donner la grammaire des termes en deux catégories distinctes, les termes généraux $M, N ::= V \mid MN$ et les *valeurs* $V ::= x \mid \lambda xM$. On donne ensuite la règle de réduction suivante $(\lambda xM)V \to M[V/x]$, qui jouit bien de la propriété de la réduction droite : Si dans le terme MN, M et N sont tous deux des redex, alors on est forcé de réduire N avant de pouvoir le faire passer comme argument d'un redex plus extérieur.

Les théories de l'Appel-Par-Nom et de l'Appel-Par-Valeur correspondent à des paradigmes concrets de la programmation, et présentent des sémantiques opérationnelles et dénotationnelles différentes. La première est proche du λ-calcul ordinaire, où il n'y a pas de contrainte sur la forme des redex. La seconde, en détachant une catégorie syntaxique spécifique dans la grammaire des termes, propose une conception du calcul qui se résume dans le paradigme suivant : « Une valeur *est*, un calcul *fait* » (Levy [Lev06]). Dans le cas d'un système pur, les valeurs sont des variables, qui représentent des objets de base du langage, et des abstractions, qui représentent des fonctions. Les calculs sont les applications, qui contiennent en particulier tous les redex.

On peut trouver une description assez exhaustive des sémantiques particulières de l'Appel-Par-Valeur dans les travaux de Pravato, Ronchi della Rocha et Roversi [PdRR99]. Nous allons nous concentrer dans ce chapitre sur un langage qui permet d'exprimer et d'encoder à la fois l'Appel-Par-Nom et l'Appel-Par-Valeur typés.

3.1.2 Une théorie générale : l'Appel-Par-Pousse-Valeur

Levy publie en 2006 *Call-By-Push-Value : Decomposing Call-By-Value and Call-By-Name* [Lev06]. Dans cette nouvelle théorie, que par chauvinisme nous continuerons d'appeler Appel-Par-Pousse-Valeur, sont construites des traductions de l'Appel-Par-Nom et de l'Appel-Par-Valeur, qui préservent notamment l'interprétation dénotationnelle. Du côté opérationnel, des propriétés de simulation et d'abstraction pleine y sont également établies.

Ce nouveau paradigme offre alors des possibilités particulièrement prometteuses, qui expliquent probablement une partie de son succès : ce que l'on peut y montrer, ce que l'on peut y construire comme preuves et outils, peut être exporté aux théories qui s'y plongent.

C'est naturellement ce qui motive notre présente contribution. Le développement de Taylor est un outil puissant pour l'étude des sémantiques de différents langages, et en donner une construction pour l'Appel-Par-Pousse-Valeur entretient la possibilité d'obtenir des résultats satisfaisants dans un cadre général et modulaire.

Si le calcul est typé, il est étendu et enrichi de façon à pouvoir exprimer des données structurées, des propriétés arithmétiques, de la récursion et de la corécursion. Nous avons affaire à ce qui se rapproche d'un langage de programmation concret, et qui permet dans le même temps d'intégrer les sémantiques construites autour des langages fonctionnels plus théoriques que sont les λ-calculs par nom et par valeur.

Une des propriétés les plus remarquables de la Logique Linéaire est la possibilité qu'elle offre de traiter la duplication de façon particulièrement fine. Cela se traduit également dans les différentes traduction du λ-calcul vers les formules de la Logique Linéaire, à travers les réseaux par exemple. En effet, l'exponentielle ! qui autorise son argument à être dupliqué permet également de contrôler les réductions possibles. Par ce contrôle, on peut également traduire les théories d'Appel-Par-Nom et d'Appel-Par-Valeur, avec des propriétés de simulation adéquates. En cela, la Logique Linéaire et l'Appel-Par-Pousse-Valeur sont intrinsèquement liés, et l'étude de leurs sémantiques sont fortement complémentaires.

Nous allons approfondir ce lien en proposant et étudiant le développement de Taylor de l'Appel-Par-Pousse-Valeur à l'aide des outils qu'offrent les sémantiques de la Logique Linéaire.

3.1.3 Développement de Taylor et stratégies

Les études croisées du développement de Taylor et des différentes théories d'évaluation trouvent leurs racines dans les premiers travaux de Girard. En effet, la naissance de la Logique Linéaire qui provient de l'examen des sémantiques quantitatives du λ-calcul s'accompagne des fameuses *traductions de Girard*. Ces traductions se font depuis les types simples du λ-calcul pur vers les formules de la Logique Linéaire. La première traduit la flèche intuitionniste $A \to B$ en la formule $!A \multimap B$. La seconde traduit $A \to B$ en $!(A \multimap B)$ (ou $!A \multimap !B$, ce qui donne le même comportement général).

Il a alors été remarqué que la première traduction générait un calcul par nom, et la seconde un calcul par valeur. En particulier, il existe une traduction par nom **tn** et une traduction par valeur **tv** du λ-calcul dans les réseaux de preuve, nous en donnons illustration aux figures 3.1 et 3.2.

En examinant les propriétés et les différences de ces constructions, on est conduit à constater que la duplicabilité des ressources qu'exprime l'exponentielle permet d'encoder différentes sémantiques opérationnelles, à l'instar de l'Appel-Par-Pousse-Valeur. En effet, rendre duplicable la partie « argument » d'une fonction, comme lors de la traduction par nom $!A \multimap B$, ou la fonction elle même, comme dans la traduction par valeur $!(A \multimap B)$, donne lieu à différentes façons de réduire les termes correspondant.

Dans le développement de Taylor, comme dans la sémantique relationnelle, cette gestion de la duplicabilité est concrétisée par la construction de multiensembles. Un élément de l'interprétation relationnelle du type $!A \multimap B$ est de la forme $([a_1, \ldots, a_k], b)$, où les a_i sont dans l'interprétation de A et b est dans l'interprétation de B. Cet objet représente intuitivement un processus de cal-

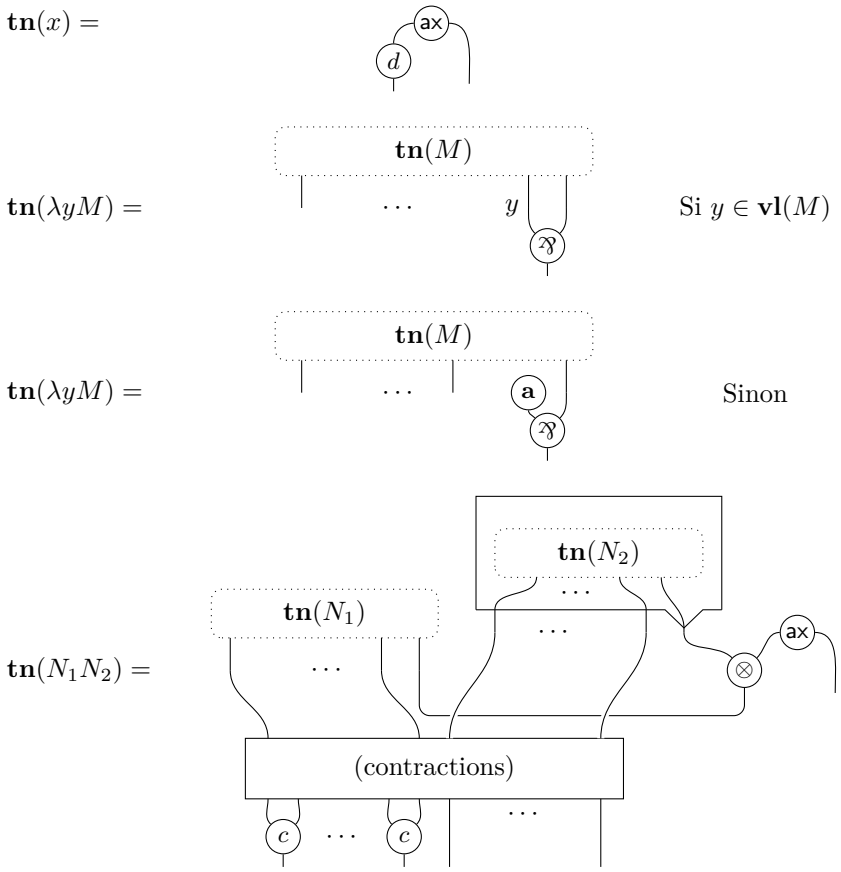


FIGURE 3.1 – Traduction par nom de Λ dans **MELL**

cul qui produit un élément de B à partir de k éléments de A. La notion de consommation de ressources apparaît.

La syntaxe des approximationss, les *termes à ressources*, reflètent ce mécanisme. Un terme à ressources de type $!A \multimap B$ devra être appliqué à un multiensemble de termes à ressources pour que la réduction s'opère. Ainsi, la règle de réduction du calcul à ressources sera de la forme $\langle \lambda x m \rangle [n_1, \ldots, n_k] \to m[n_1/x_1, \ldots, n_k/x_k]$, où les x_i dénotent ici les occurrences libres de x dans m. Nous optons pour la convention d'écrire les termes à ressources en minuscule, et les λ-termes usuels en majuscule. Cette réduction n'est bien sûr définie que si ce nombre, que l'on note $\mathsf{deg}_x(m)$ pour le *degré* de x dans m, est égal à k. De plus, les multiensembles n'étant pas ordonnés [1], alors que les occurrences de x le sont de fait, la réduction doit prendre en compte toutes les possibilités. La

1. Il existe une version ordonnée de ces constructions, où l'on n'a plus de multiensembles, mais des listes. Les sémantiques sous-jacentes ne sont alors pas les mêmes. Voir Tsukada, Asada et Ong [TAO18].

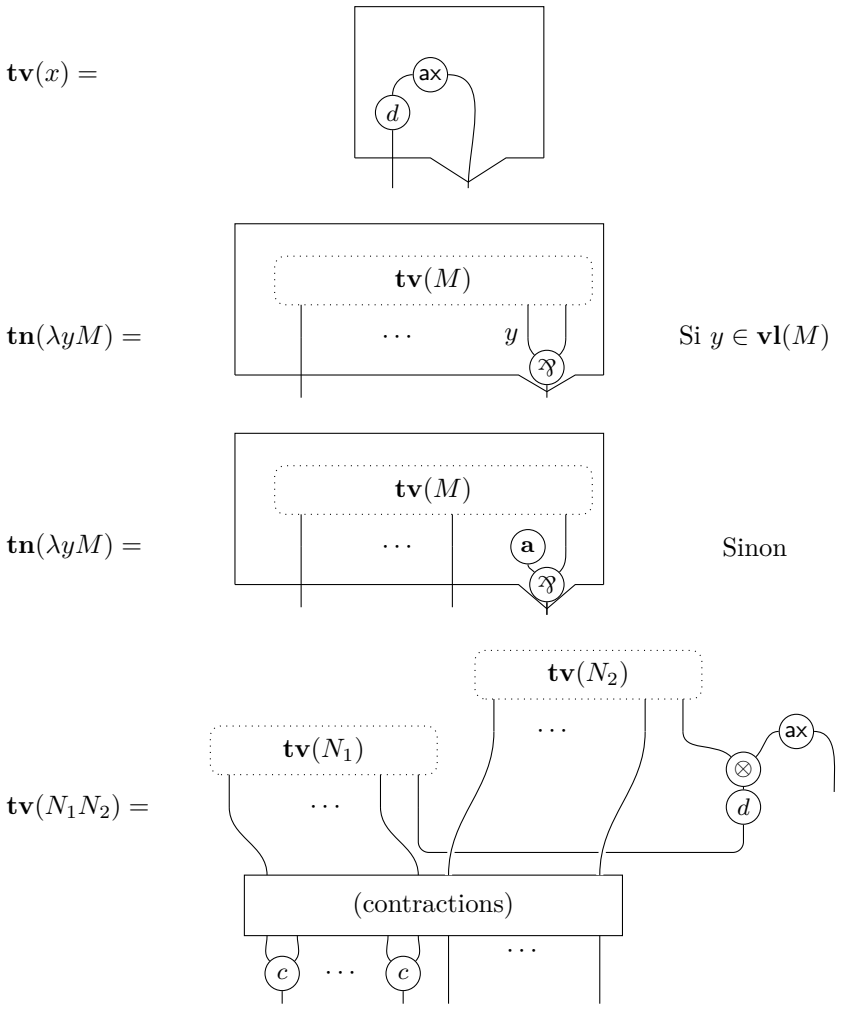


FIGURE 3.2 – Traduction par valeur de Λ dans **MELL**

forme générale de la réduction à ressources est alors la suivante :

$$\langle \lambda x m \rangle [n_1, \ldots, n_k] \to \sum_{f \in \mathfrak{S}_k} m[n_{f(1)}/x_1, \ldots, n_{f(k)}/x_k] \qquad \text{si } \mathsf{deg}_x(m) = k$$
$$\to 0 \qquad \text{sinon}$$

La dynamique associée à ce calcul à ressources apparaît dans la première définition du développement de Taylor syntaxique, due à Ehrhard et Regnier avec l'introduction du λ-calcul différentiel [ER03]. La première expression d'un calcul similaire adapté à l'Appel-Par-Valeur provient d'un travail plus récent de Ehrhard [Ehr12].

Une façon de comprendre cette construction est d'observer que les valeurs de type $A \to B$ dans le λ-calcul pur par valeur sont de la forme $\lambda x M$ [2]. Si l'on

2. Ou des variables, mais ce n'est pas le cas si les variables sont typées par des formules atomiques.

considère la seconde traduction de Girard, qui à $A \to B$ associe $!(A \multimap B)$, alors dans la mesure où l'exponentielle est traduite dans le calcul à ressources par des multiensembles, le type $!(A \multimap B)$ est habité par des valeurs de la forme $[\lambda x m_1, \dots, \lambda x m_k]$. Si alors seules les valeurs sont des multiensembles, on remarque que pour un terme à ressources $\langle \lambda x m \rangle n$, la règle de réduction donnée plus haut ne s'applique que si n est une valeur : en effet, la réduction n'est pas définie si l'argument du redex est une application. C'est une propriété de simulation qui sera formalisée en section 3.4.2.

Le développement de Taylor de l'Appel-Par-Valeur a été aprofondi par Kerinec, Manzonetto et Pagani [KMP18], qui ont notamment donné une caractérisation des formes normales et exhibé des arbres de Böhm pour l'Appel-Par-Valeur, et montré que ces derniers étaient compatibles avec la normalisation du développement de Taylor ensembliste.

Entre le λ-calcul usuel par nom, et le λ-calcul par valeur, notre démarche consiste à installer un cadre de travail qui permettrait d'étudier leurs développements de Taylor respectifs dans une seule théorie, et ainsi de donner un langage capable de subsumer les constructions de Ehrhard et Regnier [ER03], de Ehrhard [Ehr12], et de Kerinec, Manzonetto et Pagani [KMP18]. Donner donc un développement de Taylor à l'Appel-Par-Pousse-Valeur s'impose comme prometteur dans cette perspective.

Un premier travail dans ce sens à déjà été proposé par Ehrhard et Guerrieri, qui introduisent le *Bang Calcul* [EG16], retravaillé ultérieurement par Guerrieri et Manzonetto [GM19]. Le Bang Calcul est une sorte d'Appel-Par-Pousse-Valeur non typé, qui comporte une exponentielle et une déréliction explicites, ce qui permet d'y plonger l'Appel-Par-Nom et l'Appel-Par-Valeur.

Notre démarche est légèrement différente, car nous cherchons à nous rapprocher de l'Appel-Par-Pousse-Valeur vu comme un langage typé. Cela est restrictif dans la mesure où les théories que nous pourront y encoder devront être également typée (le terme (xx) ne pourra pas y être traduit, par exemple), mais cela entraîne également une étude poussée des mécanismes de duplicabilité quand ils sont dirigés par le typage et quand le langage est enrichi par des entiers, des instructions conditionnelles, des points fixes explicites permettant d'exprimer la récursion et la corécursion.

Nous partirons d'une syntaxe existante, fortement inspirée des sémantiques de la Logique Linéaire, développée par Ehrhard, dans un travail justement intitulé *Call-By-Push-Value from a Linear Logic point of view* [Ehr16a], et dont une sémantique a été développée par Ehrhard et Tasson dans une extension probabiliste [ET16].

3.1.4 Contenu du chapitre

Nous présentons dans un premier temps notre syntaxe, reprise de Ehrhard, et que nous nommons Λ_{pv}, pour l'Appel-Par-Pousse-Valeur. Nous donnons un survol des sémantiques associées, pour expliquer la problématique de la duplicabilité des types positifs, qui est liée à la structure de coalgèbre de leur interprétation. Nous développons ensuite un calcul à ressources Δ_{pv} adéquat pour l'approximation des termes Λ_{pv}, et qui rend compte de l'action du morphisme de coalgèbre de la sémantique.

Nous construisons le développement de Taylor de Λ_{pv}, à travers des ensembles de termes de Δ_{pv} d'abord, puis nous montrons que nous pouvons l'enrichir

en considérant des combinaisons linéaires de termes à ressources. Pour cela, nous établissons un résultat analogue à celui du chapitre 2 de ce mémoire, en démontrant que l'on peut définir une réduction sur des combinaisons infinies de termes à ressources, et que cette réduction est apte à simuler la sémantique opérationnelle de Λ_{pv}. Enfin, nous montrons que le développement de Taylor permet d'implémenter cette simulation, avec une propriété de commutation des coefficients impliqués.

3.2 Appel-Par-Pousse-Valeur

3.2.1 Syntaxe et sémantique opérationnelle

On considère une présentation de l'Appel-Par-Pousse-Valeur provenant de Ehrhard [Ehr16a], et adaptée pour son étude à travers les sémantiques de la Logique Linéaire.

Définition 29 (Calcul Λ_{pv} pour l'Appel-Par-Pousse-Valeur)**.**

$$\begin{aligned}\Lambda_{\mathsf{pv}} : M ::= x \mid \lambda x M \mid \langle M\rangle M \mid \mathbf{case}(M, y\cdot M, z\cdot M) \mid \mathbf{fix}_x(M) \mid (M,M) \mid \\ \pi_1(M) \mid \pi_2(M) \mid M^! \mid \mathbf{der}(M) \mid \iota_1(M) \mid \iota_2(M)\end{aligned}$$

On distingue un sous-ensemble de Λ_{pv}, les *valeurs* :

$$V ::= x \mid M^! \mid (V,V) \mid \iota_1(V) \mid \iota_2(V)$$

Types positifs : $A ::= !I \mid A \otimes A \mid A \oplus A$
Types généraux : $I ::= A \mid A \multimap I \mid \top$

$$\frac{}{\Gamma, x : A \vdash x : A} \qquad \frac{\Gamma \vdash M : I}{\Gamma \vdash M^! : !I} \qquad \frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x M : A \multimap B}$$

$$\frac{\Gamma \vdash M : A \multimap I \qquad \Delta \vdash N : A}{\Gamma, \Delta \vdash \langle M\rangle N : I}$$

$$\frac{\Gamma \vdash M : A \qquad \Delta \vdash N : B}{\Gamma, \Delta \vdash (M,N) : A \otimes B} \qquad \frac{\Gamma \vdash M : A_1 \otimes A_2}{\Gamma \vdash \pi_i(M) : A_i}\ i \in \{1,2\}$$

$$\frac{\Gamma \vdash M : A_i}{\Gamma \vdash \iota_i(M) : A_1 \oplus A_2}\ i \in \{1,2\} \qquad \frac{\Gamma \vdash M : !A}{\Gamma \vdash \mathbf{der}(M) : A}$$

$$\frac{\Gamma \vdash M_1 : A \oplus B \qquad \Delta \vdash M_2 : I \qquad \Theta \vdash M_3 : I}{\Gamma, \Delta, \Theta \vdash \mathbf{case}(M_1, y\cdot M_2, z\cdot M_3) : I} \qquad \frac{\Gamma, x : !I \vdash M : I}{\Gamma \vdash \mathbf{fix}_x(M) : I}$$

FIGURE 3.3 – Règles de typage pour Λ_{pv}

Les règles de typage sont données en figure 3.3 et les règles de réduction sont données ci-dessous :

$$\langle \lambda x M\rangle V \to_{\mathsf{pv}} M[V/x] \qquad \mathbf{der}(M^!) \to_{\mathsf{pv}} M$$
$$\pi_i(V_1, V_2) \to_{\mathsf{pv}} V_i \qquad \mathbf{fix}_x(M) \to_{\mathsf{pv}} M[(\mathbf{fix}_x(M))^!/x]$$
$$\mathbf{case}(\iota_i(V), x_1\cdot M_1, x_2\cdot M_2) \to_{\mathsf{pv}} M_i[V/x_i]$$

On définit les contextes d'évaluation E, pour tous termes M, N.

$$E ::= [] \mid \langle M\rangle E \mid \langle E\rangle M \mid \pi_i(E) \mid \iota_i(E) \mid (M,E) \mid (E,M) \mid \mathbf{case}(E, x{\cdot}M, y{\cdot}N) \mid \mathbf{der}(E)$$

et on pose comme règle supplémentaire $E[M] \to_{\mathsf{pv}} E[N]$ pour tous M, N tels que $M \to_{\mathsf{pv}} N$.

Exemples

Ce calcul est assez riche pour définir des types de données inductifs et coinductifs. Les entiers, par exemple, peuvent être définis en ajoutant à notre syntaxe le produit vide $() : \underline{0} = \iota_1(), \underline{k+1} = \iota_2(\underline{k})$, de façon à ce que tous les entiers définis ainsi aient le type $\iota = (1 \oplus \iota)$. Le successeur alors correspond à la seconde injection. Puis, si x n'a pas d'occurrence libre dans N_1, le terme $\mathbf{case}(M, x \cdot N_1, y \cdot N_2)$ est un encodage adéquat d'une conditionnelle de type « test à zéro » : $\mathbf{If}(M, N_1, y \cdot N_2)$ (où la valeur à laquelle M est évaluée est donnée comme argument de N_2 si elle est différente de 0).

Le type coinductif des flux (*streams*, en anglais) peut également être construit : soit A un type positif, $S_A = !(A \otimes S_A)$ est le type des flux paresseux de type A (la queue du flux étant encapsulée dans une exponentielle, l'évaluation est repoussée). Nous pouvons alors construire un terme de type $S_A \multimap \iota \multimap A$ qui calcule le k-ème élément d'un flux :

$$\mathbf{fix}_f\,(\lambda x\lambda y(\mathbf{If}(y, \pi_1(\mathbf{der}(x)), z \cdot \langle \mathbf{der}(f)\rangle \pi_2 \langle \mathbf{der}(x)\rangle z)))$$

et un terme de type $!(\iota \multimap A) \multimap S_A$:

$$\mathbf{fix}_f\,\left(\lambda g\,\left(\mathbf{der}(g)\underline{0}, \langle \mathbf{der}(f)\rangle(\lambda x \langle \mathbf{der}(g)\rangle \mathbf{suc}(x))^!\right)\right)$$

qui fabrique un flux en appliquant inductivement une fonction à un entier. D'autres constructions classiques, comme les listes, peuvent être importées avec les ingrédients de Λ_{pv}. Pour une présentation plus détaillée, nous renvoyons aux travaux de Ehrhard et Tasson [ET16].

3.2.2 Un survol de la sémantique dénotationnelle

3.2.2.1 Morphismes et coalgèbres

Nous donnons un rapide aperçu de la sémantique dénotationnelle de l'Appel-Par-Pousse-Valeur qui motive l'introduction du calcul à ressources de la section 3.3. Cette sémantique est basée sur celles de la Logique Linéaire, avec laquelle coïncident les types de Λ_{pv}.

Nous décrivons d'abord brièvement en quoi consiste un modèle de la Logique Linéaire (voir Melliès [Mel09] pour une présentation plus détaillée). Un tel modèle est donné par une catégorie $\mathcal{L}$, munie d'une *structure monoïdale symétrique* $(\otimes, 1, \lambda, \rho, \alpha, \sigma)$ qui est *close*[3] et nous écrivons $X \multimap Y$ pour représenter l'*objet des morphismes linéaires*. La catégorie a une structure *cartésienne*, avec un produit cartésien & et un objet terminal $\top$. $\mathcal{L}$ est également équipée

3. La plupart des modèles de la sorte qui sont étudiés sont aussi $*$-autonomes : il y a un objet $\perp$ tel que tout objet X soit isomorphe à $(X \multimap \perp) \multimap \perp$

d'une *comonade* $! : \mathcal{L} \to \mathcal{L}$, avec pour co-unité $\mathbf{der}_X \in \mathcal{L}(!X, X)$ et pour co-multiplication $\mathbf{dig}_X \in \mathcal{L}(!X, !!X)$. Cette comonade est munie d'une structure monoïdale symétrique [4] de $(\mathcal{L}, \&)$ vers $(\mathcal{L}, \otimes)$, qui consiste en les isomorphismes naturels $m^0 \in \mathcal{L}(1, !\top)$ et $m^2 \in \mathcal{L}(!X \otimes !Y, !(X \,\&\, Y))$.

En utilisant les isomorphismes m^0 et m^2 ; la fonctorialité de la comonade ! et la structure cartésienne, on peut construire une structure de *comonoïde* sur n'importe quel $!X$, qui permet l'effacement et la duplication des ressources, comme nous allons le voir.

$$\mathrm{suppr}_{!X} \in \mathcal{L}(!X, 1) \qquad \mathrm{split}^2_{!X} \in \mathcal{L}(!X, !X \otimes !X)$$

Une *coalgèbre* (P, h_P) est composée d'un objet P et d'un morphisme $h_P \in \mathcal{L}(P, !P)$ qui est compatible avec la structure de comonade, avec $\mathbf{der}_P h_P = \mathbf{Id}$ et $\mathbf{dig}_P h_P = !h_P h_P$. Toute coalgèbre hérite de la structure comonoïdale de $!P$ qui est équipée de : $\mathrm{suppr}_P \in \mathcal{L}(P, 1)$ et $\mathrm{split}^2_P \in \mathcal{L}(P, P \otimes P)$ définis comme :

$$\mathrm{suppr}_P : \; P \xrightarrow{h_P} !P \xrightarrow{w_P} 1 \qquad \mathrm{split}^2_P : \; P \xrightarrow{h_P} !P \xrightarrow{c_P} !P \otimes !P \xrightarrow{\mathbf{der}_P \otimes \mathbf{der}_P} P \otimes P.$$

Par un calcul similaire, on peut définir $\mathrm{split}^k_P \in \mathcal{L}(P, \underbrace{P \otimes \cdots \otimes P}_{k})$.

Remarquons que la structure de comonade de ! induit une structure de coalgèbre sur $!X$. De plus, toute construction sur un type positif préserve la structure de coalgèbre. Pour définir la structure coalgébrique de $P \otimes Q$, où P et Q sont tous deux des coalgèbres, définissons d'abord les morphismes $\mu^0 \in \mathcal{L}(1, !1)$ et $\mu^2 \in \mathcal{L}(!X \otimes !Y, !(X \otimes Y))$ comme :

$$\mu^0 : 1 \xrightarrow{m^0} !\top \xrightarrow{\mathbf{dig}_\top} !!\top \xrightarrow{!(m^0)^{-1}} !1$$

$$\mu^2 : !X \otimes !Y \xrightarrow{m^2} !(X \,\&\, Y) \xrightarrow{\mathbf{dig}_{X\&Y}} !!(X \,\&\, Y) \xrightarrow{!(m^2)^{-1}} !(!X \otimes !Y) \xrightarrow{!(\mathbf{der}_X \otimes \mathbf{der}_Y)} !(X \otimes Y).$$

On peut alors définir $h_{P\otimes Q} : P \otimes Q \xrightarrow{h_P \otimes h_Q} !P \otimes !Q \xrightarrow{\mu^2} !(P \otimes Q)$. La structure coalgébrique du coproduit est entièrement définie par les morphismes pour $i \in \{1, 2\} : P_i \xrightarrow{h_{P_i}} !P_i \xrightarrow{!\mathrm{in_i}} !(P_1 \oplus P_2)$.

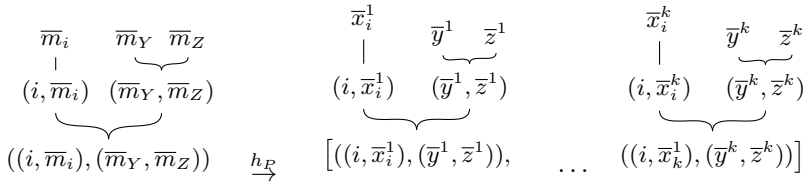
On peut alors déduire que tout type positif est interprété par une coalgèbre.

