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

Preuves, réfutations et contre-modèles dans des logiques intuitionnistes

THÈSE

présentée et soutenue publiquement le 15 décembre 2000

pour l'obtention du

Doctorat de l'université Henri Poincaré – Nancy 1

(spécialité informatique)

par

Dominique Larchey-Wendling

Composition du jury

Président :	Jean-Paul Haton, Professeur, UHP Nancy 1, Nancy, France
Rapporteurs :	Jean Goubault-Larrecq, Professeur, ENS Cachan, Cachan, France David Pym, Professeur, Queen Mary & Westfield College, Londres, Royaume Uni Michael Rusinowitch, Directeur de Recherche, INRIA, Nancy, France
Examinateurs :	Gilles Dowek, Chargé de Recherche, INRIA, Rocquencourt, France Didier Galmiche, Maître de Conférences, UHP Nancy 1, Nancy, France

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thloria.

À mes parents, à Charlie.

À mon grand-père.

Remerciements

Je tiens à remercier les membres du jury pour l'honneur qu'ils m'ont fait de participer à ma soutenance de thèse :

- M. Jean Goubault-Larrecq, professeur à l'École Normale Supérieure de Cachan, qui a accepté d'être rapporteur de ma thèse, et qui m'encourage depuis longtemps dans mes développements en gestion des ressources, concernant la recherche de preuves en particulier, et dont l'ouverture et la vivacité d'esprit sont exemplaires. Je tiens de plus à le remercier pour les nombreuses remarques constructives qu'il m'a rapportées suite à une lecture très approfondie de ce document ;
- M. David Pym, "professor of Logic" au Queen Mary and Westfield College à Londres, qui a lui aussi accepté d'être rapporteur de ma thèse, et avec qui j'ai eu de nombreux contacts très enrichissants tout au long de ma thèse. Je tiens à dire que son soutien m'a été très précieux ;
- M. Michael Rusinowitch, directeur de recherches à l'INRIA Lorraine, qui a accepté d'être rapporteur interne de ma thèse. Je suis sensible à l'intérêt dont il a fait preuve pour le contenu de ce mémoire et à l'attention qu'il a consacrée à sa lecture ;
- M. Jean-Paul Haton, professeur à l'université Henri Poincaré de Nancy et membre de l'Institut Universitaire de France, qui m'a fait l'honneur et le plaisir de présider ce jury ;
- M. Gilles Dowek, chargé de recherches à l'INRIA Rocquencourt et responsable scientifique du projet LogiCal, qui a accepté de faire partie de mon jury, et dont les nombreuses questions sont une source constructive de recherches à venir. Je tiens aussi à remercier Gilles pour ses encouragements ;
- Comment remercier un directeur de thèse pour son soutien alors que je lui dois tellement plus que cela ? Didier Galmiche a su gérer les aléas de ma vie de thésard d'une manière exceptionnelle, que ce soit du point de vue de la recherche ou du point de vue personnel. Dans un équilibre et une compréhension remarquables, il m'a aidé à garder le cap, à approfondir toujours plus, et à chercher d'autres points de vue. Voilà, merci de ta confiance, tout simplement.

Je tiens tout spécialement à remercier mes amis Alex, Stéphan, Catherine, Christophe, Fabien et Patrick sans lesquels je pense que cette thèse n'existerait pas. Mes parents, ma sœur, mes grands-parents m'ont aussi soutenu au cours de toutes ces années. Yvette, Manou, Pierrot et Stéphane m'ont fait l'honneur de venir assister à ma soutenance de thèse, ce qui m'a beaucoup touché. Merci également à mon parrain Bernard d'être venu me soutenir à cette occasion très particulière. Enfin, je remercie ma tante Corrine pour ses conseils en communication très utiles.

Parmi les collègues qui m'ont épaulé, je citerai en particulier Denis (qui m'a appris à utiliser et apprécier TEX,) Catherine (avec qui je partage un goût commun pour les films de qualité,) Raphaël, Laura, Bernard, Jean-Marc, Armelle et Daniel (qui a aussi grandement contribué aux succès du système STRIP.) J'ai beaucoup apprécié les encouragements de Noëlle Carbonell, professeur, grande conseillère et doyenne du laboratoire à ses heures perdues ainsi que ceux de Dominique Mery,[1] professeur, et à qui je souhaite une bonne et heureuse année.

Je remercie enfin tous ceux qui sont venus assister à ma soutenance de thèse et au pot qui a suivi, agrémentant le laboratoire d'une animation haute en couleurs.

[1] `http://www....`

"The light that burns twice as bright burns half as long.
And you have burned so very very brightly, Roy."
Blade Runner, 1982.

« Plus il y a d'utilisateurs de `thloria`*,*
plus il y a de gens qui ne lisent pas le manuel. »
DBR, 2000.

Résumé

Les logiques sont de puissants outils qui permettent la spécification de systèmes informatiques et la preuve de l'adéquation de leurs implantations avec ces spécifications. Dans le cadre des logiques sous-structurelles, nous mettons en place des outils de démonstration automatique et de construction de contre-modèles. Ces logiques intègrent la notion de ressource : au niveau de la recherche de preuve, la gestion des ressources permet la mise en place de procédures plus efficaces ; au niveau de l'interprétation sémantique, la notion de ressource permet de construire des modèles fidèles et complets.

Nous établissons un lien entre la notion syntaxique de réfutation et la notion sémantique de contre-modèle. Nous en déduisons des méthodes de démonstration de la propriété des modèles finis ainsi que des algorithmes de construction de contre-modèles. En logique intuitionniste propositionnelle, la gestion fine des ressources permet d'en déduire une implantation efficace de la recherche de preuves. En logique intuitionniste linéaire, les modèles à base de ressources permettent une preuve élégante de la propriété des modèles finis. Nous établissons un lien entre la sémantique des ressources et la sémantique à base de réseaux de Petri, ce qui permet de raffiner les résultats de complétude partiels connus jusqu'alors.

Mots-clés: Déduction automatique et recherche de preuve, sémantique et modèles, logiques intuitionniste et linéaire, gestion efficace des ressources.