3.2.2.2 Un exemple : Le modèle relationnel

Décrivons ces constructions dans le *modèle relationnel* de la Logique Linéaire. La catégorie **Rel** est celle des ensembles et des relations. Le produit tensoriel est donné par le produit cartésien ensembliste, et son unité est l'ensemble singleton dont l'unique élément est noté $*$. Le produit est donné par l'union disjointe, et l'objet terminal est l'ensemble vide. **Rel** peut être équipée de la comonade des multiensembles finis, où la structure comonadique de $!X$ est

$$\mathbf{der}_X = \{([a], a) \mid a \in X\} \qquad \mathbf{dig}_X = \{(\overline{m}, [\overline{m}_1, \ldots, \overline{m}_k]) \mid \overline{m}_1 + \cdots + \overline{m}_k = \overline{m}\}.$$

4. Les deux isomorphismes m^0 et m^2 correspondent à ce que l'on appelle *isomorphismes de Seely*.



où $\sum_{j=1}^{k} \overline{x}_i^j = \overline{m}_i$, $\sum_{j=1}^{k} \overline{y}^j = \overline{m}_Y$, et $\sum_{j=1}^{k} \overline{z}^j = \overline{m}_Z$.

FIGURE 3.4 – Action du morphisme de coalgèbre h_P sur un type positif

La structure comonoïdale de $!X$ est

$$\mathrm{suppr}_{!X} = \{([], *)\} \qquad \mathrm{split}^2_{!X} = \{(\overline{m}, (\overline{m}_1, \overline{m}_2)) \mid \overline{m}_1 + \overline{m}_2 = \overline{m}\}.$$

Un type positif est une combinaison finie de $\oplus, \otimes, !$. Par exemple, si $P = (!X_1 \oplus !X_2) \otimes (!Y \otimes !Z)$, alors P est une coalgèbre (voir la figure 3.4) :

$$h_P = \{((i, \overline{m}_i)\ ,\ (\overline{m}_Y, \overline{m}_Z)), [((i, \overline{x}_i^1), (\overline{y}^1, \overline{z}^1)), \ldots, ((i, \overline{x}_i^k), (\overline{y}^k, \overline{z}^k))] \mid \\ \overline{m}_i = \overline{x}_i^1 + \cdots + \overline{x}_i^k, \overline{m}_Y = \overline{y}_1 + \cdots + \overline{y}_k, \overline{m}_Z = \overline{z}_1 + \cdots + \overline{z}_k\},$$

et est équipée avec la structure comonoïdale :

$$\mathrm{suppr}_P = \{(((i, []), ([], []), *)\}$$
$$\mathrm{split}^2_P = \{((i, \overline{m}_i), (\overline{m}_Y, \overline{m}_Z)\ ,\ ((i, (\overline{m}_i^1 + \overline{m}_i^2)), ((\overline{m}_Y^1 + \overline{m}_Y^2), (\overline{m}_Z^1 + \overline{m}_Z^2))) \mid \\ \overline{m}_i^1 + \overline{m}_i^2 = \overline{m}_i, \overline{m}_Y^1 + \overline{m}_Y^2 = \overline{m}_Y, m_Z^1 + \overline{m}_Z^2 = \overline{m}_Z\}\}$$

Remarquons que les morphismes structurels sont les mêmes que ceux de $!X$, mais ils agissent au niveau des feuilles de la structure d'arbre qui décrit la formule P.

3.3 Calcul à ressources pour Λ_{pv}

Nous introduisons un calcul à ressources typé qui permettra de simuler la sémantique opérationnelle de Λ_{pv}. La construction conditionnelle est considérée à travers des tests d'égalité, et il n'y a pas de point fixe explicite. La différence principale avec d'autres calculs à ressources comme en Appel-Par-Nom ou en Appel-Par-Valeur, est que les redex de la forme $\langle \lambda x m \rangle \overline{n}$ ne suffisent pas à capturer les réductions de Λ_{pv}. En effet, la notion de valeur est trop large pour n'être approchée que par des multiensembles de dermes : $\langle \lambda x M \rangle (V_1, V_2)$ est un redex dans Λ_{pv}, on doit donc être en mesure de réduire des termes à ressources de la forme $\langle \lambda x m \rangle (v_1, v_2)$, en conservant la sensibilité du calcul à la consommation de ressources. Nous nous accommodons de cette difficulté grâce à l'introduction d'un opérateur de découpage, qui permet de dupliquer une valeur en utilisant la structure de son type positif, et qui simule la construction split de la sémantique.

3.3.1 Calcul et types

Définition 30 (Appel-Par-Pousse-Valeur à ressources Δ_{pv})**.**
La syntaxe des types est la même que celle de Λ_{pv}.

$$\begin{aligned}\Delta_{\mathsf{pv}} : m ::= {} & x \mid 1 \mid 2 \mid \lambda x m \mid \langle m \rangle m \mid (m = m) \cdot m \mid (m, m) \mid \pi_1(m) \mid \pi_2(m) \\ & \mid [m, \ldots, m] \mid \mathbf{der}(m)\end{aligned}$$

On distingue les valeurs du calcul :

$$v ::= x \mid 1 \mid 2 \mid [m, \ldots, m] \mid (v, v)$$

$$\frac{}{\Gamma, x : A \vdash x : A} \quad \frac{\Gamma \vdash m_i : I,\ i \in \{1, \ldots, k\}}{\Gamma \vdash [m_1, \ldots, m_k] : !I} \quad \frac{\Gamma, x : A \vdash m : B}{\Gamma \vdash \lambda x m : A \multimap B}$$

$$\frac{\Gamma \vdash m : A \multimap I \qquad \Delta \vdash n : A}{\Gamma, \Delta \vdash \langle m \rangle n : I} \quad \frac{\Gamma \vdash m : !A}{\Gamma \vdash \mathbf{der}(m) : A}$$

$$\frac{\Gamma \vdash m : A \qquad \Delta \vdash n : B}{\Gamma, \Delta \vdash (m, n) : A \otimes B} \quad \frac{\Gamma \vdash m : A_1 \otimes A_2}{\Gamma \vdash \pi_i(m) : A_i}\ i \in \{1, 2\}$$

$$\frac{\Gamma \vdash m : A_i}{\Gamma \vdash (i, m) : A_1 \oplus A_2}\ i \in \{1, 2\}$$

$$\frac{\Gamma \vdash m_1 : A_1 \oplus A_2 \qquad \Delta \vdash m_2 : A_i \qquad \Theta \vdash m_3 : I}{\Gamma, \Delta, \Theta \vdash (m_1 = (i, m_2)) \cdot m_3 : I}$$

FIGURE 3.5 – Règles de typage pour Δ_{pv}

3.3.2 Découpage, réduction et point fixe

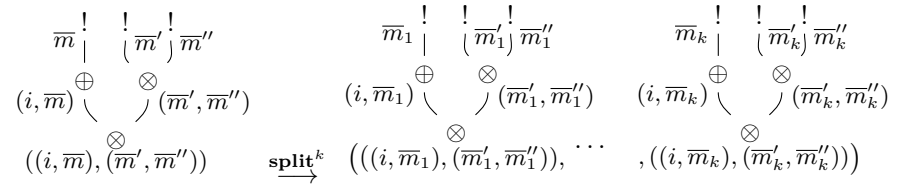
De façon à définir la dynamique du calcul à ressources que nous venons de mettre en place, nous introduisons une nouvelle construction $\mathbf{split}^k$. Sa sémantique opérationnelle consiste à dupliquer des valeurs de base, comme des variables ou des entiers, et à les répartir dans les feuilles de la structure d'arbre induite par les paires et les injections. Un exemple est donné en figure 3.6.

Cet opérateur de découpage est la contrepartie syntaxique du morphisme associé, dans la sémantique, à chaque coalgèbre P qui interprète un type positif : $\mathrm{split}^k_P \in \mathcal{L}(P, \underbrace{P \otimes \cdots \otimes P}_{k})$ (Voir section 3.2.2.1).

Définition 31 (Découpage **split**)**.**

- $\mathbf{split}^k(\overline{m}) = \{(\overline{m}_1, \ldots, \overline{m}_k) \mid \sum_{i=1}^k \overline{m}_i = \overline{m}\}$
- $\mathbf{split}^k(x) = \{(x, \ldots, x)_k\}$
- $\mathbf{split}^k(i) = \{(i, \ldots, i)_k\}$ pour $i \in \{1, 2\}$.
- $\mathbf{split}^k((m, n)) = \{((m_1, n_1), \ldots, (m_k, n_k))$
  $\mid (m_1, \ldots, m_k) \in \mathbf{split}^k(m), (n_1, \ldots, n_k) \in \mathbf{split}^k(n)\}$.

On peut maintenant définir les règles de réduction associées à Δ_{pv}, en ajoutant le terme distingué 0 au calcul.



où $\sum_{i=1}^k \overline{m}_i = \overline{m}$, $\sum_{i=1}^k \overline{m}_i' = \overline{m}'$, et $\sum_{i=1}^k \overline{m}_i'' = \overline{m}''$.

FIGURE 3.6 – Découpage d'une valeur, l'arbre du type positif est étiqueté par les ressources corespondantes.

Définition 32.

- $\langle \lambda x m \rangle n \to_{\mathsf{rpv}} m[n_1/x_1, \ldots, n_k/x_k]$ si $\mathsf{deg}_x(m) = k$ et si $(n_1, \ldots, n_k') \in \mathbf{split}^k(n)$.
- $(v = (i, v')) \cdot n \to_{\mathsf{rpv}} n$ si $v = (i, v')$. Sinon, $(v = (i, v')) \cdot n \to_{\mathsf{rpv}} 0$.
- $\mathbf{der}([m_1, \ldots, m_k]) \to_{\mathsf{rpv}} m_1$ si $k = 1$, et $\mathbf{der}([m_1, \ldots, m_k]) \to_{\mathsf{rpv}} 0$ sinon.
- $\pi_i((m_1, m_2)) \to_{\mathsf{rpv}} m_i$

On définit les contextes d'évaluation e, pour tous termes m, n de Δ_{pv} :

$$e ::= [] \mid \langle e \rangle m \mid \langle m \rangle e \mid \lambda x e \mid (e, m) \mid (m, e) \mid (e = m) \cdot n \mid (m = e) \cdot n \mid \mathbf{der}(e)$$

et posons comme règle supplémentaire $e[m] \to_{\mathsf{rpv}} e[n]$ si $m \to_{\mathsf{rpv}} n$ par l'une des règles précédentes, avec $e[0] = 0$ pour tout contexte e.

On est tenu de contraindre les tests d'égalité aux valeurs, et on ne peut les admettre pour des termes arbitraires, car cela entraînerait la perte de la confluence du calcul : par exemple, si $m = (\pi_1(m_1, m_2) = m_1) \cdot n$ était un redex, alors m se réduirait en 0, mais se réduit également en $(m_1 = m_1) \cdot n$, qui se réduit à son tour en n.

Nous établissons une première propriété du calcul, qui assure qu'il se comporte convenablement par rapport au typage.

Proposition 3 (Réduction du sujet). Pour tous termes m, n, et tout type général I, si $m : I$ et $m \to_{\mathsf{rpv}} n$, alors $n : I$.

Démonstration. Par induction sur m.

- Si $m = (\pi_i(m_1, m_2))$ et si $n = m_i$, alors il existe A_1, A_2 tels que $m_i : A_i$, et on a $m : A_i$ et $n : A_i$.
- Si $m = \mathbf{der}([n])$, alors il existe un type J tel que $n : J$, et on a $[n] : !J$ et $m : J$.
- Si $m = (v_1 = (i, v_2)) \cdot n$, alors si $n : J$ pour un certain type J, $m : J$.
- Si $m = \langle \lambda x m' \rangle v$ et $n = m'[v_1/x_1, \ldots, v_k/x_k]$ pour $k = \mathsf{deg}_x(m')$ et $(v_1, \ldots, v_k) \in \mathbf{split}^k(v)$, alors si $x : A, v : A, m' : J$, et $\lambda x m' : A \multimap J$, pour des types A, J, on a $m : J$. Pour conclure $n : J$, il suffit de s'assurer que pour tout $i \in \{1, \ldots, k\}$, $v_i : A$ ce qui se fait simplement par induction sur v, et qui implique $m'[v_1/x_1, \ldots, v_k/x_k] : A$. Ce dernier point se traite par un argument standard.

— Si $m = e[m']$ et $n = e[n']$ pour $n \to_{\mathsf{rpv}} n'$, on conclut par hypothèse de récurrence.

□

On définit pour tout $k \in \mathbf{N}$, toute variable x et tout $m \in \Delta_{\mathsf{pv}}$, un ensemble de termes $\mathbf{fix}_x^k(m)$, comme suit. Pour $x_1, \ldots, x_{\mathsf{deg}_x(m)}$ les occurrences libres de x dans m, on pose d'abord $\mathbf{fix}_x^0(m) = m[[]/x_1, \ldots, []/x_{\mathsf{deg}_x(m)}]$ puis pour tout $k \in \mathbf{N}$: $\mathbf{fix}_x^{k+1}(m) =$

$$\left\{m\left[\overline{m}_1/x_1, \ldots, \overline{m}_{\mathsf{deg}_x(m)}/x_{\mathsf{deg}_x(m)}\right] \mid \forall i \leq \mathsf{deg}_x(m) : \overline{m}_i \in (\mathbf{fix}_x^k(m))^!\right\}.$$

Cette définition provient de la dynamique du point fixe de Λ_{pv}, dont on rappelle la règle de réduction : $\mathbf{fix}_x(M) \to_{\mathsf{pv}} M[\mathbf{fix}_x(M)^!/x]$. Les termes à ressources des ensembles $\mathbf{fix}_x^k(m)$ correspondent à un nombre fini (inférieur à k) d'applications de cette règle.

3.4 Développement de Taylor

Nous introduisons dans cette section d'abord une version qualitative du développement de Taylor, pris comme ensemble d'approximations, à travers laquelle nous montrons une première propriété de simulation (proposition 4), et présentons comment les plongements de l'Appel-Par-Nom et de l'Appel-Par-Valeur dans l'Appel-Par-Pousse-Valeur, se comportent au niveau des ressources (propriété 4). Ensuite, nous introduisons les coefficients pour considérer le développement de Taylor complet, c'est-à-dire avec des coefficients. Le lemme 45 assure qu'il ne conduit pas à des problèmes de divergence, grâce à une propriété de finitude de l'antiréduction (ingrédient nécessaire, et qui a occupé l'essentiel du chapitre 2 pour les réseaux de preuve). Enfin, nous prouvons la simulation des réductions de Λ_{pv} dans le développement complet, en montrant que les coefficients commutent à ces réductions, dans le théorème 9 (qui correspond au théorème 8 pour les réseaux de preuve).

3.4.1 Définitions et simulation

Définition 33 (Support du développement de Taylor)**.**
On définit les ensembles de termes à ressources correspondant au support du développement de Taylor de Λ_{pv} :

— $\mathcal{T}_{\mathsf{pv}}(x) = \{x\}$
— $\mathcal{T}_{\mathsf{pv}}\langle M\rangle N = \{\langle m\rangle n \mid m \in \mathcal{T}_{\mathsf{pv}}(M), n \in \mathcal{T}_{\mathsf{pv}}(N)\}$
— $\mathcal{T}_{\mathsf{pv}}(\iota_i(M)) = \{(i, m) \mid m \in \mathcal{T}_{\mathsf{pv}}(M)\}$ (pour $i \in \{1, 2\}$)
— $\mathcal{T}_{\mathsf{pv}}(\mathbf{der}(M)) = \{\mathbf{der}(m) \mid m \in \mathcal{T}_{\mathsf{pv}}(M)\}$
— $\mathcal{T}_{\mathsf{pv}}(M^!) = \mathcal{T}_{\mathsf{pv}}(M)^!$
— $\mathcal{T}_{\mathsf{pv}}((M, N)) = \{(m, n) \mid m \in \mathcal{T}_{\mathsf{pv}}(M), n \in \mathcal{T}_{\mathsf{pv}}(N)\}$
— $\mathcal{T}_{\mathsf{pv}}(\pi_i(M)) = \{\pi_i(m) \mid m \in \mathcal{T}_{\mathsf{pv}}(M)\}$
— $\mathcal{T}_{\mathsf{pv}}(\mathbf{fix}_x(M)) = \{\mathbf{fix}_x^k(m) \mid m \in \mathcal{T}_{\mathsf{pv}}(M), k \in \mathbf{N}\}$
— $\mathcal{T}_{\mathsf{pv}}(\lambda x M) = \{\lambda x m \mid m \in \mathcal{T}_{\mathsf{pv}}(M)\}$

— $\mathcal{T}_{\mathsf{pv}}(\mathbf{case}(M, z_1 \cdot N_1, z_2 \cdot N_2)) = \{(m = (i, m')) \cdot n_i[m'/z_i] \mid i \in \{1, 2\}, m \in \mathcal{T}_{\mathsf{pv}}(M), n_i \in \mathcal{T}_{\mathsf{pv}}(N_i), m' \in \Delta_{\mathsf{pv}}\}$

La propriété ci-dessous permet de s'assurer que le développement de Taylor est cohérent avec l'Appel-Par-Pousse-Valeur et sa sémantique, dans la mesure où le découpage **split** est défini sur les termes qui correspondent à des valeurs, et sont donc de type positif.

Propriété 3. Soit $m \in \mathcal{T}_{\mathsf{pv}}(M)$. Pour un terme $M \in \Lambda_{\mathsf{pv}}$ et $k \in \mathbf{N}$, $\mathbf{split}^k(m)$ est défini si et seulement si M est une valeur.

Démonstration. On vérifie facilement que la syntaxe des termes à ressource v qui sont dans $\mathcal{T}_{\mathsf{pv}}(V)$ pour une valeur V correspond exactement aux valeurs à ressources de la définition 30. De même, on constate que $\mathbf{split}^k(v)$ est toujours défini pour les valeurs à ressources, mais que si $m \in \mathcal{T}_{\mathsf{pv}}(M)$ n'en est pas une, alors $\mathbf{split}^k(m)$ n'est pas défini. □

Le corollaire suivant établit que Δ_{pv} est cohérent avec Λ_{pv} de la façon suivante : une approximation m d'un terme M de Λ_{pv} est un redex de Δ_{pv} si et seulement si M est un redex de Λ_{pv}. Cette propriété est triviale pour la plupart des termes, mais moins pour ceux de la forme $\langle \lambda x m \rangle n$ (respectivement $\langle \lambda x M \rangle N$), où elle est une conséquence de la propriété 3.

Corollaire 9. Soit $\langle \lambda x m \rangle n \in \mathcal{T}_{\mathsf{pv}}(\langle \lambda x M \rangle N)$. Il existe un terme m' tel que $\langle \lambda x m \rangle n \to_{\mathsf{rpv}} m'$ en réduisant le redex le plus externe si et seulement si N est une valeur. On se rappelle par ailleurs (définition 29) que $\langle \lambda x M \rangle N \to_{\mathsf{pv}} M[N/x]$ si et seulement N est une valeur.

Lemme 40. Si M est une valeur, $k \in \mathbf{N}$, $m \in \mathcal{T}_{\mathsf{pv}}(M)$ et $(m_1, \ldots, m_k) \in \mathbf{split}^k(m)$ alors pour tout $i \in \{1, \ldots, k\}$, $m_i \in \mathcal{T}_{\mathsf{pv}}(M)$.

Démonstration. Par induction sur M, où la propriété 3 assure que $\mathbf{split}^k(m)$ est toujours défini :

— Si $M = x$, alors $m = x$ et $\mathbf{split}^k(m) = (x, \ldots, x)_k$. On conclut car $\mathcal{T}_{\mathsf{pv}}(x) = \{x\}$.

— Si $M = N^!$, alors $m = [n_1, \ldots, n_l]$, et pour tout $i \in \{1, \ldots, l\}$, $n_i \in \mathcal{T}_{\mathsf{pv}}(N)$. On a $(m_1, \ldots, m_k) = (\overline{n}_1, \ldots, \overline{n}_k)$ avec $\sum_{i=1}^k \overline{n}_i = [n_1, \ldots, n_l]$. Donc, chaque $\overline{n}_i$ est un multiensemble d'éléments de $\mathcal{T}_{\mathsf{pv}}(N)$, et $\overline{n}_i \in \mathcal{T}_{\mathsf{pv}}(N^!) = \mathcal{T}_{\mathsf{pv}}(M)$.

— Si $M = (N, N')$, alors $m = (n, n')$ pour $n \in \mathcal{T}_{\mathsf{pv}}(N)$ et $n' \in \mathcal{T}_{\mathsf{pv}}(N')$. $(m_1, \ldots, m_k) = ((n_1, n'_1), \ldots, (n_k, n'_k))$ avec $(n_1, \ldots, n_k) \in \mathbf{split}^k(N)$ et $(n'_1, \ldots, n'_k) \in \mathbf{split}^k(N')$. Par hypothèse de récurrence, pour tout $i \in \{1, \ldots, k\}$, $n_i \in \mathcal{T}_{\mathsf{pv}}(N)$ et $n'_i \in \mathcal{T}_{\mathsf{pv}}(N')$. Donc pour tout i, $(n_i, n'_i) \in \mathcal{T}_{\mathsf{pv}}(N, N') = \mathcal{T}_{\mathsf{pv}}(M)$.

— Si $M = \iota_j(N)$, alors $m = (j, n)$ pour $n \in \mathcal{T}_{\mathsf{pv}}(N)$, et on a $\mathbf{split}^k(m) = ((j, n_1), \ldots, (j, n_k))$ avec $(n_1, \ldots, n_k) \in \mathbf{split}^k(n)$. Par hypothèse de récurrence, pour tout $i \in \{1, \ldots, k\}$, $n_i \in \mathcal{T}_{\mathsf{pv}}(N)$. Donc pour tout i, $(j, n_i) \in \mathcal{T}_{\mathsf{pv}}(\iota_j(N)) = \mathcal{T}_{\mathsf{pv}}(M)$.

□

Le lemme de substitution ci-dessous est crucial pour assurer que le développement de Taylor est compatible avec la réduction de Λ_{pv}. Nous l'utiliserons pour prouver la simulation en proposition 4.

Lemme 41 (Substitution). Soit $m \in \mathcal{T}_{\mathsf{pv}}(M)$, $k = \mathsf{deg}_x(m)$, $x_1, \ldots, x_k$ les occurrences libres de x dans m, et $n_1, \ldots, n_k \in \mathcal{T}_{\mathsf{pv}}(N)$, pour $M, N \in \Lambda_{\mathsf{pv}}$. On a $m[n_1/x_1, \ldots, n_k/x_k] \in \mathcal{T}_{\mathsf{pv}}(M[N/x])$.

Démonstration. La preuve est par récurrence sur M. On ne considère que les cas représentatifs, les autres pouvant se déduire d'applications similaires de l'hypothèse de récurrence.

— Si $M = x$, alors $m = x, k = 1$, $m[n_1/x_1] = n_1$, et $M[N/x] = N$. Donc $m[n_1/x_1] \in \mathcal{T}_{\mathsf{pv}}(M[N/x])$.

— Si $M = \lambda y M'$, alors $\mathsf{deg}_x(M) = \mathsf{deg}_x(M'), m = \lambda y m'$ pour $m' \in \mathcal{T}_{\mathsf{pv}}(M')$. Par hypothèse de récurrence, $m'[n_1/x_1, \ldots, n_k/x_k] \in \mathcal{T}_{\mathsf{pv}}(M'[N/x])$. Dans la mesure où l'on a $m[n_1/x_1, \ldots, n_k/x_k] = \lambda y m'[n_1/x_1, \ldots, n_k/x_k]$, on conclut.

— Si $M = \langle M_1 \rangle M_2$, alors $m = \langle m_1 \rangle m_2$ pour $m_i \in \mathcal{T}_{\mathsf{pv}}(M_i)$, et $\mathsf{deg}_x(m) = l_1 + l_2$ pour $l_1 = \mathsf{deg}_x(m_1)$ et $l_2 = \mathsf{deg}_x(m_2)$. Par hypothèse de récurrence, on a :

$$m_1[n_1/x_1, \ldots, n_{l_1}/x_{l_1}] \in \mathcal{T}_{\mathsf{pv}}(M_1[N/x])$$
$$m_2[n_{l_1+1}/x_{l_1+1}, \ldots, n_{l_1+l_2}/x_{l_1+l_2}] \in \mathcal{T}_{\mathsf{pv}}(M_2[N/x])$$

De plus :

$$m[n_1/x_1, \ldots, n_k/x_k] = \langle m_1[n_1/x_1, \ldots, n_{l_1}/x_{l_1}] \rangle m_2[n_{l_1+1}/x, \ldots, n_{l_1+l_2}/x]$$

Comme $M[N/x] = \langle M_1[N/x] \rangle M_2[N/x]$, on conclut.

— Si $M = M'^{!}$, alors $m = [m'_1, \ldots, m'_l]$ avec $m'_i \in \mathcal{T}_{\mathsf{pv}}(M')$ pour tout i, et $\mathsf{deg}_x(m) = \sum_{i=1}^{l} k_i$ où $k_i = \mathsf{deg}_x(m'_i)$. Par hypothèse de récurrence, $m'_i[n_{k_{i-1}+1}/x_{k_{i-1}+1}, \ldots, n_{k_{i-1}+k_i}/x_{k_{i-1}+k_i}] \in \mathcal{T}_{\mathsf{pv}}(M'[N/x])$ pour tout $i \in \{1, \ldots, l\}$ (en posant $k_0 = 0$). Donc, $M[N/x] = (M'[N/x])^{!}$, et on peut conclure comme précédemment.

— Si $M = \mathbf{case}(M', z_1 \cdot N_1, z_2 \cdot N_2)$, alors $m = (m' = (i, m'')) \cdot n_i[m''/z_i]$ pour $i \in \{1, 2\}, m' \in \mathcal{T}_{\mathsf{pv}}(M'), n_i \in \mathcal{T}_{\mathsf{pv}}(N_i), m'' \in \Delta_{\mathsf{pv}}$. On conclut par hypothèse de récurrence comme ci-dessus.

□

Nous sommes maintenant en mesure de prouver la première propriété de simulation :

Proposition 4 (Simulation). Si $M \to_{\mathsf{pv}} M'$, alors pour tout $m \in \mathcal{T}_{\mathsf{pv}}(M)$, soit $m \to_{\mathsf{rpv}} 0$ soit il existe $m' \in \mathcal{T}_{\mathsf{pv}}(M')$ tel que $m \to^{=}_{\mathsf{rpv}} m'$, où $\to^{=}_{\mathsf{rpv}}$ est la clôture réflexive de $\to_{\mathsf{rpv}}$.

Démonstration. Par récurrence sur M :

— Si $M = \pi_i((M_1, M_2))$ et $M' = M_i$, alors $m = \pi_i((m_1, m_2))$ pour $m_i \in \mathcal{T}_{\mathsf{pv}}(M_i)$. On peut conclure, car $M \to_{\mathsf{pv}} M_i$ et $m \to_{\mathsf{rpv}} m_i$.

— Si $M = \mathbf{der}(N^!)$ et $M' = N$, alors $m = \mathbf{der}([n_1, \dots n_k])$, avec $n_i \in \mathcal{T}_{\mathsf{pv}}(N)$ pour tout $i \in \{1, \dots, k\}$. On conclut car $M \to_{\mathsf{pv}} N$, $m \to_{\mathsf{rpv}} n_1$ si $k = 1$ et $m \to_{\mathsf{rpv}} 0$ sinon.

— Si $M = \mathbf{fix}_x(N)$ et $M' = N[(\mathbf{fix}_x(N))^!/x]$, alors on vérifie simplement que $\mathcal{T}_{\mathsf{pv}}(M) = \mathcal{T}_{\mathsf{pv}}(M')$, en utilisant le lemme 41, et en dépliant les définitions du développement de Taylor et du point fixe. C'est pour ce cas que nous avons besoin d'une réduction réflexive.

— Si $M = (\lambda y N)V$ et $M' = N[V/y]$, alors $m = \langle \lambda y n \rangle v$ pour $n \in \mathcal{T}_{\mathsf{pv}}(N)$ et $v \in \mathcal{T}_{\mathsf{pv}}(V)$. Par la propriété 3, $\mathbf{split}^k(v)$ est défini pour tout $k \in \mathbf{N}$, donc $m \to_{\mathsf{rpv}} n[v_1/y_1, \dots, v_k/y_k]$ pour $\mathsf{deg}_y(n) = k$ et $(v_1, \dots, v_k) \in \mathbf{split}^k(v)$. Par le lemme 40, pour tout $i \in \{1, \dots, k\}$, $v_i \in \mathcal{T}_{\mathsf{pv}}(V)$, et par le lemme de substitution 41, $n[v_1/y_1, \dots, v_k/y_k] \in \mathcal{T}_{\mathsf{pv}}(N[V/y])$.