Abstract

Logics can be used as powerful tools for specifying computer systems and proving the soundness of their implementations with respect to these specifications. In the field of substructural logics, we develop tools and methods for automated deduction and counter-model generation. These logics involve the notion of resource : at the level of proof-search, the management of resources enables more efficient procedures ; at the semantic level, resource models provide sound and complete interpretations.

We develop a link between the syntactic notion of refutation and the semantic notion of counter-model. We deduce methods for proving the finite model property and algorithms for building counter-models. In propositional intuitionistic logic, we are able to provide an efficient implementation of a proof-search procedure, based on a fine management of resources. In intuitionistic linear logic, resource based models constitute the core of an elegant proof of the finite model property. Furthermore, we establish a link between resource models and Petri net based models, from which we improve the preceding partial completeness results.

Keywords: Automated deduction and proof search, models and semantic, intuitionistic and linear logics, efficient resource management.

Table des matières

Table des figures

Introduction

Les logiques formelles sont des outils qui permettent, entre autres, de modéliser et de prouver des propriétés de systèmes informatiques, dans le but de les rendre plus fiables. L'exemple le plus simple est certainement celui de la logique classique propositionnelle qui permet de spécifier et prouver certaines propriétés des circuits logiques que l'on trouve à l'intérieur de toutes les puces ou microprocesseurs. Cependant, la prise en compte de certaines propriétés qui ne jouaient pas un rôle déterminant auparavant, du fait de la formidable miniaturisation des circuits, est devenue souhaitable voir nécessaire dans la conception des circuits. Un exemple important est celui des propriétés temporelles des circuits logiques. La complexité des puces est devenue telle que la vérification d'autres types de propriétés de ces circuits va devenir de plus en plus indispensable.

L'utilisation de la logique formelle pour certifier des systèmes informatiques nous amène à l'étude et la conception d'outils automatiques ou semi-automatiques de construction de programmes et de preuves [PM 93]. Bien que la logique classique puisse être utilisée dans de nombreux cas, dans le cadre intuitionniste, l'isomorphisme de Curry-Howard [How 80, Gir 89, Coq 90] exprime une correspondance directe entre les types des programmes fonctionnels et les formules logiques d'une part, et entre les programmes et les preuves des formules logiques d'autre part. Cette correspondance peut-être étendue au cas des logiques sous-structurelles [SH 93] qui ont émergé car elles permettent la modélisation du concept de *ressource*. La notion de ressource est alors naturellement intégrée dans le langage des types [Bie 96]. Dans le cadre de la programmation fonctionnelle, ces logiques permettent de considérer des extensions du λ-calcul prenant en compte la gestion des ressources [Abr 93, Lin 92a]. Un deuxième aspect de l'isomorphisme de Curry-Howard est la construction automatique d'un programme (ou d'un système) à partir de la preuve formelle d'une spécification, ce programme étant alors certifié conforme à la spécification [Nor 90, Bar 94].

Si la spécification reste à la charge du concepteur du système, la preuve de cette spécification ou celle de l'adéquation d'un système avec cette spécification peut donc parfois se faire de manière complètement automatique : c'est le cas par exemple en logique classique propositionnelle et aussi en logique intuitionniste propositionnelle comme nous le verrons. Mais même dans ces cas simples, la vérification de la validité d'une proposition logique est un problème complexe : la logique classique est NP-complète et la logique intuitionniste PSPACE-complète [Sta 79]. Entre autres, les meilleurs algorithmes que l'on connaisse sont exponentiels en temps. Le développement de méthodes pour décider de la validité des propositions reste donc d'actualité même dans ces logiques élémentaires.

Ce problème peut être abordé de deux manières duales. La recherche d'une preuve d'une proposition à partir d'axiomes et d'un système de règles d'inférence est une manière d'aborder ce problème. Elle est fondée sur la *théorie de la preuve* [Tro 96], qui a pour but l'étude combinatoire des preuves en tant qu'objets. Cette théorie permet de décrire l'ensemble des propositions valides de manière extensionnelle, c'est-à-dire fournit des méthodes pour construire effectivement des propositions valides.

L'autre manière d'aborder le problème de la validité d'une proposition est l'utilisation de la *théorie des modèles* [Cha 73]. Son but est de fournir des outils d'interprétation des propositions dans des structures mathématiques, en leur donnant ainsi un sens : c'est ce que l'on appelle aussi la sémantique. Les propriétés de l'interprétation peuvent permettre d'apporter une réponse négative à la validité d'une proposition. On obtient alors ce que l'on appelle un *contre-modèle*. Il constitue un argument synthétique qui permet d'affirmer l'invalidité d'une proposition.[2] En ce sens, la théorie des modèles est duale à la théorie de la preuve car elle décrit les propositions valides de manière intentionnelle, en fournissant des critères de validité qui permettent d'exclure certaines propositions invalides.

Notre étude porte sur les méthodes de recherche de preuves et l'analyse de l'échec de cette recherche dans le but de construire des contre-modèles : nous verrons que l'échec correspond à la construction d'une réfutation de laquelle on cherche à extraire les informations permettant d'engendrer un contre-modèle. La *complétude* d'un système logique par rapport à une sémantique donnée nous permet d'aborder le problème de deux manières. Pour une sémantique fixée, il est possible de raffiner un système dans le but d'optimiser la recherche de preuve. De manière duale, pour un système logique fixé, nous pouvons étudier diverses interprétations sémantiques de manière à choisir celles qui se prêtent la mieux à la représentation des contre-modèles.

Une preuve de complétude sémantique constructive contient en son sein la donnée d'un algorithme de construction de contre-modèles. Nous montrons qu'il est possible de fonder une preuve de complétude sur un algorithme qui construit, pour une formule logique donnée, soit une preuve, soit une réfutation, en combinaison avec une procédure qui construit un contre-modèle à partir d'une réfutation. Si la sémantique est basée sur une classe de modèles finis, nous obtenons par là même une démonstration de la *propriété des modèles finis*.

Cette thèse porte sur l'étude de la logique intuitionniste propositionnelle et de la logique intuitionniste linéaire propositionnelle aussi bien du point de vue de la théorie de la preuve que du point de vue de la théorie des modèles. Nous étudions la complétude de systèmes de règles d'inférence ainsi que celle de sémantiques, pour ces deux logiques. Nous en dérivons des techniques de construction de preuves ou de contre-modèles. Dans le cas de la logique intuitionniste, nous fournissons de plus un algorithme efficace[3] qui utilise ces techniques pour fournir une preuve ou un contre-modèle de n'importe quelle proposition intuitionniste.

Du point de vue de la théorie de la preuve, nous présentons des systèmes de règles qui ont la propriété suivante : l'espace de recherche de preuve est fini. On dit que ces systèmes sont analytiques. S'ils sont complets, n'importe quel algorithme de recherche de preuve fournit une procédure de décision pour la validité. De point de vue de la théorie des modèles, nous nous intéressons à la propriété des modèles finis, c'est-à-dire la recherche d'une classe de modèles finis pour laquelle la logique est complète. La propriété des modèles finis est une autre condition suffisante à la décidabilité d'une logique.

Ces deux approches convergent en la notion de *réfutation* qui est une notion syntaxique duale à celle de preuve, contrairement à celle de contre-modèle, qui est une notion sémantique duale à celle de preuve. Le concept de réfutation constitue un lien entre les critères d'analycité et de propriété des modèles finis : dans un système analytique, toute proposition invalide admet une réfutation finie et nous montrons comment il est possible de construire un contre-modèle fini à partir d'une telle réfutation. Cette transformation est très simple dans le cadre de la logique intuitionniste mais reste plus complexe dans la version linéaire. D'ailleurs, d'une manière plutôt surprenante, la propriété des modèles finis pour la logique intuitionniste linéaire n'est acquise

[2] Par opposition à l'échec d'une recherche de preuve, qui n'a rien de synthétique en général.
[3] Par rapport aux algorithmes préexistants.

que depuis des travaux très récents [Oka].

La thèse se divise en quatre parties. Dans les deux premières, nous présentons les concepts à la fois syntaxiques (théorie de la preuve) et sémantiques (théorie des modèles) qui permettent d'étudier la logique intuitionniste (resp. intuitionniste linéaire) dans la troisième (resp. quatrième) partie.

Dans la première partie, nous rappelons les notions de système de règles d'inférence et de preuve sous la forme d'arbre. Nous introduisons la notion de réfutation comme une notion duale à celle de preuve. Nous présentons la problématique de la recherche de preuves, à savoir d'une part, comment assurer la complétude et la terminaison d'un algorithme de recherche de preuves basé sur un système de règles d'inférence, et d'autre part, quelles sont les informations que l'on peut obtenir dans la cas où la recherche de preuves échoue. Nous définissons les notions de système analytique et de co-validité d'un système de règles d'inférence, qui combinées, garantissent la complétude et la terminaison de la recherche de preuves. D'autre part, nous définissons la notion de réfutation, duale à la notion de preuve. Nous montrons qu'il est possible de construire une réfutation lorsque que l'algorithme de recherche de preuve échoue. De cette réfutation, nous pourrons extraire les informations suffisante à la construction d'un contre-modèle.

Dans la deuxième partie, nous développons les concepts sémantiques qui permettent de construire les modèles des deux logiques étudiées. Après des rappels généraux sur la complétion des ensembles ordonnés à partir de clôtures et sur la notion de monoïde, nous approchons le problème de la construction de modèles à travers une analyse des relations entre monoïdes ordonnés, espaces de phases et quantales. La notion de quantale et celle d'algèbre de Heyting complète, qui en est un cas particulier, permettent de construire les sémantiques algébriques des deux logiques. Nous introduisons une nouvelle technique de complétion d'un monoïde ordonné en quantale basée sur la notion de plus grande prétopologie. Cette construction est au cœur du résultat de complétude et de propriété des modèles finis présenté dans la partie 4.

Dans une troisième partie, nous nous intéressons à la logique intuitionniste. Elle est introduite par le biais du calcul des séquents de Gentzen mais les résultats fondamentaux sont obtenus à partir de systèmes dérivés du calcul LJT [Dyc 92]. Nous montrons la complétude de ce système par le biais des techniques présentées dans la première partie, ce qui nous donne directement la complétude, la propriété des modèles finis et un algorithme de construction de preuves ou de contre-modèles de Kripke. Dans un deuxième temps, nous mettons en œuvre des techniques de partage de ressources (qui sont ici les formules logiques) de manière à supprimer les duplications de sous-formules au sein du système LJT. Nous définissons le système SLJ dans lequel une forme de duplication a été supprimée. Nous démontrons les résultats de complétude et de propriété de modèles finis pour ce système. Nous décrivons un algorithme qui permet de construire soit une preuve, soit un contre-modèle d'une formule intuitionniste donnée. Cet algorithme est extrait de la preuve constructive de complétude sémantique de SLJ. Enfin, en représentant les formules et les séquents par des arbres, nous mettons en œuvre une technique de partage structurel. Le système STRIP peut-être vu comme une implantation linéaire de LJT, c'est-à-dire où toutes les duplications ont été « absorbées » par le partage des formules. Au niveau sémantique, il possède les mêmes propriétés que SLJ dont il est issu, autrement dit, il existe un algorithme qui construit soit des preuves, soit des contre-modèles. Si l'on ne considère que l'exploration de l'espace de recherche de preuve, cet algorithme s'exécute dans un espace mémoire de taille $\mathcal{O}(n \log n)$ où n est la taille de la formule à étudier. Nous décrivons les choix d'implantation de ce système et les bonnes propriétés de complexité dérivées de la linéarité.