— Si $M = \mathbf{case}(\iota_i(V), x_1 \cdot M_1, x_2 \cdot M_2)$ et $M' = M_i[V/x_i]$, alors, $m = ((i, v) = (j, n)) \cdot m_i[v/x_i]$ pour $i, j \in \{1, 2\}, v \in \mathcal{T}_{\mathsf{pv}}(V), n \in \Delta_{\mathsf{pv}}, m_i \in \mathcal{T}_{\mathsf{pv}}(M_i)$. Soit $m \to_{\mathsf{rpv}} 0$, soit $(i, v) = (j, n)$ et dans ce cas $m \to_{\mathsf{rpv}} m_i[n/x_i] = m_i[v/x_i]$. Par le lemme de substitution 41 on conclut, car on a $M \to_{\mathsf{pv}} M_i[V/x_i]$ et $m_i[v/x_i] \in \mathcal{T}_{\mathsf{pv}}(M_i[V/x_i])$.

— Si $M = E[N]$ et $M' = E[N']$, alors on peut aisément montrer qu'il existe un contexte à ressources e tel que $m = e[n]$, et $n \in \mathcal{T}_{\mathsf{pv}}(N)$. Par hypothèse du récurrence, soit $n \to_{\mathsf{rpv}} 0$, et alors $e[n] = 0$, soit il existe n' tel que $n \to_{\mathsf{rpv}} n'$ et $n' \in \mathcal{T}_{\mathsf{pv}}(N')$. On peut facilement adapter le lemme de substitution pour conclure $e[n'] \in \mathcal{T}_{\mathsf{pv}}(E[N'])$.

□

3.4.2 Plongements de l'Appel-Par-Nom et de l'Appel-Par-Valeur

l'Appel-Par-Pousse-Valeur est connu pour subsumer à la fois les stratégies d'Appel-Par-Nom et d'Appel-Par-Valeur. En particulier, les deux peuvent être plongées dans Λ_{pv}. Si l'on considère le λ-calcul simplement typé [5] Λ, on donne deux fonctions $()^v, ()^n : \Lambda \to \Lambda_{\mathsf{pv}}$, définies en figure 3.7. On ne considère pas ici

Traduction Appel-Par-Nom	Traduction Appel-Par-Valeur
$(x)^n = \mathbf{der}(x)$	$(x)^v = \mathbf{der}(x)^!$
$(MN)^n = \langle M^n \rangle (N^n)^!$	$(MN)^v = \langle \mathbf{der}(M) \rangle N$
$(\lambda x M)^n = \lambda x M^n$	$(\lambda x M)^v = (\lambda x M^v)^!$

FIGURE 3.7 – Deux traductions, $()^n$ et $()^v$, de Λ vers Λ_{pv}.

de calcul avec produits, ou autres constructeurs explicites, de façon à se concentrer dans un cadre plus simple sur la relation entre les exponentielles et les stratégies de réduction (voir les travaux d'Ehrhard et Tasson [ET16] pour les extensions). Nos traductions assurent par exemple les propriétés suivantes (pour plus de précisions sur ces traductions, voir les travaux originaux de Levy [Lev06]) :

5. Nous n'expliciterons pas le typage, car les traductions fonctionnent de la même façon qu'avec le λ-calcul pur (par exemple quand il est traduit dans les réseaux de preuve). Mais comme le calcul cible, Λ_{pv}, est typé, cette restriction est nécessaire.

$((\lambda x M)N)^v \to_{\mathsf{pv}} (M[N/x])^v$ si et seulement si N est une variable ou une abstraction, et $((\lambda x M)N)^n \to_{\mathsf{pv}} (M[N/x])^n$ pour tous M, N. On parvient donc bien à isoler les deux stratégies : on rappelle qu'en Appel-Par-Valeur usuel, les valeurs sont bien les variables et les abstractions.

Du point de vue du développement de Taylor, soient $\mathcal{T}^n$ et $\mathcal{T}^v$ respectivement le développement de Taylor classique (Ehrhard et Regnier [ER03]), par nom, et le développement par valeur (introduit par Ehrhard [Ehr12], et étudié de façon plus approfondie par Kerinec, Manzonetto et Pagani [KMP18]). On peut vérifier la correction de la construction de notre calcul à ressources Δ_{pv} et de son développement $\mathcal{T}_{\mathsf{pv}}$ par rapport à ces traductions, en utilisant $\mathcal{T}^n$ et $\mathcal{T}^v$ qui sont définis en figure 3.8. Le premier est défini sur Δ^n, qui est le calcul à ressources

Développement de Taylor par nom
$\mathcal{T}^n(x) = \{x\}$
$\mathcal{T}^n(M_1 M_2) = \{\langle m_1\rangle \overline{m}_2 \mid m_1 \in \mathcal{T}^n(M_1), \overline{m}_2 \in \mathcal{T}^n(M_2)^!\}$
$\mathcal{T}^n(\lambda x M) = \{\lambda x m \mid m \in \mathcal{T}^n(M)\}$

Développement de Taylor par valeur
$\mathcal{T}^v(x) = \{[x]_k \mid k \in \mathbf{N}\}$
$\mathcal{T}^v(M_1 M_2) = \{\langle m_1\rangle m_2 \mid m_i \in \mathcal{T}^v(M_i)\}$
$\mathcal{T}^v(\lambda x M) = \{[\lambda x m_1, \dots, \lambda x m_k] \mid m_i \in \mathcal{T}^v(M)\}$

FIGURE 3.8 – $\mathcal{T}^v : \Lambda \to P(\Delta^v)$ et $\mathcal{T}^n : \Lambda \to P(\Delta^n)$

original de Ehrhard et Regnier [ER03], et le second est défini sur Δ^v, un calcul à ressources par valeurs introduit par Ehrhard [Ehr12]. Les deux sont décrits en figure 3.9.

Δ^n	Δ^v
$m, n ::= x \mid \lambda x m \mid \langle m\rangle \overline{n}$	$m, n ::= [x]_k \mid [\lambda x m_1, \dots, \lambda x m_k] \mid \langle m\rangle n$
$\langle \lambda x m\rangle [n_1, \dots, n_k]$ $\to m[n_1/x_{f(1)}, \dots, n_k/x_{f(k)}]$ si $k = \mathsf{deg}_x(m)$ et $f \in \mathfrak{S}_k$	$\langle [\lambda x m]\rangle [n_1, \dots, n_k]$ $\to m[n_1/x_{f(1)}, \dots, n_k/x_{f(k)}]$ si $k = \mathsf{deg}_x(m)$ et $f \in \mathfrak{S}_k$

FIGURE 3.9 – Calculs à ressources par nom et par valeur

Propriété 4. Soit $\mathbf{e}$ la fonction qui supprime toutes les dérélictions d'un ensemble de termes (les dérélictions n'existent pas dans Δ^n ni dans Δ^v). Pour tout terme $M \in \Lambda$, $\mathbf{e}(\mathcal{T}_{\mathsf{pv}}((M)v)) = \mathcal{T}^v(M)$ et $\mathbf{e}(\mathcal{T}_{\mathsf{pv}}((M)^n)) = \mathcal{T}^n(M)$.

Démonstration. La preuve consiste en une récurrence sur M *via* un simple examen des définitions. Commençons par les constructions d'Appel-Par-Valeur : Le cas de la variable est immédiat, car $\mathcal{T}_{\mathsf{pv}}(x^v) = \{\mathbf{der}(x)\}^!$, et $\mathcal{T}^v(x) = \{x\}^!$. $\mathcal{T}_{\mathsf{pv}}((\lambda x M)^v) = \mathcal{T}_{\mathsf{pv}}((\lambda x M^v)^!) = \{[\lambda x m_1, \dots, \lambda x m_k] \mid k \in \mathbf{N}, m_i \in \mathcal{T}_{\mathsf{pv}}(M^v)\}$, on conclut, car par hypothèse de récurrence, $\mathbf{e}(\mathcal{T}_{\mathsf{pv}}(M^v)) = \mathcal{T}^v(M)$, et car on a :

$$\mathcal{T}^v(\lambda x M) = \{[\lambda x m'_1, \dots, \lambda x m'_l] \mid l \in \mathbf{N}, m'_i \in \mathcal{T}^v(M)\}$$

Le cas de l'application est traité par un argument similaire avec un appel à l'hypothèse de récurrence, et par le fait que $\mathbf{e}(\langle \mathbf{der}(M)\rangle N) = \langle \mathbf{e}(M)\rangle \mathbf{e}(N)$.

Pour l'Appel-Par-Nom, on considère seulement le cas de l'application, les autres étant immédiats : $\mathcal{T}_{\mathsf{pv}}((MN)^n) = \{\langle m\rangle \overline{n} \mid m \in \mathcal{T}_{\mathsf{pv}}(M^n), \overline{n} \in \mathcal{T}_{\mathsf{pv}}(N^n)^!\}$. Par hypothèse de récurrence, $\mathbf{e}(\mathcal{T}_{\mathsf{pv}}(M^n)) = \mathcal{T}^n(M)$ et $\mathbf{e}(\mathcal{T}_{\mathsf{pv}}(N^n)) = \mathcal{T}^n(N)$, et on peut conclure. □

Avec la propriété de simulation de $\mathcal{T}_{\mathsf{pv}}$ (propriété 4), la Propriété 4 éclaire le fait que l'Appel-Par-Pousse-Valeur subsume bien l'Appel-Par-Nom et l'Appel-Par-Valeur, de façon à ce que cela reste valide au niveau des constructions relatives à la consommation de ressources.

3.4.3 Finitude

Dans cette section, nous montrons un résultat (analogue au théorème 6 pour les réseaux à ressources), qui permet également de considérer une version quantitative du développement de Taylor, et d'étendre la réduction du calcul à ressources à des séries infinies avec coefficients. Les conditions de validité de ce résultat, concernant la finitude de l'antiréduction, ont été largement étudiées [CV18, PTV16, Vau17] dans des systèmes non uniformes, probabilistes, et nous avons produit une étude [Cho19] de ces différentes méthodes de démonstrations relativement à différentes stratégies de réduction.

On définit une relation de cohérence, à la façon de Ehrhard et Regnier [ER08], entre les éléments de Δ_{pv}, dont on montre qu'elle est stable par réduction à ressources, et qui est telle que le développement de Taylor d'un terme de Λ_{pv} définisse toujours une clique maximale pour cette relation (un ensemble de termes deux-à-deux cohérents). Cela nous permettra d'établir le résultat de finitude de l'antiréduction : pour tous $n \in \Delta_{\mathsf{pv}}$ et $M \in \Lambda_{\mathsf{pv}}$, $\{m \in \mathcal{T}_{\mathsf{pv}}(M) \mid m \to_{\mathsf{pv}}^{=} n\}$ a au plus un élément.

Définition 34 (Relation de cohérence sur les termes de Δ_{pv})**.**

- $x \frown x$ pour tout x.
- $i \frown j$ si $i = j$, pour $i, j \in \{1, 2\}$.
- $\lambda x m \frown \lambda x m'$, $\pi_i(m) \frown \pi_i(m')$ et $\mathbf{der}(m) \frown \mathbf{der}(m')$ si $m \frown m'$.
- $\langle m\rangle n \frown \langle m'\rangle n'$ si $m \frown m'$ et $n \frown n'$.
- $(m_1 = m_2) \cdot n \frown (m_1' = m_2') \cdot n'$ si $m_1 \frown m_1'$ et si $m_2 = m_2' \to n \frown n'$
- $[m_1, \dots m_k] \frown [m_{k+1}, \dots, m_{k+l}]$ si $\forall i, j \in \{1, \dots, k+l\}, m_i \frown m_j$.

On remarque que dans les termes de la forme $(m_1 = m_2) \cdot n$, on ne demande pas que m_1 soit cohérent avec m_2. Cela tient au fait que nous voulons que le développement de Taylor soit une clique pour la cohérence : or, le développement de Taylor pourra donner des termes comme $(\pi_1((m_1, m_2)) = m_1) \cdot n$, où les termes dont l'égalité est testée ne sont pas cohérents entre eux avant réduction, mais doivent tout de même l'être avec les autres termes de cette forme issus du développement de Taylor.

Propriété 5. Soient $m \frown m'$, et $\{n_1, \dots, n_{\mathsf{deg}_x(m)}, n_1', \dots, n'_{\mathsf{deg}_x(m')}\}$ un ensemble de termes deux-à-deux cohérents. Alors, on a :

$$m[n_1/x_1, \dots, n_{\mathsf{deg}_x(m)}/x_{\mathsf{deg}_x(m)}] \frown m'[n_1'/x_1, \dots, n'_{\mathsf{deg}_x(m')}/x_{\mathsf{deg}_x(m')}]$$

Démonstration. La propriété se montre par une simple récurrence sur m. □

Lemme 42. Pour tout terme M de Λ_{pv}, $\mathcal{T}_{\mathsf{pv}}(M)$ est une clique pour $\frown$.

Démonstration. La preuve est par induction sur M. On détaille seulement le cas du point fixe, les autres constructions pouvant être traitées par de simples applications de l'hypothèse de récurrence. Soit $M = \mathbf{fix}_x(N)$. $\mathcal{T}_{\mathsf{pv}}(M) = \{\mathbf{fix}_x^k(n) \mid n \in \mathcal{T}_{\mathsf{pv}}(N), k \in \mathbf{N}\}$. On montre d'abord par induction sur k que $\mathbf{fix}_x^k(n)$ est une clique pour tout k, puis on montrera que les éléments de $\mathbf{fix}^{k+1}(n)$ sont cohérents avec ceux de $\mathbf{fix}_x^k(n)$.

— $\mathbf{fix}_x^0(n)$ est évidemment une clique, car c'est un singleton.

— Soient $m_1, m_2 \in \mathbf{fix}_x^{k+1}(n)$. $m_1 = n[\overline{n}_1/x_1, \ldots, \overline{n}_{\mathsf{deg}_x(n)}/x_{\mathsf{deg}_x(n)}]$ et $m' = n[\overline{n}'_1/x_1, \ldots, \overline{n}'_{\mathsf{deg}_x(n)}/x_{\mathsf{deg}_x(n)}]$ où tous les $\overline{n}_i$ et $\overline{n}'_j$ sont éléments de $\mathbf{fix}_x^k(n)^!$. Par hypothèse de récurrence, $\mathbf{fix}_x^k$ est une clique, donc tous les éléments de $\overline{n}_i$ et $\overline{n}'_j$ sont cohérents deux-à-deux. Pour conclure ce cas, on applique la propriété 5 qui entraîne bien $m_1 \frown m_2$.

Soient maintenant $m_1 \in \mathbf{fix}_x^{k+1}(n)$ et $m_2 \in \mathbf{fix}_x^k(n)$. Si $k = 0$, On a $m_1 = n[\overline{n}_1/x_1, \ldots, \overline{n}_{\mathsf{deg}_x(n)}/x_{\mathsf{deg}_x(n)}]$ et $m_2 = n[[]/x_1, \ldots, []/x_{\mathsf{deg}_x(n)}]$ où les $\overline{n}_i$ et sont éléments de $\mathbf{fix}_x^0(n)$. Alors on conclut par la propriété 5, car $[] \frown \overline{n}$ pour tout multiensemble $\overline{n}$.

Si $k = l+1$, alors $m_1 = n[\overline{n}_1/x_1, \ldots, \overline{n}_{\mathsf{deg}_x(n)}]$ et $m_2 = n[\overline{n}'_1/x_1, \ldots, \overline{n}'_{\mathsf{deg}_x(n)}]$ où les $\overline{n}_i$ et sont éléments de $\mathbf{fix}_x^{l+1}(n)$ et les $\overline{n}'_j$ sont éléments de $\mathbf{fix}_x^l(n)^!$. par hypothèse de récurrence (sur l), les éléments de $\mathbf{fix}_x^l(n)$ sont cohérents avec ceux de $\mathbf{fix}_x^{l+1}(n)$, donc on peut également conclure par la propriété 5. □

La relation de cohérence que l'on a introduite permet de comparer des termes ayant la même *forme*. En particulier, si deux termes cohérents sont des rédex, il y a une façon de les réduire tous deux de manière à obtenir deux réduits cohérents entre eux. Notons que la cohérence n'est pas préservée par la réduction en général. Par exemple, soient $\pi_1(m_1, n_1) \frown \pi_1(m_2, n_2)$. $\pi_1(m_1, n_1)$ se réduit en m_1, et $\pi_1(m_2, n_2)$ en $\pi_1(m'_2, n_2)$ si m_2 se réduit en m'_2, ce qui donne bien deux réduits non cohérents. Le lemme suivant établit que si l'on réduit deux termes cohérents « au même endroit », alors la cohérence est préservée.

Lemme 43. Soient m, m', n, n' tels que $m \frown m'$, et soient :

— $m = \langle \lambda x r \rangle v$, $m' = \langle \lambda x r' \rangle v'$, avec $(v_1, \ldots, v_k) \in \mathbf{split}^k(v)$ et $(v'_1, \ldots, v'_k) \in \mathbf{split}^k(v')$, $n = r[v_1/x_1, \ldots, v_k/x_k]$, et $n' = r'[v'_1/x_1, \ldots, v'_{k'}/x_{k'}]$.

— $m = (u = (j, v)) \cdot r$, $m' = (u' = (j', v')) \cdot r'$, $n = r$ et $n' = r'$.

— $m = \pi_i(n_1, n_2), m' = \pi_i(n'_1, n'_2), n = n_i$ et $n' = n'_i$.

— $m = \mathbf{der}([r]), m' = \mathbf{der}[r'], n = r$ et $n' = r'$.

— $m = e[u]$, $m' = e[u']$, $n = e[r]$ et $n' = e[r']$ avec les réductions $u \to_{\mathsf{rpv}} r$ et $u' \to_{\mathsf{rpv}} r'$ suivant un des cas ci-dessus.

Alors $n \frown n'$. Si, de plus, $n = n'$, alors $m = m'$.

Démonstration. La plupart de ces cas se traitent par induction structurelle sur m et m'. On traite seulement le premier point qui n'est pas immédiat, mais découle de la propriété 5.

Par construction, $r \supset r'$ et $v \supset v'$. Si l'on montre que $\{v_1, \ldots, v_k, v'_1, \ldots, v'_k\}$ forme un ensemble de termes deux-à-deux cohérents, cela suffira pour conclure en appliquant la propriété 5.

On établit ce point par induction sur v. Si $v = x$, alors v' et tous les v_i, v'_j sont également égaux à x. Si $v = [n_1, \ldots, n_k]$, alors $v' = [n'_1, \ldots, n'_l]$ avec tous les n_i et n'_j deux-à-deux cohérents. Les v_i et v'_j étant des multiensembles de termes de v (respectivement de termes de v'), ils sont bien deux-à-deux cohérents également. Pour conclure que découpage **split**() conserve bien la cohérence, on applique l'hypothèse de récurrence sur la paire et l'injection. □

Notre objectif maintenant est de montrer, à l'aide des résultats précédent, que si, pour $M \in \Lambda_{\mathsf{pv}}$, $m, m' \in \mathcal{T}_{\mathsf{pv}}(M)$, on a $m \to^{=}_{\mathsf{rpv}} n$ et $m' \to^{=}_{\mathsf{rpv}} n$, alors $m = m'$.

Lemme 44. Soient m, m', n, n' tels que $m \supset m'$, $m \to^{=}_{\mathsf{rpv}} n$ et $m' \to^{=}_{\mathsf{rpv}} n'$. Si $n = n'$ alors $m = m'$.

Démonstration. La preuve est par cas sur la réduction. On traite le cas réflexif à part, et appliquons le lemme 43 pour la réduction stricte.

- Si $m = n$, alors $m = m' = n = n'$, car un terme n'est jamais cohérent avec l'un de ses réduits stricts, par construction, donc on ne peut pas avoir $m \supset m'$, $m' \to_{\mathsf{rpv}} n$ et $m = n$, car sinon l'on aurait $m' \supset n$ également.
- Sinon, m, m', n, n' respectent l'une des hypothèses du lemme 43, et on conclut.

□

En combinant les résultats précédents, on obtient le lemme suivant, qui correspond à l'objectif de cette section sur la finitude de l'antiréduction :

Lemme 45 (Finitude de l'antiréduction). Soit $n \in \Delta_{\mathsf{pv}}$, et M un terme de Λ_{pv}. $\mathbf{card}\{m \in \Delta_{\mathsf{pv}}; m \in \mathcal{T}_{\mathsf{pv}}(M), m \to^{=}_{\mathsf{rpv}} n\} \leq 1$.

Démonstration. Par le lemme 42, $\mathcal{T}_{\mathsf{pv}}(M)$ est une clique, et par le lemme 44, si $m, m' \in \mathcal{T}_{\mathsf{pv}}(M)$ se réduisent en n, $m = m'$, donc n ne peut en effet pas avoir plus d'un antiréduit dans $\mathcal{T}_{\mathsf{pv}}(M)$. □

3.4.4 Développement de Taylor complet

Dans cette section, nous considérons des combinaisons linéaires infinies de termes à ressources, c'est-à-dire des éléments de $\mathbf{S}^{\Delta}_{\mathsf{pv}}$ (pour un semi-anneau $\mathbf{S}$ avec fractions, voir section 1.4).

3.4.4.1 Réductions et découpage quantitatifs

Avant de définir le développement de Taylor quantitatif, il nous faut adapter la construction **split**, pour qu'elle soit compatible avec les combinaisons de termes et leurs coefficients.

Définition 35 (Découpage quantitatif $\mathbf{split}_+$)**.**
On définit le découpage des valeurs $\mathbf{split}_+$, qui étend la construction précédente **split** (qui était ensembliste) à des séries de termes avec coefficients. On pourra d'ailleurs vérifier que pour tout terme m et tout entier k, $\mathbf{split}^k(m) =$

$|\mathbf{split}^k_+(m)|$, c'est-à-dire que la version quantitative que nous introduisons correspond bien à la version qualitative des sections précédentes, du point de vue des termes générés par le découpage.

On pose alors pour $\alpha \in \{1,2\} \cup \mathcal{V}$ $\mathbf{split}^k_+(\alpha) = (\alpha, \ldots, \alpha)_k$ Ensuite le découpage des multiensembles, celui qui génère des coefficients, est défini comme suit :

$$\mathbf{split}^k_+(\overline{m}) = \sum_{\substack{(\overline{m}_1,\ldots,\overline{m}_k)\in\Delta^k_{\mathsf{pv}} \\ \overline{m}_1+\ldots+\overline{m}_k=\overline{m}}} \frac{|\overline{m}|!}{|\overline{m}_1|!\ldots|\overline{m}_k|!} \cdot (\overline{m}_1,\ldots,\overline{m}_k)$$

Et enfin, on définit $\mathbf{split}^k_+(m_1, m_2)$ comme précédemment, mais en intégrant les coefficients obtenus inductivement. Soient $\overrightarrow{m}_i = (m_{i,1}, \ldots, m_{i,k})$. :

$$\sum_{\substack{(m_{1,1},\ldots,m_{1,k}) \\ \in|\mathbf{split}^k_+(m_1)|}} \sum_{\substack{(m_{2,1},\ldots,m_{2,k}) \\ \in|\mathbf{split}^k_+(m_2)|}} \left(\mathbf{split}^k_+(m_1)\right)_{\overrightarrow{m}_1} \left(\mathbf{split}^k_+(m_2)\right)_{\overrightarrow{m}_2} \cdot ((m_{1,1}, m_{2,1}), \ldots, (m_{1,k}, m_{2,k}))$$

Remarque 6. Pour tout terme $m \in \Delta_{\mathsf{pv}}$, $\mathbf{split}_+(m)$ est une somme finie. En effet, il suffit de constater, pour un multiensemble $\overline{m}$ et pour $k \in \mathbf{N}$, qu'il n'existe qu'un nombre fini de k-uplets $(\overline{m}_1, \ldots, \overline{m}_k)$ tels que $\overline{m}_1 + \ldots + \overline{m}_k = \overline{m}$, et que cette propriété de finitude est conservée par la construction de $\mathbf{split}_+$ pour la paire.

Nous devons également adapter la réduction entre termes à ressources pour qu'elle puisse prendre en compte le découpage quantitatif $\mathbf{split}_+$. Nous définissons $\to_{\mathsf{rpv}^+} \subset \Delta_{\mathsf{pv}} \times \mathbf{N}^{\Delta_{\mathsf{pv}}}$, qui réduit un terme à ressources vers une somme finie de termes à ressources.

Définition 36 (Réduction quantitative $\to_{\mathsf{rpv}^+}$ pour sommes finies de termes à ressources)**.**
Soit $m \in \Delta_{\mathsf{pv}}$ tel que $\mathsf{deg}_x(m) = k$.

$$\langle \lambda x m \rangle v \to_{\mathsf{rpv}^+} \sum_{\substack{(v_1,\ldots,v_k) \\ \in\Delta^k_{\mathsf{pv}}}} \left(\mathbf{split}^k_+(v)\right)_{(v_1,\ldots,v_k)} m[v_1/x_1, \ldots, v_k/x_k]$$

Les autres règles de réduction sont inchangées par rapport à la définition 32.

Remarque 7. Si $m \to_{\mathsf{rpv}^+} \sum_{i=1}^l a_i \cdot n_i$, alors pour tout $i \in \{1, \ldots, l\}$, $m \to_{\mathsf{rpv}} n_i$. En effet, si l'on oublie les coefficients, les termes obtenus sont les mêmes que dans la version ensembliste de ces définitions. En particulier, on vérifie bien $\mathbf{split}^k(m) = |\mathbf{split}^k_+(m)|$.

Définition 37 (Réduction entre séries infinies)**.**
On définit une réduction $\Rightarrow \subseteq \mathbf{S}^{\Delta_{\mathsf{pv}}} \times \mathbf{S}^{\Delta_{\mathsf{pv}}}$. Pour une famille de termes à ressources $(m_i)_{i\in I}$ et une famille de sommes finies de termes à ressources $(\nu_i)_{i\in I}$ telles que pour tout $i \in I$, et pour tout $n \in |\nu_i|$, l'ensemble $\{j \in I \mid m_j \to^=_{\mathsf{rpv}} n\}$ soit fini.

Dans ce cas, on pose $\sum_{i\in I} a_i \cdot m_i \Rightarrow \sum_{i\in I} a_i \cdot \nu_i$ dès que $m_i \to^=_{\mathsf{rpv}^+} \nu_i$ pour tout $i \in I$.

3.4.4.2 Développement et coefficients

Tous les constructeurs de Δ_{pv} sont linéaires, dans le sens où l'on peut écrire, par exemple, $\lambda x \left(\sum_{i\in I} a_i \cdot m_i\right) = \sum_{i\in I} a_i \cdot \lambda x m_i$, ou $\left(\sum_{i\in I} a_i \cdot m_i, \sum_{j\in I} a_j \cdot m_i\right) = \sum_{i\in I}\sum_{j\in I} a_j a_i \cdot (m_i, m_j)$. Cela nous permet de donner la définition du développement de Taylor complet, avec coefficients, comme suit :

Définition 38 (Développemment de Taylor complet)**.**
On définit le développement de Taylor complet comme une fonction $()^* : \Lambda_{\mathsf{pv}} \to \mathbf{S}^{\Delta_{\mathsf{pv}}}$.

- $x^* = x$.
- $(\lambda x M)^* = \lambda x M^*$
- $(\langle M\rangle N)^* = \langle M^*\rangle N^*$
- $((M, N))^* = (M^*, N^*)$
- $(\iota_i(M))^* = (i, M^*)$
- $(\pi_i(M))^* = \pi_i(M^*)$
- $\mathbf{case}(M, x_1 \cdot N_1, x_2 \cdot N_2)^* =$

$$\sum_{i\in\{1,2\}} \sum_{r\in\Delta_{\mathsf{pv}}} ((M^*) = (i, r)) \cdot (N_i[M/x_i])^*$$

  On rappelle que si M est égal à la i-ème injection, le terme ci-dessus se réduit dans Λ_{pv} en $N_i[M/x_i]$.
- $(M^!)^* = \sum_{k\in\mathbf{N}} \frac{1}{k!}[M^*, \ldots, M^*]_k$
- $(\mathbf{der}(M))^* = \mathbf{der}(M^*)$

Le développement de Taylor du point fixe est défini à part, et inductivement. On donne une série $\mathbf{fix}_x(M)^{*k}$ pour tout $k \in \mathbf{N}$, qui correspond à k dépliages de M dans lui même par x. C'est une version quantitative des ensembles $\mathbf{fix}^k_x(m)$ de la définition 33. On pose $(\mathbf{fix}_x(M))^{*0} = (M[[]/x])^*$, et :

$$(\mathbf{fix}_x(M))^{*k+1} = \sum_{m\in\mathcal{T}_{\mathsf{pv}}(M)} \sum_{\substack{\overline{m}_1,\ldots,\overline{m}_{\mathsf{deg}_x(m)} \\ \in((\mathbf{fix}^k_x(M))^!)^{\mathsf{deg}_x(m)}}} (M^*)_m \prod_{i=1}^{\mathsf{deg}_x(m)} ((\mathbf{fix}_x(M))^{*k})^!_{\overline{m}_i} \cdot m[\overline{m}_1/x_1, \ldots, \overline{m}_{\mathsf{deg}_x(m)}/x_{\mathsf{deg}_x(m)}]$$

et on pose $(\mathbf{fix}_x(M))^* = \sum_{k\in\mathbf{N}} (\mathbf{fix}_x(M))^{*k}$.