Dans la quatrième partie, nous présentons plusieurs sémantiques de la logique intuitionniste linéaire dont la complétude et la propriété de modèles finis sont prouvées. Elles viennent complé-

ter des études préalables sur l'utilisation de réseaux de Petri [Eng 97] comme base sémantique à la logique linéaire. Ces travaux pouvaient laisser croire en l'inadéquation des réseaux de Petri comme modèles complets de la logique linéaire, car seuls des résultats partiels de complétude dans des fragments de LLI étaient proposés. Nous montrerons qu'il n'en est rien. Nous présentons la sémantique algébrique, la sémantique des phases et une sémantique basée sur la notion de monoïde ordonné. La notion de plus grande prétopologie est mise en œuvre dans la définition d'une nouvelle sémantique basée sur la notion de ressource. Elle généralise toutes les sémantiques précédentes. Pour cette nouvelle sémantique, nous montrons la complétude avec une technique fondée sur la recherche de preuve et utilisant la notion d'arbre de réfutation. De plus, nous montrons qu'il existe une technique naturelle de transformation d'une réfutation finie en un contre-modèle fini sous la forme d'un espace de ressources. Nous en déduisons la propriété des modèles finis. Nous présentons la construction effective d'un contre-modèle d'une formule invalide. Les relations que nous avons établies entre les différentes sémantiques nous permettent de dériver les propriétés de complétude et de modèles finis pour les autres interprétations à partir de la sémantique des ressources. Nous en dérivons une nouvelle sémantique de la logique linéaire intuitionniste fondée sur les réseaux de Petri, qui, elle, est complète pour tout LLI. Nous la comparons à la sémantique préexistante et discutons la problématique de la construction automatique de contre-modèles.

Dans la conclusion, nous présentons un certain nombre de perspectives. Suite aux travaux et résultats sur la recherche de preuve dans LI, nous pouvons notamment envisager la généralisation des techniques de gestion des ressources et de partage à d'autres logiques intuitionnistes ainsi que les problèmes de réduction de la taille des contre-modèles. Dans le cadre de l'étude sur la sémantique de LLI, le problème de la construction automatique de contre-modèles nous amène à rechercher les formes les plus adaptées de représentation des contre-modèles. Les réseaux de Petri étant un bon modèle de LLI, il est également important d'étudier les conséquences de notre nouvelle sémantique du point de vue de la spécification et du raisonnement sur ces réseaux.

1 | Preuves et réfutations

Introduction

Cette première partie s'attache à définir un langage, des notations et quelques concepts fondamentaux de la recherche de preuves. Ils seront mis en œuvre en permanence pour aborder la recherche de preuves dans les logiques intuitionniste au chapitre 3 et intuitionniste linéaire au chapitre 4. Notre présentation sera très générale mais des exemples concrets viendront systématiquement illustrer les notions introduites. Le concept de départ de notre présentation est celui de proposition. De manière simple, on peut considérer qu'une proposition est une entité à laquelle on peut attacher une notion de validité logique. Nous décrirons des méthodes pour définir cette validité.

Nous introduisons les notions de système déductif puis de système déductif défini à partir d'un système de règles d'inférence. Ensuite, nous présentons la notion syntaxique fondamentale de preuve vue comme un arbre décoré par des propositions. Vient ensuite la notion moins répandue de réfutation qui est un concept syntaxique dual à celui de preuve.[4] Nous étudions sous quelles conditions l'existence d'une réfutation caractérise l'invalidité d'une proposition. Ceci nous conduit à mettre en évidence les notions de co-validité et d'analycité. Nous discutons également la problématique de la recherche et de la construction automatique de preuves ou de réfutations dans les systèmes analytiques et décrivons une méthodologie pour établir la co-validité de systèmes de règles.

1.1 Systèmes déductifs

Définition 1.1.1 (Système déductif) *Un **système déductif** est une paire $(\mathcal{P}, \Vdash)$ où $\mathcal{P}$ est un ensemble dont les éléments sont appelés **propositions** (logiques) et $\Vdash \subseteq \mathcal{P}$ une relation unaire[5] sur $\mathcal{P}$ appelée **relation de validité**.*

Nous écrirons $\Vdash P$ lorsque la proposition P est un élément de $\Vdash$. Dans le cas contraire, nous écrirons $\nVdash P$. Une proposition P vérifiant $\Vdash P$ est dite **valide** et est dite **invalide** si $\nVdash P$. Les propositions, qui resteront pour toute cette partie des éléments abstraits — mis à part dans les exemples à venir, — peuvent être soit des formules logiques dans le cas des systèmes de Hilbert [Tro 96], soit des séquents dans le cadre du calcul des séquents 3.1.2 soit encore d'autres structures plus complexes comme des arbres 3.5.1.

L'exemple le plus simple d'ensemble de propositions que l'on puisse imaginer est l'algèbre de Boole suivante : $(\{0,1\},\{1\})$ où $\nVdash 0$ et $\Vdash 1$. Un exemple plus complexe est donné en figure 1.1. Il y est décrit de manière récursive l'ensemble des **formules** de la logique propositionnelle en

[4] Cette dualité entre preuves et réfutations et la correspondance entre réfutations et contre-modèles sera au centre de nos réflexions.

[5] C'est-à-dire un sous-ensemble de $\mathcal{P}$.

Cst	$C ::= \mathsf{T} \mid \mathsf{F}$
Var	$V ::= X \mid Y \mid Z \mid \ldots$
Form	$F ::= V \mid C \mid F \wedge F \mid F \vee F \mid F \to F$

$$\neg A \triangleq A \to \mathsf{F}$$
$$A \leftrightarrow B \triangleq (A \to B) \wedge (B \to A)$$

FIG. 1.1 – Les formules de la logique propositionnelle.

notation BNF. Dans ce cas $\mathcal{P} = \mathsf{Form}$. Mais la structure des propositions peut être encore plus sophistiquée. Par exemple, dans le cadre du calcul des séquents, les propositions ne sont pas les formules mais les séquents. Un **séquent** (commutatif intuitionniste) est une paire (Γ, A) notée $\Gamma \vdash A$ où Γ est un multi-ensemble (une liste où l'ordre est ignoré) de formules et A est une formule, voir en section 2.3.1 pour une définition plus précise. Par exemple $A, A, A \to B \vdash C \wedge D$ est un séquent.

Décrire la relation de validité pour l'ensemble Form peut se faire de plusieurs manières. De manière **intensionnelle**, on peut donner un critère de validité en projetant un système déductif $\mathcal{P}$ sur un autre pour lequel la validité est déjà définie. Ainsi la sémantique de Boole consiste à projeter $\mathcal{P}$ sur $\{0, 1\}$ en associant une valeur de vérité à chaque formule. Nous décrirons ce procédé appelé interprétation plus en détails en section 2.6 chapitre 2.