Remarque 8. Pour tout terme $M \in \Lambda_{\mathsf{pv}}$, $\mathcal{T}_{\mathsf{pv}}(M) = |M^*|$.

Lemme 46. Soit $M \in \Lambda_{\mathsf{pv}}$ avec $M^* = \sum_{i\in I} a_i \cdot m_i$ et $\varphi = \sum_{i\in I} a_i \cdot \nu_i$ tels que $m_i \to^{=}_{\mathsf{rpv}^+} \nu_i$ pour tout $i \in I$. Alors, pour tout $i \in I$, et pour tout $n \in |\nu_i|$, n a un coefficient fini dans φ. C'est-à-dire que la somme de tous les a_j tels que $n \in |\nu_j|$ est toujours finie.

En d'autres termes, la réduction $\Rightarrow$ est toujours définie sur le développement de Taylor.

Démonstration. C'est une conséquence immédiate du lemme 45, car M^* est une combinaison de termes à ressources à coefficients finis. □

Lemme 47. Soit $m \in \Delta_{\mathsf{pv}}$, avec $\mathsf{deg}_x(m) = k$, et V une valeur de Λ_{pv}.

$$\sum_{v \in \Delta_{\mathsf{pv}}} \sum_{\substack{(v_1,\ldots,v_k) \\ \in \Delta^k_{\mathsf{pv}}}} (V^*)_v \left(\mathbf{split}^k_+(v)\right)_{(v_1,\ldots,v_k)} \cdot m[v_1/x_1,\ldots,v_k/x_k]$$

$$= \sum_{\substack{(v_1,\ldots,v_k) \\ \in \Delta^k_{\mathsf{pv}}}} \prod_{i=1}^{k} (V^*)_{v_i} \cdot m[v_1/x_1,\ldots,v_k/x_k]$$

Le lecteur attentif ne manquera pas de remarquer dans la preuve qui suit des similitudes avec la preuve de la simulation de l'élimination de coupures dans les réseaux de preuves (lemmes 38 et 39). Même les coefficients à l'œuvre sont à peu près les mêmes, ce qui n'est pas surprenant (malgré le fait que ces égalités apparaissent pour des raisons différentes), car les quantités apparaissant dans les sémantiques sous-jacentes sont les mêmes.

Démonstration. La preuve est par induction sur V.

— Si V est une variable, alors tous les coefficients $(V^*)_{v_i}$ sont égaux à 1, et le résultat est trivial.

— Si $V = N^!$, alors on veut montrer le résultat suivant, pour tout $k \in \mathbf{N}$:

$$\sum_{\overline{n} \in \Delta_{\mathsf{pv}}} \sum_{\substack{(\overline{n}_1,\ldots,\overline{n}_k) \\ \in \Delta^k_{\mathsf{pv}}}} (N^{!*})_{\overline{n}} \left(\mathbf{split}^k_+(\overline{n})\right)_{(\overline{n}_1,\ldots,\overline{n}_k)} \cdot m[\overline{n}_1/x_1,\ldots,\overline{n}_k/x_k]$$

$$= \sum_{\substack{(\overline{n}_1,\ldots,\overline{n}_k) \\ \in \Delta^k_{\mathsf{pv}}}} \prod_{i=1}^{k} (N^{!*})_{\overline{n}_i} \cdot m[\overline{n}_1/x_1,\ldots,\overline{n}_k/x_k]$$

Où pour tout $i \leq k$, $\overline{n}_i = [n_{i,1},\ldots,n_{i,|\overline{n}_i|}]$.

En effet, rappelons que le développement de Taylor de $N^!$, qui s'écrit $\sum_{k \in \mathbf{N}} \frac{1}{k!}[N^*,\ldots,N^*]_k$, peut également s'écrire de la façon suivante (en se rappelant que les définitions du développement passent par des constructions linéaires) :

$$\sum_{k \in \mathbf{N}} \frac{1}{k!} \sum_{\substack{[n_1,\ldots,n_k] \\ \in |N^*|^!}} \prod_{i=1}^{k} (N^*)_{n_i} \cdot [n_1,\ldots,n_k]$$

Remarquons dans un premier temps que pour tout k-uplet $(\overline{n}_1,\ldots,\overline{n}_k) \in \Delta^k_{\mathsf{pv}}$, et pour tout $\overline{n} \in \Delta_{\mathsf{pv}}$, soit $\sum_{i=1}^k \overline{n}_i \neq \overline{n}$, et dans ce cas,

$$\left(\mathbf{split}^k_+(\overline{n})\right)_{(\overline{n}_1,\ldots,\overline{n}_k)} = 0$$

soit $\sum_{i=1}^k \overline{n}_i = \overline{n}$, et alors

$$\left(\mathbf{split}^k_+(\overline{n})\right)_{(\overline{n}_1,\ldots,\overline{n}_k)} = \frac{|\overline{n}|!}{|\overline{n}_1|!\ldots|\overline{n}_k|!}$$

Cela provient de la définition 35. De plus, pour tout $(\overline{n}_1, \ldots, \overline{n}_k) \in \Delta_{\mathsf{pv}}^k$, il existe un unique $\overline{n} \in \Delta$ tel que $(\overline{n}_1, \ldots, \overline{n}_k) \in \mathbf{split}^k(\overline{n})$. Donc, en utilisant également la définition de $N^{!*}$, on peut poser pour tout $\overline{n} \in \mathcal{T}_{\mathsf{pv}}(N^!)$, et pour tout $(\overline{n}_1, \ldots, \overline{n}_k) \in \mathbf{split}^k(\overline{n})$:

$$\begin{aligned}
&\left(N^{!*}\right)_{\overline{n}} \left(\mathbf{split}_+^k(\overline{n})\right)_{(\overline{n}_1,\ldots,\overline{n}_k)} \\
&= \frac{1}{|\overline{n}|!} \prod_{i=1}^{|\overline{n}|} (N^*)_{n_i} \left(\mathbf{split}_+^k(\overline{n})\right)_{(\overline{n}_1,\ldots,\overline{n}_k)} \\
&= \frac{1}{|\overline{n}|!} \prod_{i=1}^{|\overline{n}|} (N^*)_{n_i} \frac{|\overline{n}|!}{|\overline{n}_1|!\ldots|\overline{n}_k|!} \\
&= \prod_{i=1}^{|\overline{n}|} (N^*)_{n_i} \frac{1}{|\overline{n}_1|!\ldots|\overline{n}_k|!}
\end{aligned}$$

Par ailleurs, pour tout $(\overline{n}_1, \ldots, \overline{n}_k) \in \Delta^k$, on a :

$$\prod_{i=1}^{k} (N^{!*})_{\overline{n}_i} = \prod_{i=1}^{k} \left(\prod_{j=1}^{|\overline{n}_i|} (N^*)_{n_{i,j}} \right) \frac{1}{|\overline{n}_1|!\ldots|\overline{n}_k|!}$$

Où pour tout i, $\overline{n}_i = [n_{i,1}, \ldots, n_{i,|\overline{n}_i|}]$.
Il reste donc à établir pour conclure ce cas que $\sum_{(\overline{n}_1,\ldots,\overline{n}_k)\in\Delta_{\mathsf{pv}}^k} (\overline{n}_1, \ldots, \overline{n}_k)$ est égal à $\sum_{\overline{n}\in\Delta_{\mathsf{pv}}} \sum_{(\overline{n}_1,\ldots,\overline{n}_k)\in\mathbf{split}^k(\overline{n})} (\overline{n}_1, \ldots, \overline{n}_k)$, ce qui est une conséquence directe du fait que pour tout k-uplet $(\overline{n}_1, \ldots, \overline{n}_k)$, il existe un unique $\overline{n}$ tel que $(\overline{n}_1, \ldots, \overline{n}_k) \in \mathbf{split}^k(\overline{n})$, à savoir $\overline{n} = \sum_{i=1}^k \overline{n}_i$.

— Si $V = (V_1, V_2)$, alors on veut établir :

$$\begin{aligned}
&\sum_{\substack{(v_1,v_2)\\ \in\Delta_{\mathsf{pv}}}} \sum_{\substack{(u_1,\ldots,u_k)\\ \in(\Delta_{\mathsf{pv}}^2)^k}} (V_1, V_2)^*_{(v_1,v_2)} \left(\mathbf{split}_+^k((v_1, v_2))\right)_{(u_1,\ldots,u_k)} \\
&\qquad\qquad \cdot m[u_1/x_1, \ldots, u_k/x_k] \\
&= \sum_{\substack{(u_1,\ldots,u_k)\\ \in\mathcal{T}_{\mathsf{pv}}((V_1,V_2))^k}} \prod_{i=1}^{k} (V_1^*)_{v_{1,i}} \prod_{j=1}^{k} (V_2^*)_{v_{2,j}} \cdot m[u_1/x_1, \ldots, u_k/x_k]
\end{aligned}$$

Où $(u_1, \ldots, u_k) = ((v_{1,1}, v_{2,1}), \ldots, (v_{1,k}, v_{2,k}))$ où chaque $(v_{i,1}, \ldots, v_{i,k})$ apparaît dans $\mathbf{split}^k(v_i)$. Par hypothèse de récurrence, on a pour $i \in \{1, 2\}$:

$$\begin{aligned}
&\sum_{v_i\in\mathcal{T}_{\mathsf{pv}}(V_i)} \sum_{\substack{(v_{i,1},\ldots,v_{i,k})\\ \in\Delta_{\mathsf{pv}}^k}} (V_i^*)_{v_i} \left(\mathbf{split}_+^k(v_i)\right)_{(v_{i,1},\ldots,v_{i,k})} \cdot m[v_{i,1}/x_1, \ldots, v_{i,k}/x_k] \\
&\qquad = \sum_{\substack{(v_{i,1},\ldots,v_{i,k})\\ \in\Delta_{\mathsf{pv}}^k}} \prod_{j=1}^{k} (V_i^*)_{v_{i,j}} \cdot m[v_{i,1}/x_1, \ldots, v_{i,k}/x_k]
\end{aligned}$$

Ce qui nous permet de conclure ce cas car $((V_1, V_2)^*)_{(v_{1,i}, v_{2,j})} = (V_1^*)_{v_{1,i}} \times (V_2^*)_{v_{2,j}}$, et $\left(\mathbf{split}^k_+((v_1, v_2))\right)_{(u_1,\dots,u_k)} = \prod_{i=1}^2 \left(\mathbf{split}^k_+(v_i)\right)_{(v_{i,1},\dots,v_{i,k})}$.

— Si $V = \iota_i(V')$, on raisonne de la même manière en utilisant l'hypothèse de récurrence.

□

Propriété 6. $(M[N/x])^* =$

$$\sum_{m \in \mathcal{T}_{\mathsf{pv}}(M)} \sum_{(n_1,\dots,n_k) \in \mathcal{T}_{\mathsf{pv}}(N)^k} (M^*)_m \prod_{i=1}^{k} (N^*)_{n_i} \cdot m[n_1/x_1, \dots, n_k/x_k]$$

où $k = \mathsf{deg}_x(m)$.

Démonstration. Simple induction sur M. □

On peut enfin donner le résultat principal de cette section : le théorème 9 établit la simulation de la sémantique opérationnelle de Λ_{pv} dans le développement de Taylor complet.

Théorème 9. Soient $M, M' \in \Lambda_{\mathsf{pv}}$. Si $M \to_{\mathsf{pv}} M'$, alors $M^* \Rightarrow M'^*$.

Démonstration. Nous utilisons la proposition 4, et vérifions qu'elle s'étend au développement quantitatif, en remettant les coefficients à leur place.

— Si $M = \langle \lambda x N \rangle V$ et $M' = N[V/x]$, alors $M^* =$

$$\sum_{n \in \mathcal{T}_{\mathsf{pv}}(N)} \sum_{v \in \mathcal{T}_{\mathsf{pv}}(V)} (N^*)_n (V^*)_v \cdot \langle \lambda x n \rangle v$$

$$\Rightarrow \sum_{n \in \mathcal{T}_{\mathsf{pv}}(N)} \sum_{v \in \Delta_{\mathsf{pv}}} \sum_{\substack{(v_1,\dots,v_k) \\ \in \Delta^k_{\mathsf{pv}}}} (N^*)_n (V^*)_v \left(\mathbf{split}^k_+(v)\right)_{(v_1,\dots,v_k)} \cdot n[v_1/x_1, \dots, v_k/x_k]$$

$$= \sum_{n \in \mathcal{T}_{\mathsf{pv}}(N)} \sum_{(v_1,\dots,v_k) \in \Delta^k_{\mathsf{pv}}} (N^*)_n \prod_{i=1}^{k} (V^*)_{v_i} \cdot n[v_1/x_1, \dots, v_k/x_k]$$

La dernière égalité est obtenue par le lemme 47, et correspond à $N[V/x]^*$ par la propriété 6.

— Si $M = \mathbf{case}((\iota_i(V), x_1 \cdot M_1, x_2 \cdot M_2))$ et $M' = M_i[V/x_i]$, alors $M^* =$

$$\sum_{j \in \{1,2\}} \sum_{r \in \Delta_{\mathsf{pv}}} ((i, V)^* = (j, r)) \cdot N_j^*[V^*/x_{j,1}, \dots, V^*/x_{j,k}] \Rightarrow N_i^*[V^*/x_{i,1}, \dots, V^*/x_{i,k}]$$

Qui est égal à $(N[V/x])^*$ par la propriété 6.

— Si $M = \mathbf{der}(N^!)$ et $M' = N$, alors on vérifie immédiatement $(\mathbf{der}(N^!))^* = \mathbf{der}((N^!)^*) = \mathbf{der}((N^*)^!) = N^*$, car $\mathbf{der}([n_1, \dots, n_k]) \to_{\mathsf{rpv}} 0$ si $k \neq 1$.

— Si $M = \mathbf{fix}_x(N)$, alors, $M^* = (M[(\mathbf{fix}_x M)^!/x])^*$. La propriété 6 et un examen de la définition du développement de Taylor du point fixe sont suffisants pour conclure ce cas.

— Les règles de projection sont obtenues par des applications immédiates des définitions.

□

3.5 Conclusions

Nous avons présenté un calcul à ressources adéquat pour représenter la dynamique de l'Appel-Par-Pousse-Valeur, de façon à ce que les propriétés de plongement des stratégies d'évaluation soient visibles au niveau de la consommation des ressources. Nous avons ensuite proposé la construction du développement de Taylor dans cette syntaxe, et enfin démontré que la sémantique opérationnelle de Λ_{pv} pouvait y être simulée.

Deux perspectives apparaissent assez naturellement dans la continuité de ce travail et de ces résultats. La première consisterait à déterminer si ces résultats pourraient être mis en relation avec ceux du chapitre 2. En effet, Λ_{pv} comme **MELL** permettent l'encodage de stratégies de réduction, et leurs sémantiques entretiennent des liens forts. Une direction de recherche serait de développer des réseaux de preuve dans lesquels l'Appel-Par-Pousse-Valeur pourrait être plongé, et d'unifier les résultats de simulation. Si cette recherche semble s'inscrire dans le prolongement de nos travaux, notons que certaines difficultés apparaissent déjà comme non triviales. En particulier, la gestion de la duplication des types positifs, que nous avons ici traitée avec l'opérateur de découpage **split**, semble délicate à importer dans la dynamique des réseaux de **MELL**. Toutefois, les réseaux polarisés de Laurent [Lau02] par exemple, présentent une dynamique similaire autour des constructions d'arbres positifs de formules, ce qui laisse penser qu'ils seraient sans doute un bon candidat pour une étude de la sorte.

La deuxième perspective serait de traiter une stratégie d'exécution que nous n'avons pas encore considérée : l'Appel-Par-Nécessité (*Call-By-Need*). En effet, cette stratégie est étudiée pour son efficacité, dont il est dit qu'elle hérite des vertus de l'Appel-Par-Nom et de l'Appel-Par-Valeur, mais pas de leurs vices (*vices* et *vertus* sont à comprendre au regard de l'optimalité du calcul, en termes de nombre d'étapes de réduction), et permet ainsi de gérer la duplication et la suppression d'arguments de façon optimale. Donner un développement de Taylor à l'Appel-Par-Nécessité est possible, nous en avons proposé un dans un travail antérieur [Cho19], mais une difficulté subsiste : la dynamique qui fait la spécificité de cette stratégie semble ne pas pouvoir être capturée par la gestion des exponentielles, ou des multiensembles. En effet, cette dynamique paraît trop sensible à la structure des termes (la position des variables notamment) pour pouvoir être déterminée par les mécanismes de duplicabilité venant de la Logique Linéaire. La réduction doit alors être contrainte par d'autres moyens, comme une définition fine des contextes d'évaluation, ou encore par un système de types avec intersection (voir par exemple les récents travaux de Accatoli, Guerrieri et Leberle [AGL19]).

Chapitre 4

Structures réparties

Je suis venue, j'ai vu, et j'ai fait la cuisine.

Maïté [a]

a. *C'est tout simple*, éditions Robert Lafont, 1999

4.1 Introduction

4.1.1 Systèmes asynchrones

L'algorithmique répartie, ou algorithmique distribuée, est sensible au modèle de calcul utilisé à l'implémentation. Contrairement aux algorithmes classiques qui peuvent être encodés indifféremment dans une machine de Turing ou dans le λ-calcul par exemple.

On distingue deux modèles principaux pour les algorithmes distribués asynchrones :

1. Systèmes à passage de messages
2. Systèmes à mémoire partagée

Les premiers sont généralement aptes à représenter des mécanismes de communication ou les différents processus représentent des machines distantes. Les seconds sont davantage adaptés pour des communications à petite échelle, pour des systèmes multiprocesseurs par exemple, ou le calcul est partagé entre plusieurs composantes qui écrivent leurs résultats dans un espace commun. Cette distinction n'est pas une règle générale, mais elle donne l'idée d'une différence pratique entre les modèles à l'œuvre.

Dans les deux cas, le principe est celui de la répartition du calcul, qui peut apporter des améliorations par rapport à un système centralisé, en termes d'efficacité ou d'économie. Pour autant, la sensibilité au modèle implique qu'un algorithme écrit pour l'un ne peut être utilisé dans l'autre. Des traductions, ou adaptations, peuvent être effectués, mais les algorithmes ne sont pas les mêmes.

Nous considérerons exclusivement le second modèle, celui de la mémoire partagée. La raison en est que l'objet de notre étude concerne la représentation des systèmes distribués asynchrones par des outils topologiques, et que cette représentation est adaptée aux systèmes à mémoire partagée. Plus précisément, elle est adaptée à des systèmes à mémoire partagée ou l'une des opérations consiste en une lecture globale instantanée de l'état du système (*immediate atomic snapshots protocols*, voir par exemple les travaux d'Anderson pour l'introduction du modèle [And90]), nous développerons ces aspects plus loin.

Décrivons donc en quoi consistent ces systèmes : il s'agit d'ensembles de processus, ayant la capacité d'effectuer des calculs localement (chacun peut être vu comme une machine de Turing, ou un λ-terme par exemple), et celle d'écrire une valeur dans un espace de mémoire, que nous appelons *registre*, auquel tous les processus ont accès. Le système comporte donc trois types d'opérations :

1. Les calculs locaux, avec entrées et sorties.
2. L'écriture dans un registre réservé de la mémoire globale.
3. La lecture de n'importe quel registre de la mémoire globale.

Un algorithme distribué consiste (et c'est également le cas dans le modèle par passage de messages) en une série d'opérations parmi les types listés ci-dessus, exécutée par *tous* les processus du système, indifféremment. Nous détaillerons les constructions relatives à ces algorithmes plus loin. Mais insistons sur le fait que ces algorithmes ne dépendent pas du nombre de processus qui les exécutent.

Une des propriétés de ces systèmes est que l'on considère l'éventualité selon laquelle certains processus peuvent tomber en panne à n'importe quel moment de l'exécution d'un algorithme. Cette propriété est une faiblesse à l'égard du système, qui n'est pas fiable à tous les égards ; mais elle correspond à une force importante des algorithmes qui sont implémentés. En effet, ces derniers sont acceptables si toutes les exécutions dans de tels systèmes, que nous appelons *tolérants aux pannes*, parviennent néanmoins à un résultat qui respecte la spécification de la tâche que l'algorithme doit effectuer. De tels algorithmes seront par ailleurs appelés *robustes*.

Une autre propriété, qui correspond également à une faiblesse des systèmes, mais à une exigence forte sur les algorithmes, est celle de l'asynchronie. Aucune hypothèse n'est faite sur les temps de calcul de chaque processus. En particulier, lorsqu'un processus p_i constate qu'un autre processus p_j n'a pas communiqué le résultat de son calcul à la mémoire partagée, il ne peut rien en déduire. Il ne peut pas conclure que p_j est tombé en panne, car cette absence de communication peut être due à un calcul plus lent. Il ne peut pas non plus attendre la valeur en question, car p_j peut tout de même être tombé en panne, et dans ce cas il doit continuer ses opérations sans quoi il serait bloqué et ne parviendrait jamais lui-même à achever l'exécution de l'algorithme bien qu'il soit encore en état de fonctionnement, ce qui n'est pas acceptable.

Ces deux propriétés, qui sont davantage des contraintes à la conception des algorithmes, restreignent conséquemment l'ensemble des tâches qui peuvent être effectuées dans un système de la sorte. Une tâche correspond à une description

de certaines valeurs d'entrée transmises aux processus en début d'exécution, et des valeurs de sortie souhaitées en fonction. Un algorithme *résout*, ou *est solution de* la tâche, si à l'issue de son exécution tous les processus qui ne sont pas en panne ont renvoyé une valeur de sortie respectant la spécification.

Une des tâches peut-être les plus élémentaires que l'on peut exprimer est celle du consensus. Nous développerons plus loin ses spécifications, mais l'idée générale du consensus est de demander aux processus du système de s'accorder sur une valeur commune.

Un des résultats fondamentaux de la théorie de la calculabilité répartie asynchrone est l'impossibilité de la résolution de cette tâche, la première preuve est due à Fischer, Lynch et Patterson [FLP85]. Un objet central de ce chapitre est justement le consensus, que nous étudierons d'un point de vue non plus seulement algorithmique mais topologique.

4.1.2 Algorithmes et topologie

L'intervention de la topologie dans l'algorithmique distribuée provient d'une volonté de donner une caractérisation exhaustive de la calculabilité asynchrone. Elle est due à Herlihy et Shavit en 1993 [HS93], et succède à des travaux basés sur la théorie des graphes mais proposant des caractérisations moins générales [BMZ88, CM89]. La formulation la plus générale est issue d'une reprise par Herlihy et Shavit en 1999 de leurs précédents travaux [HS99].

Herlihy et Shavit proposent une traduction des systèmes à mémoire partagée et des tâches de décision dans le langage des complexes simpliciaux. Le théorème principal qui en découle est, pour une tâche donnée, l'équivalence entre l'existence d'un algorithme distribué résolvant la tâche en question et l'existence d'une fonction simpliciale entre deux complexes particuliers.

Les applications les plus remarquables de ce résultat sont les résultats d'impossibilité qui en découle. Le consensus, une fois décrit de façon topologique, est montré comme impossible par des raisonnements intrinsèques à la topologie. L'argument repose notamment sur la préservation de la connexité par les fonctions simpliciales.

Le succès de cette approche est bien établi, et a donné lieu à des champs d'études à part entière. Nous renvoyons le lecteur à un ouvrage exhaustif sur le sujet : *Distributed Computing Through Combinatorial Topology* [HKR13].

4.1.3 Contributions : Consensus aléatoire, accord d'ensemble et bornes inférieures -

L'impossibilité du consensus est un résultat qui forme un pilier de l'algorithmique distribuée moderne. Il est parfois comparé à la NP-complétude du problème SAT en algorithmique classique (Théorème de Cook-Levin).

Le consensus est une tâche qui semble tellement élémentaire, qu'il semble impossible de produire des algorithmes distribués intéressants sans elle. En effet, si décider d'une valeur commune n'est pas possible, produire un calcul complexe et efficace semble d'autant plus délicat.

Les constructions distribuées effectives consistent parfois à établir une forme faible de consensus, ou à établir le consensus en modifiant le modèle de calcul. Une de ces modifications consiste à rajouter un type d'opérations exécutable

par les processus, à savoir la production d'une valeur aléatoire. Nous parlerons de *tirages* aléatoires, ou de *lancers de pièces*.

Le premier algorithme de consensus utilisant ainsi un comportement probabiliste du système est dû à Ben-Or en 1983 [Ben83]. Dans ce cadre de travail, certaines caractérisations des problèmes doivent être révisées. Par exemple, on ne peut plus parler de l'efficacité d'un algorithme en termes de nombre d'étapes de calcul nécessaires à la résolution de la tâche, mais il faut intégrer à cette mesure la probabilité de parvenir à un résultat.

Sur les algorithmes existant, on observe que la probabilité de ne *pas* atteindre le consensus diminue au fur et à mesure que l'exécution progresse. L'approche qui nous intéresse ici consiste à renforcer cette observation en établissant que ce comportement est inhérent au consensus probabiliste.

Nous étudions dans ce chapitre des bornes inférieures de probabilité pour le consensus aléatoire. Ces bornes consistent à établir une probabilité minimale pour *n'importe quel* algorithme de consensus aléatoire de ne pas atteindre le consensus (en un nombre de tours donnés, dont dépend ladite borne).

Nous partons d'un article de Attiya et Censor, *Lower bounds for randomized consensus under a weak adversary* [AC08], qui établit des bornes inférieures, et nous traduisons une partie de leurs arguments dans le langage des complexes simpliciaux de façon à établir un lien entre le résultat topologique d'impossibilité du consensus déterministe de Herlihy et Shavit, et les bornes inférieures pour le consensus aléatoire.

Cette transposition permet d'exhiber une borne plus simplement, et dont la construction topologique donne une compréhension nouvelle de cet argument à travers une autre présentation des systèmes distribués.

Notre résultat est néanmoins moins général que celui d'Attiya et Censor, en ce que le modèle que nous exploitons est assez particulier. Nous ne considérons qu'un modèle de calcul, celui où les processus communiquent par registres d'écriture/lecture globale instantanée (*snapshot*) en mémoire partagée. De plus, nous considérons les exécutions modulo une relation d'équivalence, qui identifie les séquences d'opérations aboutissant aux mêmes états des mémoires des processus.

Ce point est inhérent à notre démarche : dans la mesure où nos outils sont topologiques, ils ne peuvent s'appliquer qu'à un modèle qui se prête à une interprétation topologique. Pour autant, les algorithmes pour le modèle à passage de messages peuvent être transcrits dans le modèle à mémoire partagée t-résilient (voir les travaux de Attiya, Bar-Noy et Doyev pour cette équivalence [ABD95]), ce qui permet de considérer que ce modèle permet une étude assez large des algorithmes distribués.

À la fin de ce chapitre, nous étudions une autre tâche, proche du consensus : l'accord d'ensemble k (*k-set agreement*). Cette tâche consiste à relâcher la spécification du consensus pour demander aux processus de s'accorder non pas sur une valeur commune unique, mais sûr au plus k valeurs différentes (quand les entrées ne sont plus binaires, mais où chaque processus propose une valeur différente). Nous montrons comment le résultat d'impossibilité de l'accord d'ensemble k, pour $k < n$ (dans un système à n processus) permet également d'établir une borne inférieure de probabilité pour cette tâche à l'aide d'arguments topologiques uniquement, en nous écartant de la preuve d'Attiya et Censor.

4.1.4 Contenu du chapitre

Nous présentons dans un premier temps le modèle de calcul avec registres d'écriture/lecture en mémoire partagée, puis nous donnons les définitions nécessaires aux constructions topologiques que nous utiliserons.

Nous présentons ensuite le lien entre la calculabilité asynchrone et les complexes simpliciaux, en décrivant brièvement le théorème fondamental de Herlihy et Shavit et son application au consensus.

Nous présentons ensuite une extension probabiliste de cette correspondance, permettant de donner une caractérisation topologique du consensus aléatoire.

Enfin, nous traduisons les arguments de Attiya et Censor à ces constructions, de façon à obtenir une borne inférieure de probabilité du consensus aléatoire à partir d'arguments essentiellement topologiques. Nous présentons dans un premier temps le modèle de calcul avec registres d'écriture/lecture en mémoire partagée, puis nous donnons les définitions nécessaires aux constructions topologiques que nous utiliserons.