On peut aussi définir la validité de manière **extensionnelle** en donnant une méthode pour construire les éléments valides. Ceci se fait généralement en donnant un système de règles qui permettent de déduire la validité de certaines formules (ou plus généralement de propositions) à partir d'autres dont la validité est soit acquise, soit déjà établie. Par exemple, en section 1.2 figure 1.2 nous décrivons un système de Hilbert pour définir la validité en logique intuitionniste.

La problématique générale en recherche de preuves est de partir d'une relation de validité définie de manière extensionnelle par un système de règles et d'en déduire un algorithme de décision efficace pour cette relation, donc une description intentionnelle, c'est-à-dire un nouveau critère de validité.

1.2 Règles logiques et systèmes

Pour toute la suite de cette partie, nous considérons un système déductif $(\mathcal{P}, \Vdash)$ fixé.

Définition 1.2.1 (Règle d'inférence) *Une **règle d'inférence n-aire** est un sous-ensemble $R \subseteq \mathcal{P}^{n+1}$ c'est-à-dire un ensemble de $(n+1)$-uplets $(H_1, \ldots, H_n, C) \in R$ appelés **instances** de la règle R.*

Nous utiliserons la notation $H_1, \ldots, H_n \blacktriangleright C\ [R]$ pour une instance de règle (dans le texte) ou bien la notation usuelle pour plus de lisibilité dans certains cas :

$$\frac{H_1 \quad \ldots \quad H_n}{C}\ [R]$$

Les propositions $H_1, \ldots, H_n$ sont appelées **prémisses** et C est appelée la **conclusion**. Le lecteur aura sans doute remarqué que nous ne donnons pas ici la définition la plus générale puisqu'il existe des systèmes où les instances de règles peuvent avoir un nombre arbitrairement grand de prémisses [Dyc 99]. Mais cette définition nous suffira pour cette thèse. Dans le cas particulier où $n = 0$, c'est-à-dire qu'il n'y a pas de prémisse, la règle est appelée **axiome**. Les autres règles — qui ont au moins une prémisse — sont appelées **règles de déduction**. On notera que nous ne ferons pas toujours la distinction entre **règle** et **instance de règle** par la suite.

L'application d'une règle consiste à déduire (la validité de) la conclusion C à partir (de la validité) des prémisses $H_1, \ldots, H_n$. Dans un processus d'analyse des propositions comme par exemple la recherche de preuves, **l'application inverse d'une règle** correspond aux remplacements de l'étude de la conclusion C par l'étude des prémisses $H_1, \ldots, H_n$. Pour le calcul des séquents par exemple, qui est basé sur le principe de la décomposition des formules, on choisit **une formule principale** que l'on décompose suivant la règle qui correspond au connecteur logique le plus extérieur de la formule principale et sa position dans le séquent — voir aussi section 3.1.2.

Définition 1.2.2 (Relation close, règle valide) *Une relation unaire $\succ \subseteq \mathcal{P}$ est dite **close** pour une règle $[R]$ si pour chacune de ses instances $H_1, \ldots, H_n \blacktriangleright C$ $[R]$, la relation $\succ C$ est vraie dès que toutes les relations $\succ H_i$ sont vraies. Une règle est dite **valide** si la relation de validité $\Vdash$ est close pour cette règle.*

Par exemple, on considère que la règle d'inférence la plus connue est celle du **modus ponens**,

$$\frac{A \to B \qquad A}{B} \text{ [mp]}$$

qui correspond à l'ensemble des instances $A \to B, A \blacktriangleright B$ [mp] où A et B peuvent prendre n'importe quelle valeur dans l'ensemble des formules Form. Cette règle est valide pour la logique classique — c'est-à-dire quand la validité $\Vdash$ est définie comme telle, — mais pour bien d'autres logiques aussi, comme les logiques intuitionnistes, modales...

Le lecteur pourra noter qu'on ne considère généralement que des règles valides, soit parce que la relation $\Vdash$ est définie à partir de ces mêmes règles — voir la définition 1.2.5, — soit parce que les règles ont été choisies de manière appropriée. Lorsqu'un axiome $\blacktriangleright C$ [Ax] est valide, la conclusion C de cet axiome[6] est valide, i.e. $\Vdash C$. Les axiomes sont des propositions considérées comme valides a priori.

Définition 1.2.3 (Prémisse et règle inversible) *Une **prémisse** i (pour $1 \leqslant i \leqslant n$) d'une règle n-aire $[R]$ est dite **inversible** si pour chaque instance $H_1, \ldots, H_n \blacktriangleright C$ $[R]$, la proposition H_i est valide ($\Vdash H_i$) dès que la conclusion C l'est ($\Vdash C$). Une **règle** est dite **inversible** si toutes ses prémisses le sont.*

Autrement dit, la prémisse H_i est inversible si et seulement si la règle $C \blacktriangleright H_i$ est valide. Par exemple, dans la règle du modus ponens $A \to B, A \blacktriangleright B$ [mp], la première prémisse $A \to B$ est inversible en logique classique ainsi qu'en logique intuitionniste. Ceci veut dire que pour que B soit valide, il faut au moins que $A \to B$ le soit. Par contre la deuxième prémisse ne l'est évidement pas — sinon n'importe quelle formule serait valide.

En logique classique ou intuitionniste la règle $A, B \blacktriangleright A \wedge B$ $[\wedge_R]$ est inversible car $A \wedge B$ est valide si et seulement si A et B le sont. Les prémisses et règles inversibles permettent des simplifications dans les arbres de réfutations (voir section 1.6.) Du point de vue de la recherche de preuves, elles permettent d'éviter le retour arrière (ou back-tracking, voir section 3.5 chapitre 3.)

Par contre la règle $[\to_L]$ du système G2i présenté un figure 1.3 :

$$\frac{\Gamma, A \to B \vdash A \qquad \Gamma, B \vdash C}{\Gamma, A \to B \vdash C} \text{ } [\to_L]$$

[6] Plutôt faudrait-il dire instance d'axiome.