Nous présentons ensuite le lien entre la calculabilité asynchrone et les complexes simpliciaux, en décrivant brièvement le théorème fondamental de Herlihy et Shavit et son application au consensus.

Nous présentons ensuite une extension probabiliste de cette correspondance, permettant de donner une caractérisation topologique du consensus aléatoire.

Nous traduisons les arguments de Attiya et Censor à ces constructions, de façon à obtenir une borne inférieure de probabilité du consensus aléatoire à partir d'arguments essentiellement topologiques.

Enfin, nous proposons une borne inférieure pour l'accord d'ensemble k en utilisant certains des arguments précédents.

4.2 Constructions

Dans cette section, nous présentons la structure générale des systèmes distribués à partage de mémoire (nous parlerons simplement de systèmes distribués), ainsi que les outils topologiques qui permettront d'en donner une sémantique géométrique plus loin.

4.2.1 Protocoles avec partage de mémoire

4.2.1.1 Mémoire partagée et registres

Un système distribué est un modèle de calcul constitué d'un nombre quelconque d'entités que l'on appellera *processus* ou *processeurs*, qui procèdent à des actions locales en exécutant des algorithmes, à la façon d'un λ-terme, d'un automate, d'une machine de Turing,... et munies d'opérations sur une *mémoire partagée*. C'est-à-dire que la valeur d'entrée de l'algorithme pour un processus donné pourra dépendre (et ce sera généralement le cas) de la sortie écrite par les autres processus sur cet espace lors de calculs antérieurs.

Chaque élément du système possède donc une mémoire locale qui lui permet d'enregistrer les valeurs écrites sur la mémoire partagée, et un espace réservé dans la mémoire partagée, qui lui permet d'écrire les résultats de ses propres calculs de façon à les communiquer aux autres processus.

La sémantique topologique qui nous intéresse dans ce chapitre est adaptée pour une classe particulière de systèmes distribués, dans lesquels les processus ne peuvent interagir avec la mémoire qu'à travers deux opérations :

— L'écriture

— La *lecture globale immédiate* (*snapshot* en anglais)

On parle de système à registres d'écriture/lecture pour désigner le type d'objets pour lesquels ces opérations doivent être implémentées.

On peut voir la mémoire globale comme un tableau de n cases mémoires, pour n processus, où chacun possède une case réservée, dans laquelle agit l'opération d'écriture. L'opération de lecture globale consiste à enregistrer le contenu du tableau entier dans la mémoire locale du processus. De plus, l'opération d'écriture écrase ce qui était auparavant dans un registre.

4.2.1.2 Tâches et protocoles

Définition 39.
Une *tâche* est une description des états initiaux globaux possibles d'un système, accompagnée d'une description des états finaux possibles, et d'une fonction de spécification δ_t, qui à tout état initial associe un sous-ensemble des états finaux.

Une tâche correspond donc :

1. à une spécification des valeurs d'entrée possibles, le cas échéant, et à une description du contenu de la mémoire globale (en général, vide).
2. à la description du contenu de la mémoire locale des processus, qui correspond aux sorties possibles.
3. à une spécification qui détermine les sorties possibles en fonction des entrées reçues.

Dans les exemples les plus simples, comme celui de la tâche suivante, nous ne donnons pas de fonction de spécification, ce qui est équivalent à dire que pour tout état initial $\mathcal{I}$, $\delta_t(\mathcal{I})$ est égal à l'ensembles des états de sortie possibles.

La tâche donnée ci-dessous n'a pas d'intérêt algorithmique en soi, elle nous servira exclusivement à donner un exemple d'instanciation élémentaire de certaines définitions.

Un exemple de tâche que nous appelons *tâche de sortie quelconque* :

1. tous les processus partent avec une valeur entière quelconque, enregistrée dans leur registre local, la mémoire globale est vide.
2. chaque processus en état de fonctionnement retourne la valeur de l'un des processus, avec l'identifiant de ce dernier.

Définition 40.
Un *protocole* est une suite d'instructions exécutée de façon uniforme par tous les processus d'un système.

On dit qu'un protocole *résout* une tâche quand, pour l'état initial spécifié par la tâche, quand tous les processus l'exécutent, ils parviennent à l'état d'arrivée spécifié par la tâche. On ne fait pas d'hypothèse sur les processus qui tombent en panne avant d'avoir fini l'exécution.

Un exemple de protocole qui résout la tâche de sortie quelconque donnée en exemple plus haut est le suivant, en posant que la mémoire globale est initialisée

à -1 partout, et la valeur locale d'entrée de chaque processus p_i (pour $i \in \{1, \ldots, n\}$) est notée `local_i`. L'opération notée `write(v)` consiste pour p_i à écrire `v` dans le i-ème registre de la mémoire partagée :

```
write(local_i);
local_i:= snapshot(shared_memory);
for (j in {1,...,n}):
 if(local_i[j] != -1):
  local_i:= (local_i[j],j);
  exit;
```

On voit que la tâche est bien accomplie par tous les processus qui exécutent le protocole, car ils écrivent leur valeur locale dans la mémoire partagée, avant d'enregistrer cette dernière. Donc si tous les autres processus sont en panne, ou n'ont pas commencé leur exécution, on est quand même assuré qu'au moins un des registres de la mémoire globale est différent de -1, et correspond bien à l'une des valeurs d'entrée des processus. Notons que la tâche en question est également résolue par le protocole qui consiste pour un processus p_i à écrire dans sa mémoire `(local_i,i)`, et que celui donné plus haut, moins trivial, l'est à valeur d'exemple d'interaction entre mémoire locale et globale.

Nous ne considérerons que des protocoles lecture globale/écriture (*snapshot/update*), mais précisons qu'il existe d'autres types de protocoles adaptés au modèle de communication par mémoire partagée, avec des opérations différentes. La lecture globale, notamment, le *snapshot*, est déjà une opération d'assez haut niveau qui peut être implémentée à partir d'opérations plus élémentaires.

4.2.1.3 Traces d'exécution et adversaires

Les exécutions des protocoles pour ces systèmes se formalisent comme des suites de lecture et d'écriture par les processus. Si le système est composé de n processus $\{p_0, \ldots, p_{n-1}\}$, alors on considère les traces d'exécution définies de la façon suivante :

Définition 41 (Traces d'exécution)**.**
Une trace d'exécution pour un système à n processeurs est un mot fini formé sur l'alphabet $\{e_i, l_j \mid i, j \in \{0, \ldots, n-1\}\}$. e_i (respectivement l_j) est à comprendre comme l'action d'écriture effectuée par le processus d'identifiant i (respectivement, l'action de lecture globale par le processus d'identifiant j).

Les traces sont munies d'une relation d'équivalence $\cong$, qui est la plus petite congruence contenant $l_i l_j \cong l_j l_i$, $e_i e_j \cong e_j e_i$, $l_i l_i \cong l_i$ et $e_j e_j \cong e_j$. Cette relation correspond à l'identification des exécutions dans lesquelles tous les processus *voient* la même chose, de la façon suivante : Si les processus i et j lisent la mémoire globale du système chacun leur tour, ils liront les mêmes valeurs s'ils sont intervertis. De même, s'ils écrivent une valeur chacun leur tour, l'état de la mémoire globale est le même quel que soit l'ordre. Deux écritures successives par le même processus mènent au même état qu'une seule, car l'écriture dans un registre écrase la valeur précédente, et deux lectures successives sont également redondantes, car la mémoire globale qui est lue n'a pas été modifiée entre les deux opérations.

Rappelons que les traces d'exécutions sont indépendantes du protocole considéré. Deux traces que l'on compare correspondent à deux exécutions distinctes

du même protocole. C'est-à-dire qu'elles permettent d'étudier les mécanismes de mémoire et de communication de façon isolée. Elles correspondent à ce qui est appelé en algorithmique distribuée un *adversaire* : un ordonnancement des processus qui fixe l'ordre de lecture et d'écriture de l'exécution.

Un protocole exécuté avec deux adversaires différents ne donnera pas toujours le même résultat, car les communications ne seront pas les mêmes, et les informations dont disposeront chaque processus seront plus ou moins complètes. Considérons un exemple simple de protocole, où chaque processus est muni d'une valeur de départ, qu'il écrit dans son espace réservé de la mémoire globale, avant de stocker le contenu de la mémoire globale et de le retourner (sous forme de tableau ou de liste par exemple). Ce qui pourrait s'écrire comme en figure 4.1 : Supposons simplement que le système comporte deux processus,

```
write(local_i);
local_i:= snapshot(shared_memory);
exit;
```

FIGURE 4.1 – Un protocole simple

p_0 et p_1, avec comme entrées des valeurs données : `input_0`$= v_0$ et `input_1`$= v_1$. La constante $\perp$ représente l'absence d'information. Comparons l'exécution de ce protocole sur deux traces d'exécution différentes : $\tau = e_0e_1l_0l_1$ et $\sigma = e_0l_0e_1l_1$.

On considère un état initial du système où la mémoire globale contient deux registres contenant la valeur $\perp$ (les processus n'ont rien écrit), et où les mémoires locales des processus contiennent leur valeur d'entrée. Cet état initial est illustré en figure 4.2. L'exécution du protocole sur τ est illustré en figure 4.3 et son exécution sur σ en figure 4.4.

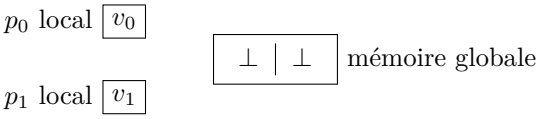


FIGURE 4.2 – État initial d'un système à deux processus, avec valeurs d'entrées fixées

4.2.1.4 Tours et protocole d'information pleine

Le protocole donné en figure 4.1, s'il est très simple, permet de modéliser la communication entre les processus de façon concise. En fait, on peut considérer que tous les protocoles pour les registres d'écriture/lecture avec mémoire partagée sont de cette forme-là, dans la mesure où toutes les opérations locales de calcul peuvent être effectuées à la fin de l'exécution. En effet, toute l'information des communications et des valeurs qui ont été accessible pendant l'exécution le restent. On vérifie facilement qu'aucune information n'est perdue, et que l'historique des communications est préservé par chaque registre. Pour cette raison, on appelle ces protocoles les *protocoles d'information pleine*.

Plus précisément, il faut étendre le protocole à une exécutions sur plusieurs tours. Du point de vue des traces d'exécution, un processus p_i a exécuté un

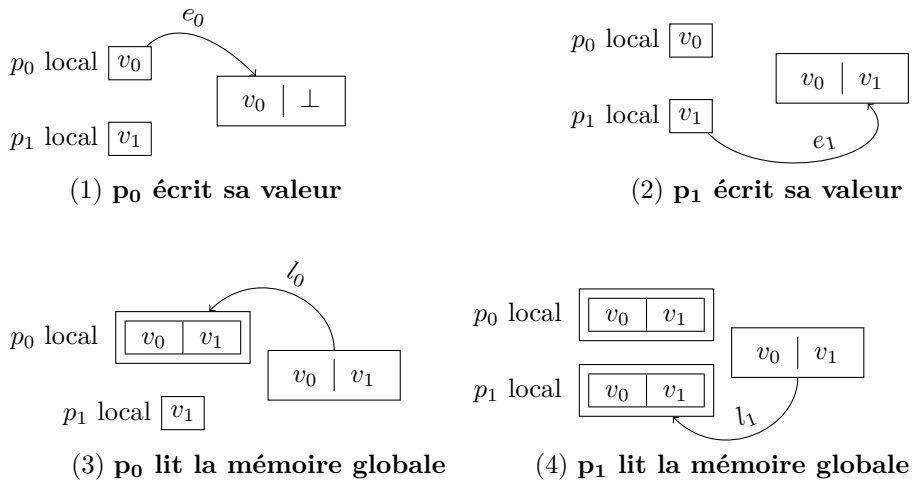


FIGURE 4.3 – Exécution du protocole sur trace/adversaire $\tau = e_0e_1l_0l_1$

protocole sur k tours si l_i apparaît k fois dans la trace d'exécution. Par exemple, la trace $e_0l_0e_1l_1e_1e_0l_0$ correspond à une exécution de deux tours par p_0 et d'un tour par p_1 (qui n'a pas fini son deuxième tour). Dans la suite de ce chapitre, nous allons considérer des exécutions de protocoles sur un nombre arbitraire de tours, pour étudier toutes les traces possibles. La trace donnée ci-dessus, par exemple, ne correspond pas à une exécution du protocole de la figure 4.1, car les processus auraient dû arrêter leurs opérations après la première opération de lecture. Pour cette raison, nous allons considérer un protocole générique, en figure 4.5, qui ne correspond pas à une tâche particulière, mais permettra l'étude générale des mécanismes de communication par partage de mémoire.

C'est-à-dire que le protocole d'information pleine suffit à décrire tous les autres protocoles, du point de vue des communications et des états de la mémoire. Les opérations locales qui caractérisent la spécificité de chaque algorithme sont omises dans nos constructions car elles ne changent rien aux différentes interactions possibles entre les processus.

Les outils topologiques que nous introduisons dans la section suivante mèneront à une modélisation des exécutions du protocole d'information pleine.

4.2.1.5 Pannes, résilience et robustesse

Soit τ une trace d'exécution pour n processus. On dit que le processus p_i *tombe en panne au tour* k si la k-ème occurrence de e_i dans τ est la dernière.

Définition 42.
On dit qu'un protocole est t-résilient, relativement à une tâche donnée si toute exécution dans laquelle au plus t processus tombent en panne résout la tâche.

On dit qu'un protocole est robuste (*wait-free* en anglais) si pour tout système à n processus, il est $n-1$-résilient. C'est-à-dire s'il résout la tâche quel que soit le nombre de processus qui tombent en panne.

Nous ne considèrerons que des protocoles robustes dans ce chapitre.

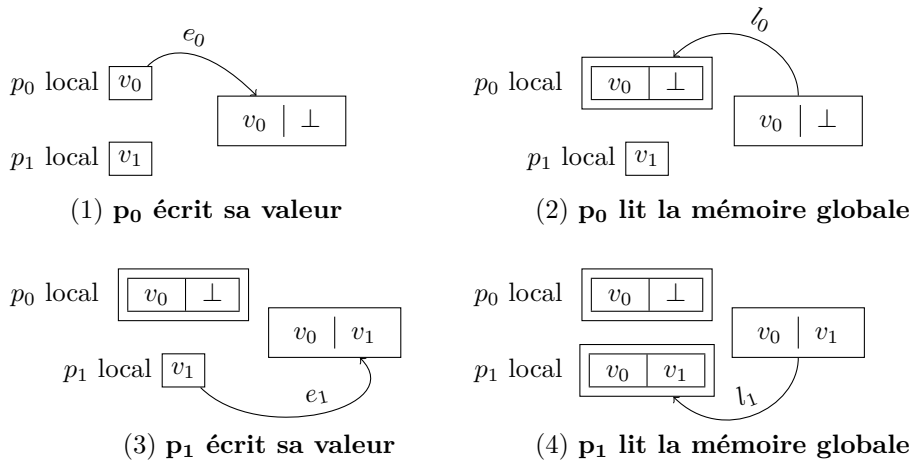


FIGURE 4.4 – Exécution du protocole sur trace/adversaire $\sigma = e_0 l_0 e_1 l_1$

```
while true:
 write(local_i);
 local_i:= snapshot(shared_memory);
```

FIGURE 4.5 – Le protocole d'information pleine

4.2.2 Outils topologiques

4.2.2.1 Simplexes, complexes simpliciaux abstraits et fonctions simpliciales

Nous donnons les constructions basiques relatives à l'approche topologique des systèmes distribués. Le lecteur pourra se rapporter à l'ouvrage de Herlihy, Kozlov et Rajsbaum [HKR13] pour approfondir le sujet.

Définition 43 (Complexes simpliciaux abstraits [HKR13])**.**
Soient S un ensemble et $\mathcal{C}$ une famille de sous-ensembles de S, on dit que $\mathcal{C}$ est un *complexe simplicial abstrait sur* S si les conditions suivantes sont satisfaites :

1. Si $X \in \mathcal{C}$ et $Y \subseteq X$, alors $Y \in \mathcal{C}$.
2. $\{v\} \in \mathcal{C}$ pour tout $v \in S$.

Les éléments de S sont appelés sommets, et les éléments de $\mathcal{C}$ sont appelés simplexes. Si $X \neq \emptyset \in \mathcal{C}$, et $\mathbf{card}(X) = k$, alors on dit que X est un simplexe de dimension $k-1$; on pourra aussi parler de $(k-1)$-simplexe. On utilisera souvent les lettres grecques miniscules pour représenter les simplexes.

On parlera simplement de complexes simpliciaux, car nous n'aurons affaire à aucune définition alternative ou variante de ces constructions.

Définition 44 (Fonctions simpliciales [HKR13])**.**
Soient $\mathcal{C}$ un complexe simplicial sur S et $\mathcal{C}'$ un complexe simplicial sur S'. Une fonction $f : S \to S'$ est dite *fonction simpliciale* si elle préserve les simplexes. C'est-à-dire, si pour tout $\{x_1, \ldots, x_k\} \in \mathcal{C}$, on a $\{f(x_1), \ldots, f(x_k)\} \in \mathcal{C}'$.

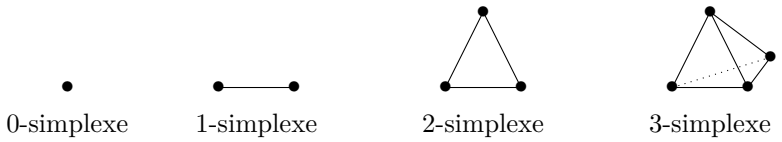


FIGURE 4.6 – représentation géométrique des simplexes de dimension 0 à 3

4.2.2.2 Complexes colorés et subdivision chromatique standard

L'opération sur les complexes simpliciaux qui nous intéresse ne se fait pas sur des complexes simpliciaux comme définis plus haut, mais sur des *complexes colorés*.

Définition 45 (Simplexe coloré standard [Koz12])**.**
Le *simplexe coloré standard* de dimension n consiste en un n-simplexe dont les sommets sont étiquetés par les éléments de $\{0, \ldots, n\}$, et les simplexes de dimension d sont étiquetés par les sous-ensembles de $\{0, \ldots, n\}$ de cardinalité $d+1$. Le simplexe coloré standard de dimension n est noté Δ^n.

Nous pouvons maintenant donner la définition de la subdivision chromatique standard, qui est la construction centrale qui nous permettra plus loin de définir le complexe de protocole. Le pendant calculatoire de cette définition correspond à la concaténation des traces d'exécution. C'est-à-dire que pour un simplexe σ qui correpond à une trace particulière, la subdivision chromatique de ce simplexe donnera un complexe, dont les simplexes correspondront à l'exécution représentée par σ, suivie d'un nouveau tour d'exécution. La subdivision peut être itérée ainsi autant de fois que le nombre de tours d'exécution que l'on souhaite considérer.

Définition 46 (Subdivision chromatique standard [HS99, Koz12])**.**
La *subdivision chromatique du complexe coloré standard de dimension n* est définie comme le complexe simplicial $\chi(\Delta^n)$:

- Les sommets de $\chi(\Delta^n)$ sont étiquetés par toutes les paires (p, V) telles que $V \subseteq \{0, \ldots, n\}$ et $p \in V$. Il y a donc $2^n(n+1)$ sommets.
- Les n-simplexes de $\chi(\Delta^n)$ sont formés par tous les ensembles de sommets $\{(0, V_0), (1, V_1), \ldots, (n, V_n)\}$ tels que
  - Pour tous $i, j \in \{0, \ldots, n\}$, soit $V_i \subseteq V_j$ soit $V_j \subseteq V_i$.
  - Pour tous $i, j \in \{0, \ldots, n\}$, si $i \in V_j$, alors $V_i \subseteq V_j$.

On appellera $\chi(\Delta^n)$ la *subdivision chromatique* de Δ^n.

Cette opération peut naturellement être répétée un nombre arbitraire de fois, car la subdivision chromatique d'un complexe coloré est un complexe coloré (ce qui est problématique avec la subdivision barycentrique, une autre opération sur les complexes, qui ne préserve pas les propriétés nécessaires, voir Herlihy et Shavit [HS99]). L'itération est définie en appliquant la subdivision telle que définie ci-dessus à tous les simplexes maximaux obtenus par la subdivision précédente, en partant non pas de $V = \{0, \ldots, n\}$, mais de $V = \{e_0, \ldots, e_n\}$ où les e_i sont les étiquettes des sommets du simplexe obtenu par subdivision.

Nous donnons en figure 4.7 une illustration de la subdivision chromatique itérée une et deux fois du 2-simplexe.

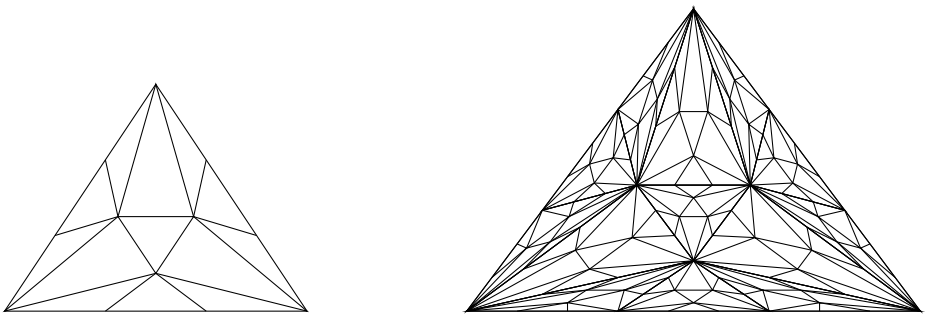


FIGURE 4.7 – $\chi(\Delta^2)$ et $\chi^2(\Delta^2)$

4.3 Géométrie de la calculabilité asynchrone

Dans cette section, nous allons faire le lien entre les protocoles et exécutions dans les systèmes distribués présentés en section 4.2.1 et les outils topologiques décrits en section 4.2.2.

4.3.1 Une interprétation topologique de la communication

4.3.1.1 Complexes et mémoire partagée

Il nous faut comprendre dans un premier temps comment les complexes simpliciaux permettent de modéliser la communication de processus par une mémoire partagée de façon satisfaisante. La correspondance se fera par trois constructions :

1. Le complexe d'entrée : un complexe simplicial qui représentera l'ensemble des états initiaux possibles d'un système pour une spécification de tâche donnée.
2. Le complexe de sortie : un complexe simplicial qui représentera les états du système correspondant à des résolutions de la tâche.
3. Le complexe de protocole : un complexe simplicial qui représentera les différentes traces d'exécution possibles (voir définition 41 et section 4.3.1.3).

Définition 47.
Le *complexe d'entrée* $\mathcal{I}_t^n$ d'une tâche donnée t, pour n processus est défini en fonction des valeurs d'entrée possibles de la tâche.

Soit $\{1, \ldots, n\}$ l'ensemble des identifiants des processus, et E_i l'ensemble des valeurs que peut recevoir le processus p_i pour traiter la tâche t.

— Les sommets de $\mathcal{I}_t^n$ sont étiquetés par tous les couples (i, x) tels que $i \in \{1, \ldots, n\}$ et $x \in E_i$.

— Les simplexes de dimension d de $\mathcal{I}_t^n$ sont étiquetés par tous les ensembles de sommets $\{(i_0, x_0), \ldots, (i_d, x_d)\}$ tels que $i_j \neq i_{j'}$ pour tous $j, j' \in \{1, \ldots, d\}$.

Définition 48.
Le *complexe de sortie* $\mathcal{O}_t^n$ d'une tâche donnée t pour n processus est définie en fonction des possibles états du système après une résolution de la tâche.

Ces états sont définis comme un ensemble S d'ensembles $\{(1, v_1), \ldots, (n, v_n)\}$, entièrement déterminé par t.

— Les sommets de $\mathcal{O}_t^n$ sont étiquetés par tous les couples (p_i, v_i) apparaissant dans les ensembles éléments de S.

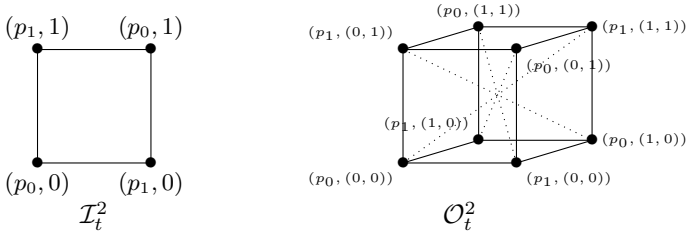
— Les simplexes de dimension d de $\mathcal{O}_t^n$ sont étiquetés par tous les ensembles $\{(p_{i_0}, v_{i_0}), \ldots, (p_{i_d}, v_{i_d})\}$ inclus dans un ensemble élément de S tels que $i_j \neq i_{j'}$ pour tous $j, j' \in \{1, \ldots, d\}$ (sinon le simplexe est de dimension inférieure).

Considérons la tâche de sortie quelconque donnée en 4.2.1.2, qui demande à tous les processus de partir avec une valeur entière et de retourner la valeur de l'un des processus avec l'identifiant de ce dernier, on peut construire les complexes d'entrée et de sortie comme suit, pour deux processus p_0 et p_1 :

Les sommets de $\mathcal{I}_t^2$ correspondent à l'ensemble $\{(p_0, k_0), (p_1, k_1) \mid k_0, k_1 \in \mathbf{N}\}$ et les simplexes sont soit des sommets, soit des arêtes qui sont de la forme $\{(p_0, k_0), (p_1, k_1)\}$.

Pour déterminer le complexe de sortie, il faut d'abord déterminer les valeurs correspondant à une résolution de la tâche. Pour que la tâche soit résolue, il faut et il suffit que les processus renvoient une valeur entière accompagnée de l'identifiant de p_0 ou de p_1. Les sommets de $\mathcal{O}_t^n$ sont donc tous les couples $(0, (i, v_i))$ et $(1, (j, v_j))$ tels que $i, j \in \{0, 1\}$ et $v_i, v_j \in \mathbf{N}$. Les simplexes sont soit des sommets, soit des arêtes de la forme $\{(0, (i, v_i)), (1, (j, v_j))\}$.

Les deux complexes sont infinis. Mais si l'on remplace $\mathbf{N}$ par $\{0, 1\}$, en demandant à ce que les entrées soient binaires, alors on peut donner une représentation finie de $\mathcal{I}_t^n$ et de $\mathcal{O}_t^n$ comme en figure 4.8. Nous ne représentons pas les étiquettes des simplexes de dimension 1, entendu qu'une arête entre deux sommets étiquetés respectivement par i et j est un simplexe d'étiquette $\{i, j\}$.



$\mathcal{I}_t^2$ $\mathcal{O}_t^2$

FIGURE 4.8 – Un complexe d'entrée et un complexe de sortie pour la tâche de sortie quelconque, avec entrées binaires et exécutée par deux processus

On peut par exemple remarquer que par construction, aucun simplexe n'est de la forme $\{(p_i, v), (p_i, v')\}$. En effet, cela correspondrait à un état initial ou final dans lequel le même p_i pourrait avoir deux valeurs distinctes, ce qui ne correspond jamais à aucun état. On peut également vérifier que toutes les configurations possibles sont représentées (les arêtes pointillées ont le même statut que les autres, et sont estompées pour la lisibilité). C'est-à-dire que dans $\mathcal{O}_t^2$ qui décrit les états de sortie acceptables pour la tâche de sortie quelconque. Chaque processus peut retourner soit sa propre valeur (0 ou 1), et dans ce cas son identifiant, ce qui correspond aux sommets $(p_i, (i, v))$ pour $v \in \{0, 1\}$, soit la valeur

de l'autre processus, avec son identifiant, ce qui correspond à $(p_i, (j, v))$, pour $i \neq j$ et $v \in \{0, 1\}$. Et ces différentes possibilités ne sont pas exclusives : toutes les configurations pour p_0 et p_1 sont compatibles, ce qui s'exprime effectivement par la présence d'arêtes entre chaque type de sommet.

4.3.1.2 Dimensions des simplexes et pannes

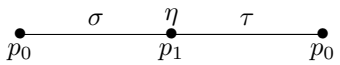
La dimension des simplexes dans les complexes d'entrée et de sortie correspond au nombre de processus qui participent. Tout simplexe de dimension 0 représente une configuration dans laquelle les processus associés aux sommets adjacents ne participent pas à l'exécution, sont tombés en panne.

Nous avons parlé des complexes d'entrée et de sortie, mais pas encore des complexes qui modélisent les exécutions. L'idée générale est qu'à une trace d'exécution τ, on fait correspondre un simplexe dans un complexe. Si le complexe comporte deux simplexes adjacents (d'intersection non vide), ils correspondent alors à deux exécutions observationellement équivalentes pour un ensemble de processus. Celui associé justement au simplexe correspondant à l'intersection en question.

Avant de définir le complexe de protocole, qui généralise cette idée, donnons quelques exemples. Considérons un système à deux processus p_0 et p_1, et deux traces d'exécution du protocole d'information pleine (figure 4.5) : $\sigma = e_0 l_0 e_1 l_1$ et $\tau = e_0 e_1 l_0 l_1$. On remarque qu'à la suite de ces exécutions, l'état mémoire de p_1 est le même (mêmes traces qu'en figures 4.3 et 4.4 auxquelles on peut se rapporter), alors que p_0 a accès à la valeur transmise par p_1 dans τ mais pas dans σ, ou il retient la valeur non définie $\perp$. Dans la compréhension topologique de ces exécutions, nous allons les considérer comme deux simplexes distincts, mais à l'intersection desquels un simplexe de dimension inférieure correspondra à l'exécution amenant à l'état de p_1 commun à σ et à τ.

Cette exécution est donnée par la trace suivante : $\eta = e_0 e_1 l_1$. Elle correspond à une panne de p_0 avant la fin de son tour. Ainsi, à l'issue de l'exécution de η, le processus p_0 « ne sait pas encore » s'il aura accès à la valeur de p_1 ou non. Si oui, alors l'exécution continue dans τ, sinon dans σ. Mais dans η, l'action de p_0 est interrompue avant une lecture de la mémoire globale.

La représentation de ces exécutions se fait alors par le complexe suivant :



On n'a volontairement pas encore donné la définition formelle du complexe, pour donner une intuition de la représentation des communications à l'œuvre ici.

4.3.1.3 Le complexe de protocole

Le complexe de protocole est une généralisation et une formalisation de l'idée que nous venons de présenter. La première définition vient de l'article fondateur de Herlihy et Shavit [HS99], nous considérons sa reformulation par Goubault, Mimram et Tasson [GMT18].

Définition 49 (Complexe de protocole)**.**
Le *complexe de protocole* est le complexe simplicial abstrait défini comme suit :

— Les arêtes sont étiquetées par les paires (i, m_i), où $i \in \{1, \ldots, n\}$ est l'identifiant d'un processus p_i, et où m_i correspond à l'état de la mémoire locale du processus p_i.

— Les simplexes de dimension d sont étiquetés par des ensembles de la forme $\{(i_0, m_{i_0}), \ldots, (i_d, m_{i_d})\}$ où $i_j \neq i_{j'}$ pour tous $j, j' \in \{1, \ldots, d\}$.

La mémoire locale correspond au contenu du registre propre à chaque processus à l'issue de l'exécution du protocole d'information pleine. Les simplexes de dimension inférieure à celle du complexe correspondent à des exécutions non terminées, où certains processus sont tombés en panne.

Le complexe de protocole pour un tour d'exécution et deux processus est représenté en figure 4.9. On omet le contenu de la valeur transmise, pour plus de généralité. Et, par exemple, l'étiquette correspondant à p_0 contenant dans sa mémoire locale la valeur transmise par p_1 sera écrite $(p_0, \langle 0, 1\rangle)$ au lieu de $(p_0,$ $\boxed{v_0 \mid v_1}$$)$. Si p_0 n'a pas accès à la valeur transmise par p_1, on écrira $(p_0, \langle 0, \bot\rangle)$. On se rappelle au passage qu'un processus à toujours accès à sa propre valeur, car le protocole d'information pleine exige d'écrire dans la mémoire partagée avant de lire cette dernière.

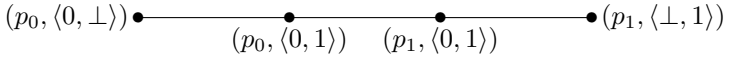


FIGURE 4.9 – Complexe de protocole, deux processus, un tour

Une propriété centrale du complexe de protocole, est que la subdivision chromatique standard en est un sous-complexe. En particulier, le complexe de la figure 4.9 est isomorphe à $\chi(\Delta^1)$, la subdivision chromatique du 1-simplexe.

Plus généralement, la subdivision itérée k fois correspond à une large classe des exécutions du protocole d'information pleine sur k tours. On illustre en figure 4.10 la subdivision chromatique du 2 simplexe, étiquetée par les états mémoire locale des trois processus p_0, p_1 et p_2.

On remarque que les bords du complexe (pour une seule itération de la subdivision) correspondent à la subdivision du 1-simplexe, autrement dit, au complexe de protocole pour deux processus illustré en figure 4.9. Et il s'agit en effet des exécutions correspondant à l'action de deux processus, où celui associé au sommet opposé au segment subdivisé en question, est tombé en panne avant toute opération.

4.3.2 Le Théorème de la calculabilité asynchrone (Herlihy-Shavit 1999)

4.3.2.1 Brève description du thèorème

Le théorème fondamental qui relie les algorithmes distribués aux structures topologiques est dû à Herlihy et Shavit [HS99]. Il détermine l'existence d'un protocole qui résout une tâche par l'existence d'une fonction simpliciale entre une subdivision chromatique du complexe d'entrée et le complexe de sortie.

Il nous faut d'abord définir à nouveau les tâches de décision, mais en termes topologiques.

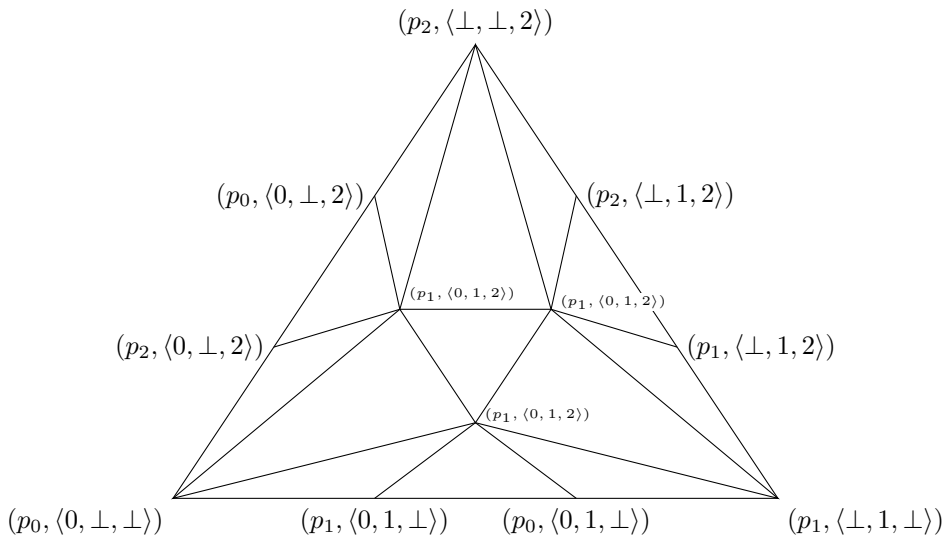


FIGURE 4.10 – Subdivision chromatique : un sous-complexe du complexe de protocole

Définition 50 (Caractérisation topologique des tâches de décision)**.**
On considère une tâche t de la définition 39. Une tâche de décision (topologique) τ correspondant à t est la donnée :

- d'un complexe d'entrée $\mathcal{O}_\tau^n$ pour tout entier $n \in \mathbf{N}$ correspondant aux états initiaux possibles pour t.
- d'un complexe de sortie $\mathcal{I}_\tau^n$ pour tout entier $n \in \mathbf{N}$ correspondant aux états mémoires finaux qui sont des résolutions de t.
- d'une relation $\delta_n^\tau \subseteq \mathcal{O}_\tau^n \times \mathcal{I}_\tau^n$ pour tout $n \in \mathbf{N}$, contenant les paires de simplexes qui correspondent à la spécification de δ_t.

Dans les exemples simples considérés jusqu'à maintenant, sans spécification, on avait $\delta_\tau = \mathcal{I}_\tau^n \times \mathcal{O}_\tau^n$.

Nous ne rentrerons pas dans les détails de ce résultat bien connu, dont l'énoncé est le suivant (on écrit $(\mathcal{I}_\tau, \mathcal{O}_\tau, \delta_t)$ et pas $(\mathcal{I}_\tau^n, \mathcal{O}_\tau^n, \delta_t^n)$, en supposant un n fixé :

Théorème 10 (Calculabilité asynchrone, Herlihy et Shavit). Une tâche de décision $\tau = (\mathcal{I}_\tau, \mathcal{O}_\tau, \delta_t)$ est résoluble par un protocole si et seulement s'il existe une subdivision chromatique $\zeta(\mathcal{I})$ et une fonction simpliciale $\mu : \zeta(\mathcal{I}) \to \mathcal{O}$ telle que pour tout simplexe σ dans $\zeta(\mathcal{I})$, $\mu(\sigma) \in \delta_\tau(\mathsf{supp}(\sigma, \mathcal{I}))$.

Où $\mathsf{supp}(\sigma, \mathcal{I})$ est le plus petit simplexe ι de $\mathcal{I}$ tel que $\sigma \in \zeta(\iota)$.

Intuitivement, le théorème établit qu'une tâche est résoluble s'il existe une fonction simpliciale entre le complexe d'entrée subdivisé, et le complexe de sortie, qui envoie les simplexes d'entrée vers des simplexes de sortie en respectant la spécification.

Notons qu'un élément clef de la preuve de ce théorème est l'étude des propriétés topologiques du complexe de protocole et de la subdivision chromatique standard du complexe d'entrée.

Nous allons maintenant développer la signification de ce résultat en évoquant sa plus célèbre application.

4.3.2.2 Une application : l'impossibilité du consensus

Le consensus distribué est une tâche très simple : les processus partent chacun avec une valeur, et doivent tous s'accorder sur une valeur commune. Nous considérons ici que les entrées sont binaires et quelconques.

On fixe n un nombre de processus, et l'on définit la tâche du consensus d'abord dans sa version algorithmique puis dans sa version topologique.

Définition 51 (Consensus distribué (binaire))**.**

- Les entrées possibles sont comprises dans $\{0, 1\}$ pour chaque processus.
- Il y a deux états de sortie valides : tous les processus (vivants) écrivent 0, ou tous les processus écrivent 1.
- La spécification se fait par la contrainte de non trivialité : la valeur décidée doit être la valeur qui a été proposée par l'un des processus. C'est-à-dire que toutes les sorties sont légales, sauf pour deux états initiaux : Si η_0 (respectivement η_1) est l'état initial où tous les processus partent avec valeur 0 (respectivement 1), alors $\delta(\eta_i)$ est l'état final où tous les processus écrivent i.

Le premier résultat d'impossibilité du consensus est dû à Fischer, Lynch et Paterson [FLP85]. Le théorème établit que le consensus est impossible à atteindre par deux processus dès lors que l'un d'entre eux est susceptible de tomber en panne.

La preuve est algorithmique, elle consiste en une analyse des états globaux du système. Elle passe par un argument par l'absurde, supposant que si un protocole résolvant le consensus existait, alors ses exécutions présenteraient un dernier état (dit *bivalent*) dans lequel les deux valeurs, 0 et 1 sont encore une sortie possible. Il est alors montré qu'il est impossible qu'une opération sur les registres puisse faire sortir d'un tel état, et faire passer le système dans un état *univalent*, ou seule une valeur est encore atteignable.

Le Théorème de la calculabilité asynchrone permet de donner une nouvelle preuve de cette impossibilité, en raisonnement uniquement sur la structure des objets topologiques impliqués. De plus, cette méthode est plus générale que la preuve évoquée plus haut, car elle peut être appliquée à une large variété de tâches de décision.

Donnons pour présenter cette application la caractérisation topologique de la tâche du consensus :

Définition 52 (Consensus distribué (binaire), version topologique)**.**

- Le complexe d'entrée du consensus binaire $\mathcal{I}_\tau^n$ est étiqueté par les ensembles $\{\{(p_0, i_0), \ldots, (p_n, i_n)\} \mid \{i_0, \ldots, i_n\} \subseteq \{0, 1\}\}$.
- Le complexe de sortie $\mathcal{O}_\tau^n$ est étiqueté par $\{\{(p_0, i), \ldots, (p_n, i)\} \mid i \in \{0, 1\}\}$.

— Pour tout simplexe σ de dimension d étiqueté par $\{(p_{j_0}, i_{j_0}), \ldots, (p_{j_d}, i_{j_d})\}$ de $\mathcal{I}_\tau^n$, si pour tout $j, j' \in \{j_0, \ldots, j_d\}$, $j = j'$, alors $\delta_\tau(\sigma) = \{\sigma\}$. Sinon, $\delta_\tau(\sigma) = \mathcal{O}_\tau^n$.

Le troisième point correspond à la condition de non-trivialité de la tâche, qui peut être reformulée en demandant que si tous les processus démarrent avec une valeur commune, cette valeur est la seule sortie possible. Et l'état final est donc identique à l'état initial : la seule valeur qui peut être décidée est la seule valeur qui a été proposée.

Une première remarque que l'on peut faire est que le complexe de sortie est déconnecté : l'intersection entre le simplexe correspondant à un consensus sur 0 et celui correspondant à un consensus sur 0 est vide. Ce qui est naturel, car si ce n'était pas le cas, il y aurait un état de sortie dans lequel certains processus décideraient 0, et d'autres décideraient 1, ce qui violerait la spécification de la tâche. En revanche, le complexe d'entrée, lui, est connexe. En effet, on peut facilement vérifier que toutes les combinaisons possibles sont reliées entre elles.

Les simplexes d'entrée et de sortie pour $n = 3$ sont représentés en figure 4.11, où l'on a mis en évidence les simplexes correspondant à un état d'entrée ou de sortie où les valeurs sont communes. On note indifféremment $p_i : i$ et (p_i, i) pour représenter un identifiant de processus et sa valeur d'entrée ou de sortie.

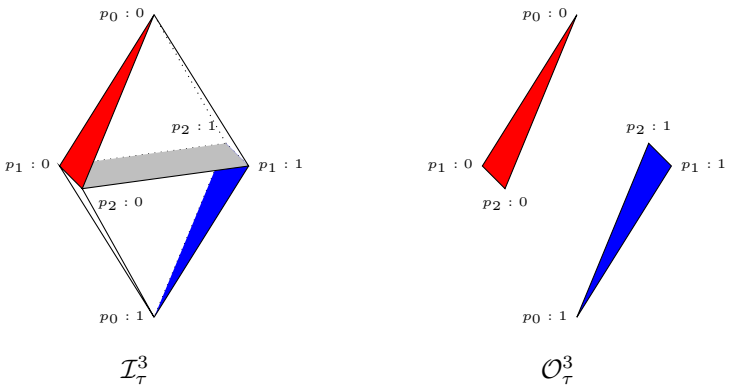


FIGURE 4.11 – Complexe d'entrée et complexe de sortie pour consensus binaire à trois processeurs

On peut alors appliquer le Théorème de la calculabilité asynchrone pour démontrer l'impossibilité du consensus. Par un argument topologique, il est établi que la subdivision chromatique préserve la connexité, et que cette même connexité est également préservée par les fonctions simpliciales. Or, la spécification impose que deux simplexes (le rouge et le bleu en figure 4.11), connectés car dans $\mathcal{I}_\tau^n$, soient envoyés sur deux complexes déconnectés (également le rouge et le bleu dans la même figure) dans $\mathcal{O}_\tau^n$. Il n'existe donc pas de fonction simpliciale qui respecte la spécification de la tâche, et donc en appliquant le théorème 10 on établit qu'il n'existe pas de protocole qui résolve le consensus.

4.4 Algorithmes probabilistes et consensus aléatoire

Pour autant, la quête du consensus ne s'arête pas là. Il existe plusieurs façon de modifier le modèle de calcul de façon à pouvoir concevoir des protocoles qui résolvent le consensus. Aspnes [Asp02] les classe parmi quatre types de démarche :

1. Randomisation du système, où sont autorisées des étapes de calcul probabilistes.
2. Synchronisation partielle, où l'on fait par exemple des hypothèses sur le temps de calcul des processus.
3. Détecteurs de panne (pour comprendre cette approche, on renvoie par exemple aux travaux d'Aguilera, Chen et Toueg [ACT98], ou à ceux de Dolev, Friedman, Keidar et Malkhi [DFKM97])
4. Renforcement des primitives, où l'on ajoute aux opérations atomiques d'écriture et de lecture, des opérations plus complètes, comme *test-and-set*. Cette approche est reliée à la hiérarchie de consensus proposée par Herlihy [Her91].

L'objet de notre étude concerne la première approche. Cela est en particulier motivé par le fait que, souhaitant approfondir les liens entre algorithmes distribués et topologie, les trois autres approches se prêtent moins à l'interprétation de la calculabilité asynchrone dans les complexes simpliciaux. Le fait d'introduire de l'aléatoire dans le modèle n'altère pas son aspect asynchrone, et ne l'enrichit pas avec des objets algorithmiques qui rendraient insuffisante l'analyse seule des communications par mémoire partagée.

Nous donnons en figure 4.12 un exemple de protocole résolvant le consensus à l'aide de l'introduction d'une opération probabiliste. Il s'agit du protocole de Chor-Israeli-Li [CIL94], dans sa version modifiée et adaptée par Aspnes [Asp02].

```
preference := input;
round := 1;
while True :
  write(preference,round);
  snapshot(shared_memory);
  maxround := max{R.round | R in shared_memory};
  If(for all R such that
                R.round≤maxround-1,R.preference = v):
    return v;
  Else
    If(exists v such that for all R where
                 R.round=maxround,R.preference = v):
      preference := v;
    If coin(1/2n):
      round := max{round+1,maxround-2};
```

FIGURE 4.12 – Un exemple de protocole aléatoire : Chor-Israeli-Li [CIL94]

On note l'utilisation de l'opération `coin(p)`, qui est ici un booléen évalué à `true` avec proabilité p, et à `false` avec probabilité $1-p$. Nous choisissons d'illustrer les protocoles aléatoires par celui-ci, car justement les jeters aléatoires de pièces sont des opérations *locales*, ce qui rend le modèle plus général que d'autres algorithmes plus efficaces mais plus complexes, comme ceux par exemple basés sur l'implémentation subsidiaire d'une *pièce partagée*, agissant comme un oracle (voir Aspnes et Herlihy [AH90]).

On pourra donc paramétrer une exécution d'un protocole probabiliste par une suite de tirages : l'exécution sera alors déterministe, une fois un tel tirage fixé. Si le protocole est exécuté par n processus sur k tours, on fixe $\overrightarrow{c} = (\overrightarrow{c}_1, \ldots, \overrightarrow{c}_n) \in (\{0,1\}^k)^n$, des suites de 0 et de 1 correspondant au résultat des tirages aléatoires de chaque processus participant au protocole.

4.4.0.1 Complexes et tirages aléatoires

Pour un système à n processus, on note $\overrightarrow{c} = (\overrightarrow{c}_1, \ldots, \overrightarrow{c}_n)$ une suite de tirages aléatoires. L'idée de notre approche topologique du consensus aléatoire est la suivante : le complexe de protocole va être paramétré par ces tirages. C'est-à-dire que, partant du complexe d'entrée $\mathcal{I}$, nous allons construire une subdivision $\chi(\mathcal{I})$ pour tout $\overrightarrow{c}$ correspondant à un tirage pour un seul tour, qui sera notée $\chi(\mathcal{I})_{\overrightarrow{c}}$. Pour tout $\overrightarrow{c}$, $\chi(\mathcal{I})_{\overrightarrow{c}}$ est isomorphe à $\chi(\mathcal{I})$, et ne diffère qu'en ce que les sommets reçoivent tous l'étiquette supplémentaire $\overrightarrow{c}$. Les simplexes de $\chi(\mathcal{I})_{\overrightarrow{c}}$ seront notés $\sigma_{\overrightarrow{c}}, \tau_{\overrightarrow{c}}, \ldots$.

Il y a donc, pour un complexe d'entrée à n processus, 2^n subdivisions considérées, et 2^{nk} k-itérations de subdivisions, de façon à ce que tous les vecteurs $\overrightarrow{c} = (\overrightarrow{c}_1, \ldots, \overrightarrow{c}_n)$ tels que les $\overrightarrow{c}_i$ sont de longueur k soient représentés.

Étant donné qu'un algorithme probabiliste auquel on fournit une suite de tirages fixée se comporte simplement comme un algorithme déterministe, on peut considérer que $\chi^k(\mathcal{I})_{\overrightarrow{c}}$ représente toutes les exécutions possibles d'un protocole, en tenant compte des choix probabilistes. Toutefois, la plupart de ces choix seront redondants pour une majorité de protocoles, car notre construction prend en compte la possibilité pour chaque processus de faire un tirage à chaque tour. Ce qui, en pratique, est très largement surestimé. Voir par exemple le protocole de Chor-Israeli-Li en figure 4.12 pour s'en convaincre. Mais cette redondance sera traitée par la fonction de décision.

Dans la mesure où nous avons construit un ensemble de complexes de protocole, correspondant chacun à un algorithme déterministe *différent*, nous pouvons utiliser les outils introduits par Herlihy et Shavit pour caractériser les fonctions de décision dans ce réglage.

Considérons les tâches qui, à l'instar du consensus, possèdent une solution dans les protocoles randomizés mais pas dans un réglage déterministe. Il n'est pas envisageable de supposer l'existence d'un $\chi^k(\mathcal{I})_{\overrightarrow{c}}$ tel qu'il existe une fonction simpliciale μ respectant les conditions du Théorème de la calculabilité asynchrone. En effet, cela reviendrait à dire que l'algorithme a une version déterministe qui résout la tâche.

4.4.0.2 Fonctions de décision partielles

Nous sommes donc conduits à considérer des fonctions de décision partielles. Une telle fonction, pour $k \in \mathbf{N}$, est une fonction simpliciale partielle prenant

pour domaine l'ensemble des complexes de protocole $\{\chi^k(\mathcal{I})_{\overrightarrow{c}} \mid \overrightarrow{c} \in (\{0,1\}^k)^n\}$ et pour codomaine le complexe de sortie $\mathcal{O}$. Cette fonction est entièrement déterminée par la définition du protocole, elle associe à toute exécution τ et à tout tirage $\overrightarrow{c}$ les valeurs de sortie de l'algorithme (prises comme un simplexe de $\mathcal{O}$) s'il résout la tâche pour ces données, et n'est pas définie sinon : on écrira $\mu(\tau_{\overrightarrow{c}}) = \perp$ le cas échéant.

Si μ est une fonction de décision pour un protocole de consensus aléatoire, alors on a une contrainte qui dirige sa caractérisation. Soient η_0 le simplexe d'entrée où tous les processus partent avec la valeur 0, et η_1 celui où ils partent avec la valeur 1 (voir la figure 4.11) : Pour tout k, pour tout $\overrightarrow{c}$, et tout $\tau_i \in \chi^k(\eta_i)_{\overrightarrow{c}}$, $\mu(\tau) = \perp$ ou $\mu(\tau) \subseteq \{(p_0, i), \ldots, (p_n, i)\}$. C'est la condition de non trivialité : quand une seule valeur est proposée, c'est la seule valeur qui peut être décidée (s'il y en a une).

4.5 Bornes pour le consensus et l'accord d'ensemble

Dans cette section, nous apportons une contribution qui établit un lien entre un résultat de Attiya et Censor au sujet du consensus aléatoire [AC08], et l'interprétation topologique des systèmes à mémoire partagée que nous avons esquissée dans la première partie de ce chapitre.

Ce résultat établit une borne inférieure pour le consensus aléatoire. C'est-à-dire qu'il donne une valeur, paramétrée par le nombre de processus du système, par le nombre d'erreurs possibles, et par le nombre de tours effectués, qui est la probabilité minimale que le système n'*atteigne pas* le consensus.

Pour un système à n processus et k tours, Attiya et Censor exhibent une constante c telle que la probabilité qu'un protocole f-résilient n'atteigne pas le consensus au bout de $k(n-f)$ étapes est d'au moins $\frac{1}{c^k}$. On rappelle qu'un protocole f-résilient est un algorithme distribué dont la terminaison est conditionnée à un nombre maximal f de processus pouvant tomber en panne.

Nous donnons rapidement une idée de la preuve avant de montrer comment nous la transposons dans le langage topologique.

Une définition essentielle est celle de *chaîne d'indistinguabilité*. Il s'agit de considérer une suite d'exécutions, $\tau_1, \ldots, \tau_n$ telles qu'entre τ_i et τ_{i+1}, il y ait toujours un ensemble de processus qui observent la même chose. C'est-à-dire que les exécutions donnent deux-à-deux indiscernables pour au moins un processus, il a accès aux mêmes informations exactement, et se comportera donc de la même façon pour chacune d'elles.

La spécification du consensus implique en particulier que, si deux exécutions ainsi indistinguibles correspondent toutes deux à une résolution du consensus, alors les valeurs décidées sont les mêmes. En effet, les processus pour lesquels ces exécutions sont les mêmes doivent bien sûr se comporter de la même façon dans chacune d'elles.

La preuve consiste alors à considérer les exécutions possibles sur k tours, et à construire une chaîne d'indistinguabilité entre une exécution τ^0 sur un état initial univalent (où τ^0 peut être définie comme une trace d'exécution accompagnée de la description des valeurs d'entrée), où tous les processus ont la même valeur d'entrée 0 , et une exécution τ^1 sur un état initial univalent, mais où les

processus ont comme valeur d'entrée 0.

Une fois cette chaîne construite, on peut attester qu'il est impossible qu'il existe un lancer de pièces $\overrightarrow{c}$ telles que toutes les exécutions de la chaîne atteignent le consensus. En effet, si τ^0 atteint le consensus, la valeur décidée doit être 0, par non trivialité, et si τ^1 l'atteint, 1 doit être décidé de même. Or, si toutes les exécutions de la chaîne correspondent à une résolution (tous les processus en état de marche ont décidé d'une valeur de sortie à l'issue de cette exécution), elles doivent toutes parvenir à la même valeur, car deux-à-deux indistinguables par au moins un processus.

Si, donc, on pose q_k la probabilité maximale qu'un protocole n'atteigne pas le consensus en k tours, on peut formuler le théorème d'Attiya et Censor comme suit :

Théorème 11 (Attiya et Censor [AC08]). Soient τ^0 et τ^1 deux exécutions d'un protocole aléatoire de consensus sur un état initial où tous les processus ont, respectivement, 0 et 1 comme valeur d'entrée. S'il existe une chaîne d'indistinguabilité de longueur m entre τ^0 et τ^1, alors la probabilité que le protocole ne termine pas après k étapes, q_k, est d'au moins $\frac{1}{m+1}$.

Idée de la preuve. Supposons par l'absurde que q_k soit inférieur à $\frac{1}{m+1}$. On a donc $q_k(m+1) < 1$, et donc la probabilité qu'il existe un lancer de pièces $\overrightarrow{c}$ tel que pour ce tirage, les $m+1$ exécutions de la chaîne d'indistinguabilité résolvent le consensus est non nulle. Or, on a vu que toutes les exécutions d'une telle chaîne ne pouvaient pas terminer sans violer les spécifications du consensus. □

4.5.1 Chaînes d'indistinguabilité

Nous donnons ici notre contribution, qui consiste essentiellement à utiliser les outils topologiques déjà présentés pour présenter le résultat d'Attiya et Censor dans ce langage. L'idée est de considérer la structure des complexes simpliciaux pour définir naturellement les chaînes d'indistinguabilité comme des suites de simplexes adjacents, qui respectent les propriétés des chaînes d'indistinguabilité telles que définies par Attiya et Censor par le principe même de la construction des complexes de protocole.

Définition 53.
Soit $\mathcal{C}$ un complexe simplicial. Une *chaîne d'indistinguabilité* de longueur m dans $\mathcal{C}$ est une suite $\sigma_0, \ldots, \sigma_m$ de simplexes de $\mathcal{C}$ telle que pour tout $i \in \{0, \ldots, m-1\}$, $\sigma_i \cap \sigma_{i+1} \neq \emptyset$. On parlera parfois simplement de chaîne.

Cette définition naturelle correspond au fait que l'intersection entre deux simplexes σ et τ dans le complexe de protocole représente une vision partielle d'un ensemble de processus pour lesquels σ et τ sont indistinguibles.

Pour construire les chaînes d'indistinguabilité, nous revenons sur les propriétés de la subdivision chromatique standard, pour les appliquer au complexe de protocole qui contient cette dernière. En figure 4.13, on illustre le complexe de protocole pour le consensus binaire, dans lequel on a exhibé une chaîne d'indistinguabilité entre deux simplexes (de dimension 0) issus des subdivisions des deux états initiaux univalents opposés.