Le système Hi

$$\frac{A \to B \qquad A}{B}\ [\text{mp}]$$

$$[\wedge\text{-Ax}]\quad \begin{cases} (A \wedge B) \to A, (A \wedge B) \to B, \\ A \to B \to (A \wedge B) \end{cases}$$

$$[\vee\text{-Ax}]\quad \begin{cases} A \to (A \vee B), B \to (A \vee B), \\ (A \to C) \to (B \to C) \to (A \vee B) \to C \end{cases}$$

$$[\to\text{-Ax}]\quad \begin{cases} A \to B \to A, \\ (A \to B \to C) \to (A \to B) \to A \to C \end{cases}$$

$$[\mathsf{Cst}\text{-Ax}]\quad \mathsf{F} \to A, A \to \mathsf{T}$$

FIG. 1.2 – Système de Hi Hilbert pour la logique intuitionniste, LI.

Le système G2i

$$\frac{}{\Gamma, \boxed{A} \vdash \boxed{A}}\ [\text{Ax}]$$

$$\frac{}{\Gamma, \boxed{\mathsf{F}} \vdash C}\ [\mathsf{F}_L] \qquad \frac{}{\Gamma \vdash \boxed{\mathsf{T}}}\ [\mathsf{T}_R]$$

$$\frac{\Gamma \vdash C}{\Gamma, \boxed{\mathsf{T}} \vdash C}\ [\mathsf{T}_L] \qquad \frac{\Gamma \vdash A}{\Gamma \vdash \boxed{A \vee B}}\ [\vee^1_R]$$

$$\frac{\Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma, \boxed{A \vee B} \vdash C}\ [\vee_L] \qquad \frac{\Gamma \vdash B}{\Gamma \vdash \boxed{A \vee B}}\ [\vee^2_R]$$

$$\frac{\Gamma, A, B \vdash C}{\Gamma, \boxed{A \wedge B} \vdash C}\ [\wedge_L] \qquad \frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash \boxed{A \wedge B}}\ [\wedge_R]$$

$$\frac{\Gamma, A \to B \vdash A \qquad \Gamma, B \vdash C}{\Gamma, \boxed{A \to B} \vdash C}\ [\to_L] \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash \boxed{A \to B}}\ [\to_R]$$

FIG. 1.3 – Calcul des séquents G2i pour la logique intuitionniste, LI.

n'est pas inversible en sa première prémisse. Les règles $[\vee^1_R]$ et $[\vee^2_R]$ ne sont pas inversibles non plus. Au chapitre 3, nous étudierons les conséquences de ces non-inversibilités au niveau de la construction automatique des preuves.

Définition 1.2.4 (Système de règles) *Un **système de règles** est un ensemble de règles. Un système est **valide** si toutes ses règles le sont.*

Par exemple, la figure 1.2 décrit un système de règles valides pour la logique intuitionniste. Il comprend une seule règle de déduction, la règle du modus ponens [mp]. Ces systèmes où les propositions sont des formules et où seule la règle du modus ponens n'est pas un axiome sont appelés systèmes à la Hilbert. Encore une fois il y a des exceptions, par exemple dans le cadre des logiques modales, la règle $F \blacktriangleright \Box F$. Mais l'idée générale des systèmes de Hilbert est de minimiser les règles de déduction.

Un autre exemple de système de règles valides pour les séquents intuitionnistes est G2i donné en figure 1.3. Il s'agit ici du calcul sans règles structurelles (contraction ou affaiblissement) ni coupure. De nombreuses variantes de ce système existent, dont certaines seront présentées dans le chapitre 3. Ce système illustre la notion de formule principale, encadrée dans chaque règle, qui est la formule sur laquelle la règle agit. Il y a les règles droites de suffixe R, et les règles gauches de suffixe L qui correspondent à la position de la formule principale par rapport au séparateur $\vdash$.

Proposition 1.2.1 *Soit $\mathcal{S}$ un système de règles. Il existe une plus petite relation notée $\vdash_{\mathcal{S}}$ close pour les règles de $\mathcal{S}$.*

Démonstration. On peut décrire $\vdash_{\mathcal{S}}$ de manière intentionnelle comme l'intersection de toutes les relations closes pour les règles de $\mathcal{S}$, i.e. $\vdash_{\mathcal{S}} \triangleq \bigcap\{\succ \mid \succ \text{ est close pour } \mathcal{S}\}$. Il suffit alors de montrer que la relation $\vdash_{\mathcal{S}}$ ainsi définie est close pour toutes les règles de $\mathcal{S}$, car dans ce cas, elle sera forcément la plus petite. Soit donc $H_1, \ldots, H_n \blacktriangleright C$ $[R]$ une instance d'une règle $R \in \mathcal{S}$. Si on suppose que pour tout i on a $\vdash_{\mathcal{S}} H_i$ alors étant donné une relation $\succ$ arbitraire close pour $\mathcal{S}$, on a $\vdash_{\mathcal{S}} \subseteq \succ$ par définition de $\vdash_{\mathcal{S}}$. Donc pour tout i, on a aussi $\succ H_i$. Or $\succ$ est en particulier close pour la règle $[R]$ et donc on a $\succ C$. Ceci pour n'importe quel choix de $\succ$ close pour $\mathcal{S}$. Ainsi, par définition de $\vdash_{\mathcal{S}}$, on a aussi $\vdash_{\mathcal{S}} C$. La relation $\vdash_{\mathcal{S}}$ est donc close pour la règle $[R]$. □

Corollaire 1.2.2 *Si $\mathcal{S}$ est un système de règles valides alors $\vdash_{\mathcal{S}}$ est plus forte que $\Vdash$, i.e. $\vdash_{\mathcal{S}} \subseteq \Vdash$.*

Démonstration. Dans le cas d'un système valide, la relation de validité $\Vdash$ est close pour ce système, et donc plus grande que la plus petite d'entre elles, c'est-à-dire $\vdash_{\mathcal{S}}$. □

Définition 1.2.5 (Système complet) *Un système de règles $\mathcal{S}$ est dit **complet** pour le système déductif $(\mathcal{P}, \Vdash)$ si $\vdash_{\mathcal{S}}$ et $\Vdash$ sont égales, i.e. $\vdash_{\mathcal{S}} = \Vdash$.*

Par exemple, le système de Hilbert Hi donné en figure 1.2 est complet pour la logique intuitionniste [Tro 88]. Il ne l'est donc évidement pas pour la logique classique.

Différentes approches sont possibles pour obtenir des systèmes complets. La plus triviale est de poser $\Vdash \triangleq \vdash_{\mathcal{S}}$ c'est-à-dire de définir la validité à partir d'un système de règles, et c'est peut-être le cas le plus fréquent. On peut par exemple considérer que la logique linéaire a été définie à partir du calcul de séquent, en supprimant les règles structurelles du calcul des séquents classique, voir section 4.1. Si la validité est déjà définie par une sémantique par exemple, alors on peut essayer de raffiner un système en ajoutant des règles valides jusqu'à obtenir la complétude. Ce processus de modélisation n'est pas toujours facile à réaliser.

1.3 Arbres de preuve

Dans toute cette partie on suppose fixé un système déductif $(\mathcal{P}, \Vdash)$ ainsi qu'un système de règles $\mathcal{S}$ valide pour $\mathcal{P}$, et ainsi $\vdash_{\mathcal{S}} \subseteq \Vdash$.

Définition 1.3.1 (Arbre de preuve) *Un **arbre de preuve** est un arbre **fini** dont les nœuds sont indexés par des propositions de $\mathcal{P}$ et vérifiant la propriété suivante : pour tout nœud f d'index C, il existe une instance de règle $H_1, \ldots, H_n \blacktriangleright C$ $[R]$ ayant C comme conclusion et telle que le nœud f ait n fils indexés respectivement par les prémisses $H_1, \ldots, H_n$.*

$$\dfrac{\dfrac{}{A \to B \to A}\,[\to\text{-Ax}] \quad \dfrac{\dfrac{}{A \to C \to A}\,[\to\text{-Ax}] \quad \dfrac{}{(A \to C \to A) \to (A \to C) \to A \to A}\,[\to\text{-Ax}]}{(A \to B \to A) \to A \to A}\,[\text{mp}]}{A \to A}\,[\text{mp}]$$

FIG. 1.4 – Un arbre de preuve dans un système de Hilbert, avec $C \equiv B \to A$.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{B, A, (A \wedge B) \to C \vdash A}\,[\text{Ax}] \quad \dfrac{}{B, A, (A \wedge B) \to C \vdash B}\,[\text{Ax}]}{B, A, (A \wedge B) \to C \vdash A \wedge B}\,[\wedge_R] \quad \dfrac{}{B, A, C \vdash C}\,[\text{Ax}]}{B, A, (A \wedge B) \to C \vdash C}\,[\to_L]}{B \wedge A, (A \wedge B) \to C \vdash C}\,[\wedge_L]}{(A \wedge B) \to C \vdash (B \wedge A) \to C}\,[\to_R]$$

FIG. 1.5 – Un arbre de preuve dans un calcul des séquents.

Si on considère le système de règles Hi en figure 1.2, la figure 1.4 représente un arbre de preuve dans ce système de règles. On pourra remarquer que le nom de chaque règle est rapporté sur la figure bien que cela ne soit pas nécessaire d'après la définition. On remarque que les feuilles d'un arbre de preuve, n'ayant pas de fils, ne peuvent être indexées que par des (conclusions d')axiomes.

Un autre exemple de preuve, dans le calcul des séquents G2i cette fois, est présenté en figure 1.5. Par rapport au système Hi de Hilbert, où la règle [mp] possède une infinité d'instances pour une conclusion donnée, on peut remarquer que dans le calcul de séquents, il n'y a qu'un nombre fini d'instances de règles possibles pour un séquent de conclusion donné, ce qui est un des deux critères d'analycité, voir section 1.5.3.

Proposition 1.3.1 *Si $\mathcal{T}$ est un arbre de preuve et P est une proposition indexant l'un des nœuds de $\mathcal{T}$ alors P est valide, i.e. $\Vdash P$.*

Démonstration. On raisonne par induction sur la taille de l'arbre de preuve. On considère la règle

$$\dfrac{H_1(f_1) \quad \cdots \quad H_n(f_n)}{C(r)}$$

qui correspond à la racine r de l'arbre de preuve. La notation $H(f)$ veut dire que le nœud f est indexé par la proposition H. Si cette règle est un axiome $n = 0$ alors l'arbre est réduit à une feuille et C est valide car cet axiome est valide. Sinon on considère les n sous-arbres dont les racines sont les fils $f_1, \ldots, f_n$ de la racine r. Il est clair que chaque sous-arbre est un arbre de preuve et donc, par hypothèse d'induction, leurs nœuds sont indexés par des propositions valides. En particulier, les prémisses $H_1 \ldots, H_n$ sont des propositions valides. Comme la règle $H_1, \ldots, H_n \blacktriangleright C$ est valide, la proposition C est valide elle aussi. Ainsi tous les index des nœuds sont valides. □

Donc les arbres de preuve permet de construire des propositions valides. Nous nous intéressons maintenant à la réciproque. À quelle condition une formule valide admet-elle un arbre de preuve ?

Définition 1.3.2 (Preuve) *On appelle **preuve** d'une proposition P tout arbre de preuve dont P indexe la racine.*

Ainsi s'il existe une preuve de P alors cette proposition est valide $\Vdash P$. Mais peut-on fabriquer toutes les propositions valides avec des preuves ? Dans le cadre d'un système complet, la réciproque est vraie aussi.