La propriété 7 nous permettra de raisonner sur des chaînes d'indistinguabilité construites à partir de la subdivision chromatique. Elle établit que si deux

simplexes sont adjacents, alors leurs subdivisions chromatiques forment des complexes également d'intersection non vides. Cela tient à ce que la subdivision est stable pour l'intersection ($\chi(\sigma \cap \tau) = \chi(\sigma)\chi(\tau)$).

Cette propriété permettra ensuite de construire une chaîne à partir d'un nombre quelconque d'itérations de la subdivision sur un simplexe donné. C'est l'objet du lemme 48.

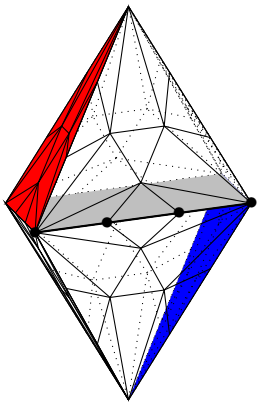


FIGURE 4.13 – Le complexe de protocole pour le consensus binaire, et une chaîne d'indistinguabilité.

Propriété 7. Soient $k \in \mathbf{N}$ et $\mathcal{C}$ un complexe simplicial.

— $\chi^k(\mathcal{C}) = \bigcup_{\sigma \in \mathcal{C}} \chi^k(\sigma)$.

— Soient $\sigma, \tau \in \mathcal{C}$. Si $\sigma \cap \tau \neq \emptyset$, alors $\chi^k(\sigma) \cap \chi^k(\tau) \neq \emptyset$.

Démonstration. Le premier point s'établit par un exament de la définition de la subdivision chromatique. Le second est une conséquence du premier, car si $\eta \in \sigma \cap \tau$, alors $\chi^k(\eta) \in \chi^k(\sigma) \cap \chi^k(\tau)$. □

On montre maintenant que la subdivision chromatique itérée k fois du 1-simplexe forme une chaîne d'indistinguabilité de longueur 3^k. Intuitivement, il s'agit de formaliser le fait que la subdivision du 1-simplexe peut se voir comme un sectionnement en trois simplexes adjacents, qui seront à leur tour subdivisés, tout en continuant à former une chaîne, ce qui explique la borne 3^k, que l'on retrouvera plus tard.

Lemme 48. Soit $k \in \mathbf{N}$ et Δ^1 le simplexe coloré standard de dimension 1. $\chi^k(\Delta^1)$ forme une chaîne d'indistinguabilité de longueur 3^k.

Démonstration. La preuve est par induction sur k. Par définition, Δ_1 est formé de deux sommets étiquetés par $\{0\}$ et $\{1\}$, et d'un simplexe maximal étiqueté par $\{0,1\}$. La subdivision chromatique de Δ^1 est donc formée de 4 sommets, étiquetés par les paires $(0,\{0\}), (0,\{0,1\}), (1,\{1\}), (1,\{0,1\})$, et des trois 1-simplexes : $\{(0,\{0\}),(1,\{0,1\})\}$, $\{(1,\{0,1\}),(0,\{0,1\})\}$, et $\{(0,\{0,1\}),(1,\{1\})\}$ (voir la définition 46). On constate immédiatement que ces simplexes forment une chaîne d'indistinguabilité (dans l'ordre donné ci-dessus), qui est bien de longueur 3.

Si $k = l+1$, alors par hypothèse de récurrence, $\chi^l(\Delta^1)$ forme une chaîne d'indistinguabilité $(\sigma_1, \ldots, \sigma_{3^l})$, où pour tout $i \in \{1, \ldots, 3^l\}$, σ_i est un simplexe de

dimension 1, qui est naturellement isomorphe à Δ^1. On peut donc répéter l'argument ci-dessus pour s'assurer que $\chi(\sigma_i)$ forme une chaîne d'indistinguabilité de longueur 3. Par la propriété 7, pour tout $i \in \{1, \ldots, 3^l\}$, $\chi(\sigma_i) \cap \chi(\sigma_{i+1}) \neq \emptyset$. Donc $(\chi(\sigma_1), \ldots, \chi(\sigma_{3^l}))$ est une suite de 3^l complexes deux-à-deux adjacents (*i.e.* d'intersection non vide). Il faut néanmoins s'assurer que les suites induites par les $\chi(\sigma_i)$ sont concaténables. C'est-à-dire que si tout $\chi(\sigma_i)$ peut s'ordonner en une suite $(\tau_{i,1}, \tau_{i,2}, \tau_{i,3})$, alors $(\tau_{i,1}, \tau_{i,2}, \tau_{i,3}, \tau_{i+1,1}, \tau_{i+1,2}, \tau_{i+1,3})$ forme également une chaîne d'indistinguabilité.

Pour cela, remarquons que deux simplexes adjacents σ, τ sont de la forme $\sigma = \{(a_1, e_1), (a_2, e_2)\}$, $\tau = \{(a_2, e_2), (a_3, e_3)\}$, où les e_i sont des étiquettes. On peut vérifier aisément que la subdivision de σ peut s'écrire comme une chaîne, dont le dernier simplexe est $\{(a_1, \{e_1, e_2\}), (a_2, \{e_2, e_2\})\}$, et que la subdivision de τ peut s'écrire comme une chaîne dont le dernier simplexe est $\{(a_2, \{e_2, e_2\}), (a_3, \{e_2, e_3\})\}$. On peut donc conclure.

C'est une façon de formaliser, par exemple, une lecture de gauche à droite du complexe illustré en figure 4.9.

□

Lemme 49. Soient $n, k \in \mathbf{N}$, et $\mathcal{I}^n$ le complexe d'entrée pour le consensus binaire à n processus.

Il existe une chaîne d'indistinguabilité de longueur 3^k reliant la subdivision du simplexe d'entrée $\eta_0 = (p_0 : 0, \ldots, p_n : 0)$ et celle du simplexe d'entrée $\eta_1 = (p_0 : 1, \ldots, p_n : 1)$.

Démonstration. Il s'agit ici d'appliquer directement le lemme 48 : en se rappelant de la définition du complexe d'entrée, on peut considérer n'importe quel couple de processus distincts (p_i, p_j) pour lesquels le simplexe à une dimension $\{p_i : 0, p_i : 1\}$ appartient au complexe d'entrée. Voir la figure 4.14 pour le cas où $n = 2$ et $k = 1$ et la figure 4.15 pour $k = 2$.

On a bien alors deux étiquettes e_i et e_j telles que $\{(p_i, e_i)\} \in \chi^k(\eta_0)$, $\{(p_j, e_j)\} \in \chi^k(\eta_1)$, et telles que l'on ait une chaîne $(\tau_1, \ldots, \tau_{3^k})$ avec $(p_i, e_i) \in \tau_1$ et $(p_j, e_j) \in \tau_{3^k}$.

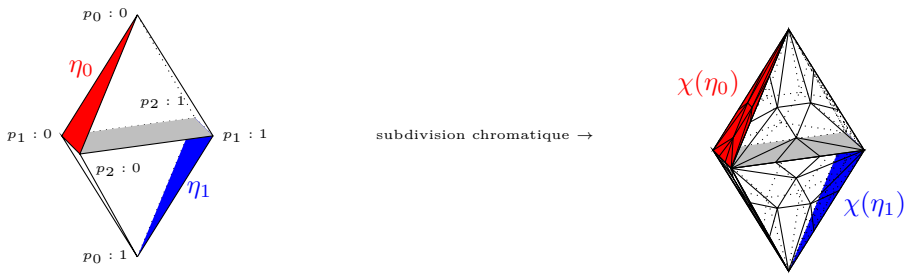


FIGURE 4.14 – Le complexe d'entrée pour le consensus binaire, et sa subdivision chromatique.

□

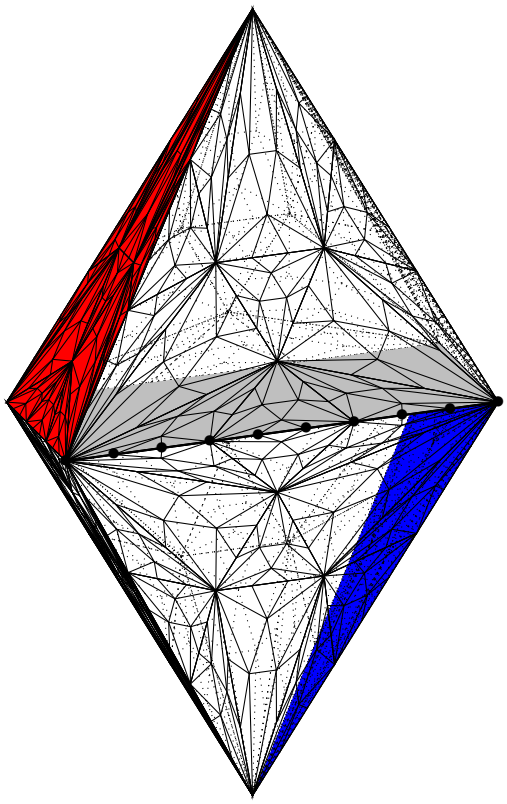


FIGURE 4.15 – Exemple pour $k = 2$: On a bien 9 simplexes de dimension 1, adjacents deux-à-deux

4.5.2 Probabilité de désaccord

Nous disposons désormais des ingrédients nécessaires pour établir le résultat principal de ce chapitre, à savoir la transposition des travaux de Attiya et Censor dans le langage des complexes simpliciaux.

Pour ce faire, revenons aux constructions dressées en section 4.4. La gestion des probabilités se fait isolément, au niveau des lancers de pièces $\overrightarrow{c}$. Pour un protocole donné, est fixée une probabilité à chacun des lancers. Si par exemple l'algorithme considéré est muni d'un tirage uniforme sur $\{0,1\}$, alors la probabilité de $\overrightarrow{c} \in (\{0,1\}^k)^n$ est de $\frac{1}{2^{kn}}$. Si, comme dans le cas du protocole de Chor, Israeli et Li, l'algorithme demande un tirage de 1 avec probabilité $\frac{1}{2n}$, la probabilité associée à chaque tirage sera adaptée en fonction. On note $p_A(\overrightarrow{c})$ la probabilité du tirage $\overrightarrow{c}$ ainsi déterminée par le protocole A.

Définition 54 (Probabilité de succès ou d'échec d'une exécution)**.**
Soient une tâche de décision et un algorithme probabiliste A qui soit correct pour cette tâche. On définit, pour toute trace d'exécution τ sur k tours, la probabilité $p_A(\tau)$ que cette exécution pour A résolve la tâche :

$$p_A(\tau) = \frac{\sum_{\overrightarrow{c} \in \{(\{0,1\}^k)^n | \mu(\tau_{\overrightarrow{c}}) \neq \perp\}} p_A(\overrightarrow{c})}{\sum_{\overrightarrow{c} \in (\{0,1\}^k)^n} p_A(\overrightarrow{c})}$$

C'est-à-dire que $p_A(\tau)$ est calculée comme la proportion des tirages $\overrightarrow{c}$ pour lesquels $\mu(\tau_{\overrightarrow{c}})$ est définie.

On définit $\overline{p}(\tau)$ la probabilité que τ n'achève pas la résolution de la tâche comme $1 - p(\tau)$.

Enfin, on définit la probabilité globale qu'un algorithme A ne termine pas au bout de k tours comme $q_A^k = \max\{\overline{p}(\tau) \mid \tau \in \chi^k(\mathcal{I})\}$.

On peut maintenant formuler et prouver le résultat principal de cette section, qui permet d'établir une borne inférieure à q_A^k pour tout algorithme A, et pour tout k en fonction de k.

L'idée de la preuve de ce théorème est la suivante : par définition, il existe dans le complexe d'entrée du consensus binaire un simplexe dans lequel tous les processus proposent la valeur 0, et un simplexe dans lequel ils décident tous 1. Ces deux simplexes univalents sont reliés par des 1-simplexes, par définition du complexe d'entrée à nouveau : ces 1-simplexes correspondent chaque fois à deux processus proposant deux valeurs distinctes. Dans le complexe de protocole pour k tours, ces 1-simplexes sont subdivisés k fois, on peut donc construire une chaîne d'indistinguabilité de longueur 3^k. Ce qui amène la conclusion de la preuve est le fait que les extrémités de cette chaîne correspondent à deux processus qui, s'ils donnent une sortie pour le consensus, en donnent deux différentes. On aboutit alors à une contradiction, car si q_A^k était inférieur à $\frac{1}{3^k}$, il existerait une probabilité non nulle que toutes les exécutions représentées dans la chaîne d'indistinguabilité atteignent le consensus pour une même suite de tirages $\overrightarrow{c}$, ce qui est impossible car les valeurs décidées devraient être les mêmes.

Théorème 12. Pour tout algorithme (robuste) de consensus aléatoire A, et pour tout $k \in \mathbf{N}$, $q_A^k \geq \frac{1}{3^k}$.

Démonstration. Supposons au contraire l'existence d'un algorithme A tel que $q_A^k < \frac{1}{3^k}$. Par le lemme 49, il existe une chaîne de longueur 3^k entre $\chi^k(\eta_0)$ et $\chi^k(\eta_1)$, où η_0 et η_1 sont les deux simplexes d'entrée univalents.

Considérons $\tau_1, \ldots, \tau_{3^k}$ les simplexes formant cette chaîne. Par définition, la probabilité qu'au moins l'un d'entre eux corresponde à une exécution qui ne termine pas est $\sum_{i=1}^{3^k} \overline{p}(\tau_i) \leq 3^k(q_A^k)$. Par hypothèse, $3^k(q_A^k) < 1$. Donc, il existe une probabilité non nulle telle que $\mu((\tau_i)_{\overrightarrow{c}}) \neq \perp$ pour tout $i \in \{1, \ldots, 3^k\}$.

Or, on peut supposer sans perte de généralité $\chi^k(\eta_0) \cap \tau_1 \neq \emptyset$ et $\chi^k(\eta_1) \cap \tau_{3^k} \neq \emptyset$, c'est-à-dire que la chaîne relie un simplexe issu de η_0 à un simplexe issu de η_1. Or, par la spécification de la tâche de consensus, si $\sigma_i \in \chi^k(\eta_i)$, alors $\mu(\sigma_i) = \perp$ ou $\mu(\sigma_i) \subseteq \{(p_0, i), \ldots, (p_{n-1}, i)\}$. En particulier, pour $\sigma_0 \in \chi^k(\eta_0) \cap \tau_1$ et $\sigma_1 \in \chi^k(\eta_1) \cap \tau_{3^k}$, $\mu(\sigma_0) \cap \mu(\sigma_1) = \emptyset$. Or, comme μ est une fonction simpliciale et que les deux seuls simplexes de sortie sont $\zeta_0 = \{(p_0, 0), \ldots, (p_{n-1}, 0)\}$ et $\zeta_1 = \{(p_0, 1), \ldots, (p_{n-1}, 1)\}$, on vérifie facilement que pour tout $i \in \{1, \ldots, 3^k - 1\}$, si $\mu((\tau_i)_{\overrightarrow{c}}) \subseteq \zeta_j$, alors $\mu((\tau_{i+1})_{\overrightarrow{c}}) \subseteq \zeta_j$. On obtient donc une contradiction, ce qui nous permet de conclure. □

Une façon de comprendre le résultat que nous venons de présenter est de le voir comme une version quantitative du résultat d'impossibilité de Herlihy et Shavit. C'est-à-dire, dans le cas déterministe, qu'il suffit de montrer que la subdivision du simplexe de dimension 1 représentant deux processus ne peut pas être envoyé sur le complexe de sortie par une fonction simpliciale. Cette impossibilité s'étend aux dimensions supérieures par les propriétés des fonctions

simpliciales et de la subdivision : une fonction simpliciale, quand restreinte aux sous-complexes de dimension inférieure, est toujours une fonction simpliciale.

4.5.3 Accord d'ensemble k

L'accord d'ensemble k (*k-set agreement*) est une tâche dont la spécification est proche de celle du consensus, et pour laquelle le théorème de la calculabilité asynchrone de Herlihy et Shavit permet également de donner une preuve d'impossibilité, dès que k est inférieur au nombre de processus. L'argument consiste également à établir que la subdivision chromatique du complexe d'entrée ne peut pas atteindre le complexe de sortie par une fonction simpliciale qui respecte la spécification de la tâche.

Pour établir une borne inférieure de probabilité, de la même façon que pour le consensus, il nous suffit de connaître la restriction topologique à la résolution de la tâche. Nous allons à nouveau utiliser l'impossibilité de l'accord d'ensemble dans le cas déterministe, et montrer que la borne exhibée permet de raisonner par l'absurde en se ramenant à ce cas.

4.5.3.1 Description topologique de la tâche et de son impossibilité

Commençons par définir les complexes d'entrée et de sortie de l'accord d'ensemble :

Définition 55 (Accord d'ensemble k)**.** On définit $\alpha_k = (\mathcal{I}^n_{\alpha_k}, \mathcal{O}^n_{\alpha_k}, \delta_{\alpha_k})$ la tâche d'accord d'ensemble k pour n processus, et pour tout $1 < k < n$. Soit $\mathcal{V}_n = \{v_0, \ldots, v_{n-1}\}$ un ensemble de valeurs deux-à-deux distinctes.

- Le complexe d'entrée $\mathcal{I}^n_{\alpha_k}$ est composé d'un unique simplexe maximal étiqueté par les ensembles $\{(p_0, i_0), \ldots, (p_{n-1}, i_{n-1})\}$ où $\{i_0, \ldots, i_{n-1}\} \subseteq \mathcal{V}_n$.
- Les simplexes maximaux du complexe de sortie $\mathcal{O}^n_{\alpha_k}$ sont étiquetés par les ensembles $\{\{(p_0, j_0), \ldots, (p_k, j_k)\} \mid \{j_0, \ldots, j_k\} \subset \mathcal{V}_n\}$. L'inclusion est stricte car $k < n - 1$.
- La spécification δ_{α_k} est telle que pour tout simplexe σ de dimension d étiqueté par $\{(p_{i_0}, j_{i_0}), \ldots, (p_{i_d}, j_{i_d})\}$, $\delta_{\alpha_k}(\sigma) \in \{(p_{i_0}, j'_{i_0}), \ldots, (p_{i_d}, j'_{i_d}) \mid \{j'_{i_0}, \ldots, , j'_{i_d}\} \subseteq \{j_{i_0}, \ldots, , j_{i_d}\}\}$

On déduit de cette définition que le complexe de sortie est constitué de simplexes correspondant à des exécutions où les processus ont choisi des valeurs parmi un ensemble de k éléments, compris dans les valeurs d'entrée.

La spécification δ_α, à l'instar de celle du consensus, impose que les valeurs décidées soient incluses dans les valeurs proposées.

Nous représentons le complexe de sortie pour $n = 2$ en figure 4.16. On y a illustré par trois couleurs distinctes les trois configurations de sortie où les trois processus décident de la même valeur. En recollant les simplexes identiques sur la figure, on constate que l'on obtient un complexe connecté, mais avec un « trou ».

Le résultat d'impossibilité utilise cette structure. On peut le reformuler de la façon suivante :

Propriété 8 (Impossibilité topologique de l'accord d'ensemble k, Herlihy et Shavit [HS99], théorème 3.3). Pour tout $l \in \mathbf{N}$, il n'existe aucune fonction simpliciale entre $\chi^l(\mathcal{I}^n_{\alpha_k})$ et $\mathcal{O}^n_{\alpha_k}$ qui respecte la spécification δ_{α_k}.

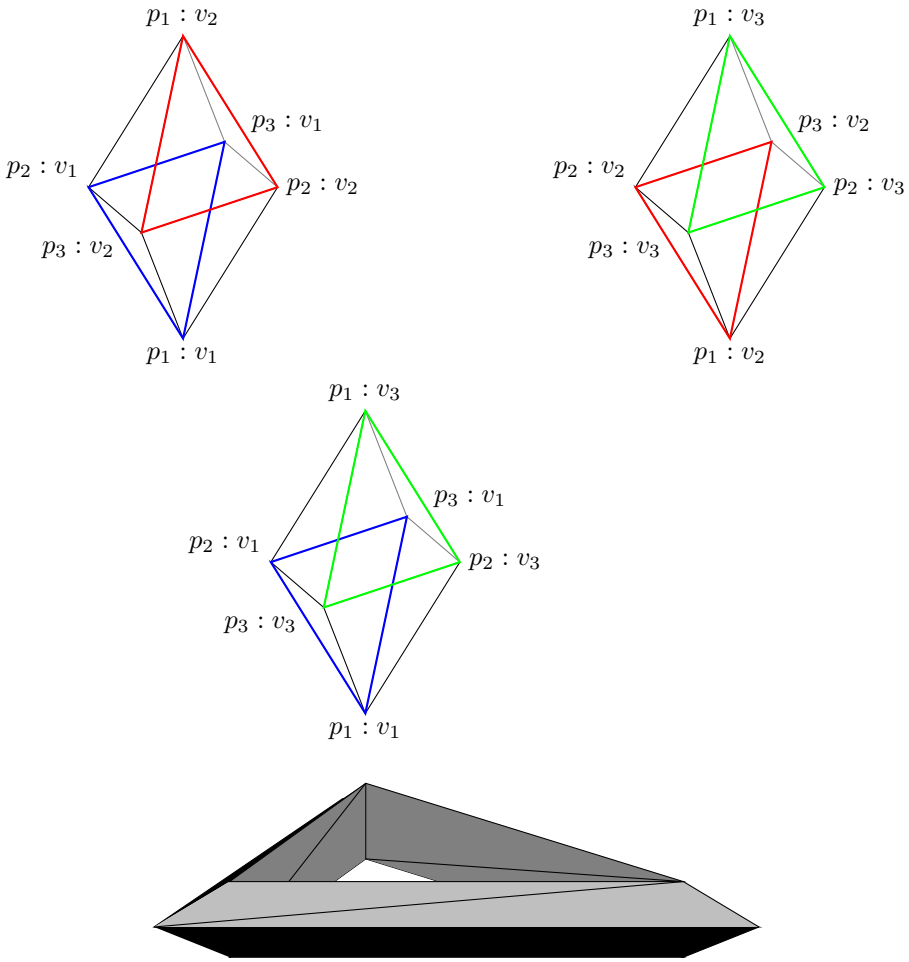


FIGURE 4.16 – Le complexe de sortie pour l'accord d'ensemble 2, avec $n = 3$.

Nous ne redonnons pas de preuve de cette propriété, mais nous arrêtons cependant un instant sur les arguments dont elle dépend.

Par la définition de la tâche, il existe $\sigma = \{(p_0, v_0), \ldots, (p_k, v_k)\} \in \mathcal{I}^n_{\alpha_k}$ um simplexe de dimension $k+1$ dans lequel chaque processus propose une valeur différente. La spécification implique que chaque face propre de ce simplexe correspond à une exécution qui, si elle donne une sortie, donne une sortie correspondant à une des valeurs associées à cette face.

Herlihy et Shavit utilisent le lemme de Sperner, qui est un analogue combinatoire du théorème de point fixe de Brouwer. Il s'applique de la façon suivante : Soit une fonction simplicale $\mu : \chi^l(\sigma) \to \mathcal{O}^n_{\alpha_k}$ telle que pour toute face propre ρ de σ, où $\rho = \{(p_{i_0}, j_{i_0}), \ldots, (p_{i_d}, j_{i_d})\}$, μ restreinte à $\chi^l(\rho))$ ne soit définie que sur des simplexes de $\mathcal{O}^n_{\alpha_k}$ dont les valeurs dont incluses dans $\{i_0, \ldots, i_d\}$ (c'est-à-dire une fonction simpliciale respectant la spécification δ_{α_k}). Le lemme de Sperner permet d'établir que dans ce cas, il existe un simplexe $\eta \in \chi^l(\sigma)$ de dimension maximale ($k+1$ donc) telle que pour toute paire de sommets (e_i, e_j)

de η, si $i \neq j$, alors $\mu(e_i) = (p_i, v_i)$ et $\mu(e_j) = (p_j, v_j)$ pour $v_i \neq v_j$.

On parvient donc à une contradiction, car il n'existe aucun simplexe dans $\mathcal{O}^n_{\alpha_k}$ de dimension $k+1$ dont les sommets sont associés à des valeurs v_i deux-à-deux distinctes. C'est le principe même de la définition de la tâche d'accord d'ensemble k d'exiger qu'aucun état de sortie ne puisse comporter $k+1$ valeurs différentes. L'intuition graphique attachée à ce raisonnement, en regard de la figure 4.16, est que la fonction simpliciale attachée à un protocole résolvant l'accord d'ensemble devrait associer un simplexe du complexe de protocole à un simplexe du complexe de sortie qui n'existe pas, car il correspond au *trou* de ce complexe.

4.5.3.2 Accord d'ensemble probabiliste et borne inférieure

Nous allons raisonner de la même façon qu'avec le consensus probabiliste, mais en adaptant l'argumentation aux considérations que nous venons de présenter. En particulier, la notion de chaîne d'indistinguabilité ne sera plus au cœur de notre argument. Nous utiliserons le fait que pour un simplexe particulier du complexe d'entrée, tous les simplexes de dimension maximale de sa subdivision chromatique ne peuvent correspondre à une résolution de la tâche. Nous avons décrit ce résultat dans sa version déterministe plus haut.

Définition 56. On définit la fonction $f_\chi : \mathbf{N} \to \mathbf{N}$, qui à un entier n associe le nombre de simplexes de dimension n dans la subdivision du simplexe coloré standard $\chi(\Delta^n)$.

Nous ne précisons pas les valeurs de f, car cela demanderait d'établir une définition inductive sur la construction de la subdivision chromatique, ce qui demanderait une étude topologique poussée de la structure des complexes qui en sont issus. Cette étude alourdirait notre propos, et nous la réservons pour de futurs travaux.

Par les exemples et définitions que l'on a déjà présenté dans ce chapitre, on peut néanmoins remarquer que $f_\chi(1) = 3$, et que $f_\chi(2) = 13$.

Nous revenons aux constructions probabilistes de la section4.4 pour que notre méthode puisse s'adapter à la nouvelle tâche qui nous occupe (citons par exemple le travail de Mostefaoui et Raynal qui étudient en détail l'accord d'ensemble aléatoire [MR01]).

On définit comme précédemment, pour tout algorithme A d'accord d'ensemble probabiliste la probabilité $\overline{p}_A(\tau)$ que A n'atteigne *pas* l'accord d'ensemble sur l'exécution correspondant à τ (on rappelle que τ est un adversaire, un ordonnancement des opérations de lecture globale et d'écriture, qui correspond à un simplexe du complexe de protocole).

De même, on définit $r^l_A = \max\{\overline{p}_A(\tau) \mid \tau \in \chi^l(\mathcal{I}^n_{\alpha_k})\}$.

Théorème 13. Pour tout algorithme (robuste) A d'accord d'ensemble k aléatoire, $r^l_A \leq \frac{1}{f_\chi(k+1)^l}$

Démonstration. Supposons par l'absurde que $r^l_A > \frac{1}{f_\chi(k+1)^l}$. Considérons également σ un simplexe d'entrée étiqueté par $k+1$ valeurs différentes (qui existe par définition de $\mathcal{I}_{\alpha_k}$). Par la définition 56, $\chi^l(\sigma)$ contient $f_\chi(k+1)^l$ simplexes de dimension $k+1$. On note ces simplexes $\{\tau_1, \ldots, \tau_{f(k+1)^l}\}$ Par définition, la probabilité qu'au moins l'un de ces simplexes correspondent à une exécution

qui ne termine pas est de $\sum_{i=1}^{f_\chi(k+1)^l} \overline{p}(\tau_i) \leq f_\chi(k+1)^l(r_A^l)$. Par hypothèse, $f_\chi(k+1)^l(r_A^l) < 1$.

On en déduit qu'il existe une probabilité non nulle que tous les simplexes τ_i correspondent à une résolution de l'algorithme pour *la même* suite de lancers de pièces $\overrightarrow{c}$. C'est-à-dire qu'il existerait une suite $\overrightarrow{c}$ et une fonction simpliciale μ respectant la spécification de la tâche telle que $\mu((\tau_i)_{\overrightarrow{c}}) \neq \perp$ pour tout $i \in \{1, \ldots, f_\chi(k+1)^l\}$. Or on peut dans ce cas revenir à la propriété 8. En particulier, le lemme de Sperner s'applique aussi bien dans ce cas, et l'on déduit qu'il existe $i \in \{1, \ldots, f_\chi(k+1)^l\}$ tel que si $\mu((\tau_i)_{\overrightarrow{c}})$ est défini, alors $\mu((\tau_i)_{\overrightarrow{c}})$ est égal à un simplexe étiqueté par $k+1$ valeurs différentes, qui n'apparaît pas dans $\mathcal{O}_\alpha^n$. On parvient donc à une contradiction, ce qui nous permet de conclure. □

À nouveau, ce résultat peut être vu comme une version quantitative du résultat topologique d'impossibilité. Nous avons utilisé l'hypothèse sur la probabilité d'échec pour se ramener au cas déterministe et importer l'argument de Herlihy et Shavit.

Remarquons ici que contrairement au consensus, l'accord d'ensemble k nécessite que l'on considère les simplexes de dimension $k+1$, et que la borne dépend donc du paramètre k de la tâche, qui est borné par le nombre de processus. Pour le consensus, l'argument de connexité nécessitait seulement de remarquer que deux processus choisis étaient toujours reliés par un 1-simplexe, dont la subdivision permettait de construire une chaîne suffisante à notre argument. Pour l'accord d'ensemble, nous choisissons également un simplexe, mais comme il y a plus de deux valeurs, ce simplexe dans lequel toutes les valeurs sont différentes est de dimension $k+1$. Et comme le lemme de Sperner ne permet pas de choisir de façon déterministe l'élément de la subdivision qui ne respecte pas la spécification, il nous faut raisonner sur tous les simplexes qui composent la subdivision d'un complexe d'entrée, en sachant qu'au moins un d'entre eux ne peut pas satisfaire les conditions voulues. C'est pour cela que la borne dépend du nombre de simplexes de dimension maximale apparaissant dans la subdivision d'un simplexe choisi.

4.5.3.3 Une application : le consensus

Nous pouvons comparer les bornes inférieures pour l'accord d'ensemble k et pour le consensus de façon à se convaincre que les deux méthodes employées sont cohérentes entre elles.

En effet, il est bien connu que le consensus est la même tâche que l'accord d'ensemble 1 (le lecteur peut comparer les définitions correspondantes pour s'en assurer). Or, on a vu que $f_\chi(1) = 3$. En particulier, cela implique que la borne inférieure de probabilité de l'accord d'ensemble 1 pour l tours d'exécution, en suivant le théorème 13, est égale à $\frac{1}{3^l}$. C'est bien la même borne que celle donnée pour le consensus depuis le théorème 12.

4.6 Conclusions

Nous avons présenté dans ce chapitre un résultat de borne inférieure pour le consensus probabiliste, dans les systèmes de registres à mémoire partagée d'écriture/lecture globale (*snapshot*). Ce résultat consiste en une transposition

de l'argument d'Attiya et Censor au modèle des complexes simpliciaux adaptés à un cadre probabiliste. Les bornes obtenues sont du même ordre de grandeur dans le cas d'algorithmes robustes, malgré le fait que nous travaillons dans un modèle différent. Cette différence, nous l'avons dit, tient à ce que l'on ait un quotient sur les traces d'équivalence et n'isolons pas les opérations de la même façon que dans celui d'Attiya et Censor qui est à plus bas niveau, et considère également des opérations plus élémentaires.

Le même type de résultat adapté à la tâche de l'accord d'ensemble est également prometteur quant à cette démarche. En effet, on parvient à nouveau à utiliser les arguments topologiques dans un cadre quantitatif de façon à obtenir un nouveau résultat concernant le comportement probabiliste de ces algorithmes.

Cette approche semblant prometteuse, une direction de recherche qui paraît intéressante serait de continuer à explorer d'autres tâches se prêtant à ce type d'argument, et d'aller éventuellement vers une généralisation de notre démarche. C'est-à-dire que nous utilisons le résultat général de la calculabilité asynchrone, mais que nous revenons aux propriétés particulières de différents complexes simpliciaux pour pouvoir en donner une version quantitative. Peut-être les outils topologiques permettraient de donner un résultat plus général sur les tâches et les protocoles probabilistes.

Une autre perspective de recherche dans la continuité de ces travaux serait de relier les modèles présentés dans ce chapitre avec les réseaux de preuve du chapitre 2. Traiter les effets de mémoire, l'asynchronicité et la tolérance aux pannes paraît être un défi intéressant. Une direction envisageable serait de s'inspirer des récents travaux de Hamdaoui [Ham18] qui a étendu l'encodage du π-calcul dans les réseaux, encodage initialement dû à Ehrhard et Laurent [EL07], à un λ-calcul concurrent avec références.

Encore un autre point qui permettrait de prolonger notre étude serait de se pencher sur les algorithmes t-résilients. En effet, notre approche est restreinte aux protocoles robustes, car l'interprétation topologique de la communication que nous avons présentée ne permet pas de caractériser simplement une tolérance bornée aux pannes. La restriction que cela impose sur la communication donne lieu à la construction d'un complexe de protocole dont la structure est bien moins simple que celde que nous avons présentée, en particulier il contient des trous et n'est pas simplement connexe.

Annexes

Nomenclature

$\sim_l^{p,I}$ Relation d'adjacence entre deux arbres d'une structure p, pour la position d'interrupteurs I, indexée par une étiquette l (définition 3)

$\mathbf{a}_\kappa, \overline{\mathbf{a}}_\lambda$ Constantes indexées pour les affaiblissements et coaffaiblissements dans $\mathbf{DLL}_0$, respectivement sur $\mathbf{U}_?$ et $\mathbf{U}_!$. (définition 9).

$\mathbf{U}_?, \mathbf{U}_!$ Ensembles d'affaiblissements et coaffaiblissements dans les réseaux (définition 9)

ξ^* Si ξ est un chemin dans le développement de Taylor du réseau P, ξ^* est un chemin dans P. (définition 23)

t^* Si t est un arbre dans le développement de Taylor du réseau P, t^* est un arbre de P. (définition 23)

$\mathbf{ar}(b)$ Arité de la boîte b (nombre de portes auxiliaires) (définition 20)

$\mathbf{A}$ Atomes : feuilles des arbres dans les réseaux (variables, (co)unités et (co)affaiblissements).

$\mathbf{C}(p)$ Ensemble des coupures d'une structure p

$\mathbf{Ch}(p)$ Ensemble des chemins (d'interrupteurs) d'une structure p

$\frown$ Relation de cohérence sur Δ_{pv}

:: Concaténation de suites

$\langle t|s\rangle$ Coupure entre les deux arbres t et s

$\mathsf{deg}_x(M)$ Nombre d'occurrences libres de x dans M

$\Delta^k(\overrightarrow{t})$ Ensemble des découpages de $(\overrightarrow{t})$ en k suites de termes, respectant l'ordre relatif de $\overrightarrow{t}$ (définition 13)

Δ^n Chapitre 3 : Calcul à ressources pour l'Appel-Par-Nom (figure 3.9)

Δ^n Chapitre 4 : simplexe standard de dimension n (définition 45)

Δ^v Chapitre 3 :Calcul à ressources pour l'Appel-Par-Valeur (figure 3.9)

$\mathbf{DLL}_0$ Réseaux à ressources (fragment multilinéaire de la Logique Linéaire différentielle, définition 9)

$\mathbf{DLL}_0^+$ Sommes finies de réseaux à ressources

$\mathbf{deg}(p)$ $\max\{\mathbf{deg}_p(t) \mid t \in \mathbf{SA}(p)\}$

$\mathbf{deg}_p(t)$ Degré de saut de t : nombre d'affaiblissements ou de co-unités α tels que $S_p(\alpha) = t$ (définition 18)

$\mathbf{E}^k(t)$ Nombre d'unités dont le saut pointe vers t, héréditairement, à travers une suite de k coupures evanescentes (définition 19)

$\overline{\xi}$ Chemin construit en parcourant le chemin ξ à rebours (définition 3)

Λ_{pv} Appel-Par-Pousse-Valeur (définition 29)

$\mathbf{Ch}^\bullet(p)$ Ensemble des chemins longs de p (définition 17).

$\mathbf{ln}(\chi)$ Nombre d'arbres qui composent le chemin χ (définition 6)

$\mathbf{ln}(p)$ Longueur maximale d'un chemin dans p (définition 6)

$\mathbf{MLL}^-$ Réseaux multiplicatifs sans unités

Δ_{pv} Appel-Par-Pousse-Valeur à ressources (définition 30)

$\mathbf{S}^X$ Semi module des termes de l'ensemble X sur un semi anneau $\mathbf{S}$ avec fractions (section 1.4)

$\mathbf{SA}(p)$ Sous-arbres d'une structure p

$\mathbf{split}^k(m)$ Découpage d'un terme $m \in \Delta_{\mathsf{pv}}$ en un k-uplet de termes de la même forme (définition 31)

$\mathbf{split}_+$ Version quantitative de **split**, avec coefficients (définition 35).

S_p Fonction de saut, ayant les affaiblissements (et co-unités) de p pour domaine, et $\mathbf{SA}(p)$ pour codomaine (définition 9)

$\mathcal{T}$ Développement de Taylor des réseaux de preuve. $\mathcal{T} : \mathbf{MELL} \to \mathbf{S}^{\mathbf{DLL}_0}$ (définition 22)

$\mathcal{T}^n$ Développement de Taylor du λ-calcul en Appel-Par-Nom (figure 3.8)

$\mathcal{T}^v$ Développement de Taylor du λ-calcul en Appel-Par-Valeur (figure 3.8)

$\mathcal{T}_{\mathsf{pv}}$ Développement de Taylor ensembliste pour l'Appel-Par-Pousse-Valeur (définition 33)

Θ, Ξ Contextes de boîte (définition 20)

$\mathbf{taille}(p)$ Taille de la structure p (nombre de sous-arbres)

$\mathbf{Ptaille}(P)$ Taille d'un réseau de **MELL** à toute profondeur. Le contenu des boîtes est considéré.

$\Rightarrow$ Extension de la réduction parallèle de $\mathbf{DLL}_0$ aux combinaisons linéaires de réseaux à ressources (définition 28)

$\Rightarrow_r$ Extension de l'expansion Rétoré aux combinaisons linéaires de réseaux à ressources (définition 28)

$\Rightarrow_{(r)}$ Extension de $\rightrightarrows_{(r)}$ aux combinaisons linéaires de réseaux à ressources (définition 28)

$\to_{\mathsf{ax}}$ Élimination d'une coupure d'axiome (définition 2)

$\to_{\langle !|c\rangle}$ (resp.$\rightrightarrows_{\langle !|c\rangle}$) Réduction (resp. réduction parallèle) contraction/! dans $\mathbf{DLL}_0^+$ (définition 13)

$\to_{\langle c|\overline{\mathbf{a}}\rangle}$ (resp.$\rightrightarrows_{\langle \overline{\mathbf{a}}|c\rangle}$) Réduction (resp. réduction parallèle) contraction/coaffaiblissement dans $\mathbf{DLL}_0$ (définition 13)

$\rightarrow_{\langle !|d\rangle}$ (resp.$\rightrightarrows_{\langle !|d\rangle}$) Réduction (resp. réduction parallèle) déréliction/! dans $\mathbf{DLL}_0$ (définition 13)

$\rightarrow_{\epsilon}$ (resp.$\rightrightarrows_{\epsilon}$) Réduction (resp. réduction parallèle) évanescente dans $\mathbf{DLL}_0$ (définition 13)

$\rightarrow_{m}$ Élimination d'une coupure multiplicative (définition 2)

$\rightarrow_{\langle a_0|\mathbf{a}\rangle}$ Réduction affaiblissement/boîte dans **MELL** (définition 21)

$\rightarrow_{\langle a_0|a\rangle}$ Réduction boîte/boîte dans **MELL** (définition 21)

$\rightarrow_{\langle a_0|c\rangle}$ Réduction contraction/boîte dans **MELL** (définition 21)

$\rightarrow_{\langle a_0|d\rangle}$ Réduction déréliction/boîte dans **MELL** (définition 21)

$\rightarrow_{\mathsf{pv}}$ Réduction dans l'Appel-Par-Pousse-Valeur (définition 29)

$\rightarrow_{\mathsf{r_{ass}}}$ (resp. $\rightrightarrows_{\mathsf{r_{ass}}}$) Expansion (resp. parallèle) Rétoré pour l'associativité (définition 25)

$\rightarrow_{\mathsf{r_c}}$ (resp. $\rightrightarrows_{\mathsf{r_c}}$) Expansion (resp. parallèle) Rétoré pour la contraction unaire (définition 25)

$\rightarrow_{\mathsf{r_{perm}}}$ (resp. $\rightrightarrows_{\mathsf{r_{perm}}}$) Expansion (resp. parallèle) Rétoré pour la permutation (définition 25)

$\rightarrow_{\mathsf{rpv}}$ Réduction dans Δ_{pv} (définition 32)

$\rightarrow^{=}_{\mathsf{rpv}}$ Clôture réflexive de $\rightarrow_{\mathsf{rpv}}$

$\rightarrow_{\mathsf{rpv}^+}$ Réduction à ressources générant des sommes finies dans Δ_{pv} (définition 36)

$\gg$ Réduction au support : si $p \rightarrow p_1 + \ldots + p_n$ dans $\mathbf{DLL}_0^+$, alors $p \gg p_i$ pour tout i, dans $\mathbf{DLL}_0$

$\gg_{(r)}$ Réduction au support $\gg$ de $\mathbf{DLL}_0$ suivie d'une expansion Rétoré parallèle

$\rightrightarrows$ Élimination des coupures en parallèle (définie dans **MLL** et dans $\mathbf{DLL}_0^+$)

$\rightrightarrows_{(r)}$ Réduction parallèle de $\mathbf{DLL}_0^+$ suivie d'une expansion Rétoré parallèle

$\rightrightarrows_{\mathsf{ax}}$ Élimination des coupures d'axiomes en parallèle (définition 4)

$\rightrightarrows^{\mathsf{neut}}_{\langle \overline{\mathbf{a}}|c\rangle}$ Expansion Rétoré pour la neutralité de l'affaiblissement (définition 24)

$\rightrightarrows_{m}$ Élimination de coupures multiplicatives en parallèle (définition 4)

$\rightrightarrows_{\mathsf{r}}$ Expansion Rétoré en parallèle

$\rightarrow_0$ Réduction en zéro dans $\mathbf{DLL}_0^+$ (définition 13)

$\mathbf{U_1}, \mathbf{U}_\perp$ Unités et co-unités multiplicatives dans les réseaux

$\mathcal{V}$ Ensemble dénombrable de variables

ξ^- Si $p \rightrightarrows q$ et $\xi \in \mathbf{Ch}(q)$, ξ^- est un chemin de p de mêmes extrémités que ξ (lemme 3)

a_i^b Porte de la boîte b d'indice i ($i = 0$ pour la porte principale) (définition 20)

B_Θ	Domaine du contexte de boîte Θ (définition 20)
$f_{\mathbf{ln}}$	Fonction qui borne l'augmentation de la longueur des chemins sous réduction multiplicative parallèle (lemme 6)
I_p	Position d'interrupteurs pour la structure p (définition 3)
$x, \overline{x}$	Axiome

Table des figures

Bibliographie

[ABD95] Hagit Attiya, Amotz Bar-Noy, and Danny Dolev. Sharing memory robustly in message-passing systems. *J. ACM*, 42(1) :124–142, 1995.

[AC08] Hagit Attiya and Keren Censor. Lower bounds for randomized consensus under a weak adversary. In *Proceedings of the Twenty-Seventh Annual ACM Symposium on Principles of Distributed Computing, PODC 2008, Toronto, Canada, August 18-21, 2008*, pages 315–324, 2008.

[ACT98] Marcos Kawazoe Aguilera, Wei Chen, and Sam Toueg. Failure detection and consensus in the crash-recovery model. In *Distributed Computing, 12th International Symposium, DISC '98, Andros, Greece, September 24-26, 1998, Proceedings*, pages 231–245, 1998.

[AGL19] Beniamino Accattoli, Giulio Guerrieri, and Maico Leberle. Types by need. In *Programming Languages and Systems - 28th European Symposium on Programming, ESOP 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings*, pages 410–439, 2019.

[AH90] James Aspnes and Maurice Herlihy. Fast randomized consensus using shared memory. *J. Algorithms*, 11(3) :441–461, 1990.

[And90] James H. Anderson. Composite registers. In *Proceedings of the Ninth Annual ACM Symposium on Principles of Distributed Computing, Quebec City, Quebec, Canada, August 22-24, 1990*, pages 15–29, 1990.

[Asp02] James Aspnes. Randomized protocols for asynchronous consensus. *CoRR*, cs.DS/0209014, 2002.

[Ben83] Michael Ben-Or. Another advantage of free choice : Completely asynchronous agreement protocols (extended abstract). In *Proceedings of the Second Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing, Montreal, Quebec, Canada, August 17-19, 1983*, pages 27–30, 1983.

[BM10] Patrick Baillot and Damiano Mazza. Linear logic by levels and bounded time complexity. *Theoretical Computer Science*, 411(2) :470–503, 2010.

[BMZ88] Ofer Biran, Shlomo Moran, and Shmuel Zaks. A combinatorial characterization of the distributed tasks which are solvable in the presence of one faulty processor. In *Proceedings of the Seventh Annual*

ACM Symposium on Principles of Distributed Computing, Toronto, Ontario, Canada, August 15-17, 1988, pages 263–275, 1988.

[Boo84] George Boolos. Don't eliminate cut. *J. Philosophical Logic*, 13(4) :373–378, 1984.

[CF58] Haskell Brooks Curry and Robert Feys. *Combinatory Logic*. Number vol. 2 in Combinatory Logic. North-Holland Publishing Company, 1958.

[Cho15] Jules Chouquet. Exponentielles et connexité dans les réseaux à la Ehrhard. Mémoire de master, Université Paris-Diderot, 2015.

[Cho16] Jules Chouquet. L'identité des preuves : normalisation, réseaux et séparation. Mémoire de master, Université Paris 1 - Panthéon Sorbonne, 2016.

[Cho19] Jules Chouquet. Taylor expansion, finiteness and strategies. In *MFPS 2019*, 2019.

[CIL94] Benny Chor, Amos Israeli, and Ming Li. Wait-free consensus using asynchronous hardware. *SIAM J. Comput.*, 23(4) :701–712, 1994.

[CM89] Benny Chor and Lior Moscovici. Solvability in asynchronous environments (extended abstract). In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989*, pages 422–427, 1989.

[CT20] Jules Chouquet and Christine Tasson. Taylor expansion for Call-By-Push-Value. In *CSL 2020*, À PARAÎTRE (janvier 2020).

[CV18] Jules Chouquet and Lionel Vaux Auclair. An application of parallel cut elimination in unit-free multiplicative linear logic to the Taylor expansion of proof nets. In *CSL 2018*, 2018.

[dC15] Daniel de Carvalho. The relational model is injective for multiplicative exponential linear logic. *CoRR*, abs/1502.02404, 2015.

[DFKM97] Danny Dolev, Roy Friedman, Idit Keidar, and Dahlia Malkhi. Failure detectors in omission failure environments. In *Proceedings of the Sixteenth Annual ACM Symposium on Principles of Distributed Computing, Santa Barbara, California, USA, August 21-24, 1997*, page 286, 1997.

[Dos02] Kosta Dosen. Identity of Proofs Based on Normalization and Generality. *arXiv Mathematics e-prints*, page math/0208094, Aug 2002.

[DS19] Abhishek De and Alexis Saurin. Infinets : The parallel syntax for non-wellfounded proof-theory. In *Automated Reasoning with Analytic Tableaux and Related Methods - 28th International Conference, TABLEAUX 2019, London, UK, September 3-5, 2019, Proceedings*, pages 297–316, 2019.

[EG16] Thomas Ehrhard and Giulio Guerrieri. The bang calculus : an untyped lambda-calculus generalizing call-by-name and call-by-value. In *Principles and Practice of Declarative Programming*, 2016.

[Ehr05] Thomas Ehrhard. Finiteness spaces. *Mathematical Structures in Computer Science*, 2005.

[Ehr10] Thomas Ehrhard. A finiteness structure on resource terms. In *LICS 2010, 11-14 July 2010, Edinburgh, United Kingdom*, 2010.

[Ehr12] Thomas Ehrhard. Collapsing non-idempotent intersection types. In *Computer Science Logic CSL 2012*, pages 259–273, 2012.

[Ehr14] Thomas Ehrhard. A new correctness criterion for MLL proof nets. In *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS),CSL-LICS '14, Vienna, Austria, July 14 - 18, 2014*, pages 38 :1–38 :10, 2014.

[Ehr16a] Thomas Ehrhard. Call-by-push-value from a linear logic point of view. In *Programming Languages and Systems - 25th European Symposium on Programming, ESOP 2016*, pages 202–228, 2016.

[Ehr16b] Thomas Ehrhard. An introduction to differential linear logic : proof-nets, models and antiderivatives. *CoRR*, abs/1606.01642, 2016.

[EL07] Thomas Ehrhard and Olivier Laurent. Interpreting a finitary pi-calculus in differential interaction nets. In *CONCUR 2007 - Concurrency Theory, 18th International Conference, CONCUR 2007, Lisbon, Portugal, September 3-8, 2007, Proceedings*, pages 333–348, 2007.

[ER03] Thomas Ehrhard and Laurent Regnier. The differential lambda-calculus. *Theor. Comput. Sci.*, 2003.

[ER05] Thomas Ehrhard and Laurent Regnier. Differential interaction nets. *Electr. Notes Theor. Comput. Sci.*, 123 :35–74, 2005.

[ER08] Thomas Ehrhard and Laurent Regnier. Uniformity and the Taylor expansion of ordinary lambda-terms. *Theor. Comput. Sci.*, 403(2-3) :347–372, 2008.

[ET16] Thomas Ehrhard and Christine Tasson. Probabilistic Call-By-Push-Value. *CoRR*, abs/1607.04690, 2016.

[FLP85] Michael J. Fischer, Nancy A. Lynch, and Mike Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2) :374–382, 1985.

[Fre80] Gottlob Frege. *The Foundations of Arithmetic : A Logico-Mathematical Enquiry Into the Concept of Number*. Northwestern University Press, 1980.

[Fre99] Gottlob Frege. *Idéographie*. Bibliothèque des textes philosophiques. J. Vrin, 1999.

[Gen55] Gerhard Gentzen. *Recherches sur la déduction logique : (Untersuchungen über das Logische Schliessen)*. Philosphie de la matière. Presses Univ. de France, 1955.

[Gir87] Jean-Yves Girard. Linear logic. *Theor. Comput. Sci.*, 50 :1–102, 1987.

[Gir06] Jean-Yves Girard. *Le point aveugle : cours de logique. 1. Tome 1, Vers la perfection*. Number vol. 1 in Visions des sciences. Hermann, 2006.

[GM19] Giulio Guerrieri and Giulio Manzonetto. The bang calculus and the two girard's translations. *CoRR*, abs/1904.06845, 2019.

[GMT18] Éric Goubault, Samuel Mimram, and Christine Tasson. Geometric and combinatorial views on asynchronous computability. *Distributed Computing*, 31(4) :289–316, 2018.

[GPT16] Giulio Guerrieri, Luc Pellissier, and Lorenzo Tortora de Falco. Computing connected proof(-structure)s from their Taylor expansion. In *1st International Conference on Formal Structures for Computation and Deduction, FSCD 2016, June 22-26, 2016, Porto, Portugal*, pages 20 :1–20 :18, 2016.

[GPT19] Giulio Guerrieri, Luc Pellissier, and Lorenzo Tortora de Falco. Proof-net as graph, taylor expansion as pullback. In *Logic, Language, Information, and Computation - 26th International Workshop, WoLLIC 2019, Utrecht, The Netherlands, July 2-5, 2019, Proceedings*, pages 282–300, 2019.

[Ham18] Yann Hamdaoui. *Concurrency, references and Linear Logic*. Thèse de doctorat, Université Paris-Diderot, September 2018.

[Her91] Maurice Herlihy. Wait-free synchronization. *ACM Trans. Program. Lang. Syst.*, 13(1) :124–149, 1991.

[HHS18] Willem Heijltjes, Dominic J D Hughes, and Lutz Straßburger. Proof nets for first-order additive linear logic. Research Report RR-9201, Inria, September 2018.

[HKR13] Maurice Herlihy, Dmitry N. Kozlov, and Sergio Rajsbaum. *Distributed Computing Through Combinatorial Topology*. Morgan Kaufmann, 2013.

[HS93] Maurice Herlihy and Nir Shavit. The asynchronous computability theorem for t-resilient tasks. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, May 16-18, 1993, San Diego, CA, USA*, pages 111–120, 1993.

[HS99] Maurice Herlihy and Nir Shavit. The topological structure of asynchronous computability. *J. ACM*, 46(6) :858–923, 1999.

[KMP18] Emma Kerinec, Giulio Manzonetto, and Michele Pagani. Revisiting Call-by-value Böhm trees in light of their Taylor expansion. *CoRR*, abs/1809.02659, 2018.

[Koz12] Dmitry N. Kozlov. Chromatic subdivision of a simplicial complex. *Homology Homotopy Appl.*, 14(2) :197–209, 2012.

[Laf90] Yves Lafont. Interaction nets. In *Conference Record of the Seventeenth Annual ACM Symposium on Principles of Programming Languages (POPL), San Francisco, California, USA, January 1990*, pages 95–108, 1990.

[Laf94] Yves Lafont. From proof-nets to interaction nets. In *Advances in Linear Logic*, pages 225–247. Cambridge University Press, 1994.

[Laf97] Yves Lafont. Interaction combinators. *Information and Computation*, 137(1) :69–101, 1997.

[Lau02] Olivier Laurent. *Etude de la polarisation en logique*. Thèse de doctorat, Université Aix-Marseille II, March 2002.

[Lev06] Paul Blain Levy. Call-by-push-value : Decomposing call-by-value and call-by-name. *Higher-Order and Symbolic Computation*, 19(4) :377–414, 2006.

[LMMP13] Jim Laird, Giulio Manzonetto, Guy McCusker, and Michele Pagani. Weighted relational models of typed lambda-calculi. In *LICS 2013*, pages 301–310, 2013.

[Mac94] Ian Mackie. *Geometry of implementation*. PhD thesis, Imperial college of science, technology and medicine, September 1994.

[Maz06] Damiano Mazza. Linear logic and polynomial time. *Mathematical Structures in Computer Science*, 16(6) :947–988, 2006.

[Mel09] Paul-André Melliès. Categorical semantics of linear logic. In *In : Interactive Models of Computation and Program Behaviour, Panoramas et Synthèses 27, Société Mathématique de France 1–196*, 2009.

[MR01] Achour Mostefaoui and Michel Raynal. Randomized k-set agreement. In *Proceedings of the Thirteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, SPAA '01, pages 291–297, New York, NY, USA, 2001. ACM.

[PdRR99] Alberto Pravato, Simona Ronchi della Rocca, and Luca Roversi. The call-by-value λ-calculus : a semantic investigation. *Mathematical Structures in Computer Science*, 9(5) :617–650, 1999.

[Pra75] Dag Prawitz. Ideas and results in proof theory. *Journal of Symbolic Logic*, 40(2) :232–234, 1975.

[PT17] Michele Pagani and Paolo Tranquilli. The conservation theorem for differential nets. *Mathematical Structures in Computer Science*, 27(6) :939–992, 2017.

[PTV16] Michele Pagani, Christine Tasson, and Lionel Vaux. Strong normalizability as a finiteness structure via the Taylor expansion of lambda-terms. In *FOSSACS 2016*, pages 408–423, 2016.

[Reg92] Laurent Regnier. *Lambda-calcul et réseaux*. PhD thesis, Université Paris 7, Paris, France, December 1992.

[Ret03] Christian Retoré. Handsome proof-nets : perfect matchings and cographs. *Theor. Comput. Sci.*, 294(3) :473–488, 2003.

[Str06] Lutz Straßburger. Proof nets and the identity of proofs. *CoRR*, abs/cs/0610123, 2006.

[TAO18] Takeshi Tsukada, Kazuyuki Asada, and C.-H. Luke Ong. Species, profunctors and Taylor expansion weighted by SMCC. In *LICS 2018, Oxford, UK, July 09-12, 2018*, pages 889–898, 2018.

[Tor03] Lorenzo Tortora de Falco. The additive multiboxes. *Annals of Pure and Applied Logic*, 120(1) :65 – 102, 2003.

[Tra09] Paolo Tranquilli. Confluence of pure differential nets with promotion. In *Computer Science Logic, 23rd international Workshop, CSL 2009, 18th Annual Conference of the EACSL, Coimbra, Portugal, September 7-11, 2009. Proceedings*, pages 500–514, 2009.

[Vau09] Lionel Vaux. The algebraic lambda calculus. *Mathematical Structures in Computer Science*, 19(5) :1029–1059, 2009.

[Vau17] Lionel Vaux. Taylor expansion, β-reduction and normalization. In *26th EACSL Annual Conference on Computer Science Logic, CSL 2017, Stockholm, Sweden*, pages 39 :1–39 :16, 2017.

[Wid01] Filip Widebäck. *Identity of Proofs*. Acta Universitatis Stockholmiensis. Almqvist & Wiksell International, 2001.

[WR27] Alfred North Whitehead and Bertrand Russell. *Principia Mathematica*. Cambridge University Press, 1925–1927.