Proposition 1.3.2 *Dans un système $\mathcal{S}$ complet — i.e. $\vdash_{\mathcal{S}} = \Vdash$, — si P est une proposition valide alors il existe une preuve de P.*

Démonstration. Nous définissons la relation unaire $\succ$ par

$$\succ P \text{ si et seulement si il existe une preuve de } P$$

Cette relation est close pour les règles du système $\mathcal{S}$. En effet, soit $H_1, \ldots, H_n \blacktriangleright C$ une instance de règle dont on suppose que les prémisses H_i vérifient $\succ H_i$. Alors il existe pour chaque i une preuve de H_i, c'est-à-dire un arbre de preuve $\mathcal{T}_i$ dont H_i indexe la racine. On considère alors l'arbre de racine r indexé par C dont les fils sont les racines des arbres $\mathcal{T}_i$.

$$\frac{\dfrac{\cdots}{\mathcal{T}_1(H_1)} \quad \cdots \quad \dfrac{\cdots}{\mathcal{T}_n(H_n)}}{\mathcal{T}(C)}$$

Nous obtenons un nouvel arbre de preuve dont la racine est indexée par C. Et ainsi $\succ C$. La relation $\succ$ est donc close pour n'importe quelle instance de règle. Donc d'après la proposition 1.2.1, on a $\Vdash = \vdash_{\mathcal{S}} \subseteq \succ$. □

1.4 Arbres de réfutation

Dans cette partie, on se donne un système déductif $(\mathcal{P}, \Vdash)$ et un système de règles $\mathcal{S}$ valide. On ne suppose pas a priori que ce système soit complet.

1.4.1 Définition

La notion d'arbre de réfutation est une notion syntaxique duale à celle de preuve, contrairement à la notion de contre-modèle qui est une notion sémantique duale à celle de preuve. L'un des thèmes importants développés dans cette thèse est qu'il existe des liens étroits entre les réfutations et les contre-modèles — voir section 3.3.4 chapitre 3 et section 4.4.9 chapitre 4.

Définition 1.4.1 (Arbre de réfutation) *Un **arbre de réfutation** est un arbre **fini ou infini** dont les nœuds sont indexés par des propositions de $\mathcal{P}$ et vérifiant la propriété suivante : pour tout nœud f d'index C, il existe une correspondance (bijective) entre les fils s de f et les instances de règles $H_1^s, \ldots, H_{n_s}^s \blacktriangleright C$ $[R_s]$ ayant C comme conclusion. Chaque fils s de f est indexé par l'une des prémisses H_i^s de l'instance $[R_s]$. Un **arbre de réfutation partiel** ne doit vérifier la condition précédente que pour ses nœuds internes, aucune condition n'étant imposée sur ses feuilles.*

Il est fondamental de comprendre que la condition sur les nœuds d'un arbre de réfutation est la duale de la condition sur les nœuds des preuves. Dans une preuve, chaque nœud correspond au choix d'une règle et les fils sont les prémisses de cette règle. Dans une réfutation, on prend toutes les règles possibles mais on choisit une prémisse par règle.

Les arbres de réfutation peuvent être infiniment larges (un nœud peut avoir un nombre infini de fils,) comme c'est par exemple le cas pour le système de Hilbert en figure 1.2 car la règle $A \to B, A \blacktriangleright B$ [mp] possède une infinité d'instances à B fixé. Ils peuvent aussi être infiniment hauts (i.e. ils peuvent avoir des branches infinies) et c'est aussi le cas du système de Hilbert Hi. Ce système n'est donc pas bien adapté à la construction d'arbres de réfutation. Le calcul des séquents permet quant à lui de définir des systèmes où les arbres de réfutation sont finis. En considérant le calcul des séquents G2i pour la logique intuitionniste, voici un exemple d'arbre de réfutation :

$$\dfrac{\dfrac{\overline{A \vdash B}}{A \vdash A \wedge B} \quad \dfrac{\overline{B \vdash A}}{A \vee B \vdash A}}{A \vee B \vdash A \wedge B}$$

Soit $\mathcal{R}$ un arbre de réfutation. Nous écrivons $P \in \mathcal{R}$ lorsque P est l'index de l'un des nœuds de $\mathcal{R}$ et $P \notin \mathcal{R}$ sinon. Nous définissons la relation $\succ_{\mathcal{R}}$ par « $\succ_{\mathcal{R}} P$ si et seulement si $P \notin \mathcal{R}$, » qu'on notera aussi $\succ_{\mathcal{R}} = (\cdot) \notin \mathcal{R}$.[7]

Proposition 1.4.1 *Soit $\mathcal{S}$ un systèmes de règles et $\mathcal{R}$ un arbre de réfutation pour ce système. Alors la relation $\succ_{\mathcal{R}} = (\cdot) \notin \mathcal{R}$ est close pour les règles de $\mathcal{S}$.*

Démonstration. En effet soit $H_1, \ldots, H_n \blacktriangleright C$ $[R]$ une instance de règle. On suppose que pour chaque prémisse on a $\succ_{\mathcal{R}} H_i$, i.e. $H_i \notin \mathcal{R}$. On veut montrer que l'on a $\succ_{\mathcal{R}} C$. Si C devait être l'index de l'un des nœuds r de $\mathcal{R}$ alors r aurait un fils correspondant à l'instance de règle [R]. Ce fils serait alors indexé par l'un des H_i ce qui contredit $H_i \notin \mathcal{R}$. Donc on a aussi $\succ_{\mathcal{R}} C$. □

1.4.2 Réfutations

Définition 1.4.2 (Réfutation) *Une **réfutation** d'une proposition P est un arbre de réfutation dont l'index de la racine est P.*

Théorème 1.4.2 *Si P est invalide ($\nVdash P$) alors P admet une réfutation.*

Démonstration. Le principe de la preuve consiste en la construction d'une suite croissante d'arbres de réfutation partiels indexés par des propositions invalides, dont la limite sera une réfutation de P.

Étant donné un arbre de réfutation partiel $\mathcal{A}$ dont tous les index sont invalides, on construit un nouvel arbre de réfutation partiel $\mathbb{F}(\mathcal{A})$ en rajoutant des fils aux feuilles de $\mathcal{A}$. Pour chaque feuille f indexée par la proposition C invalide $\nVdash C$, on lui ajoute autant de fils que d'instances de règle $H_1^R, \ldots, H_{n_R}^R \blacktriangleright C$ $[R]$ ayant C comme conclusion. Comme C est invalide et que la règle $[R]$ est valide, il s'en suit que l'un des H_i^R est invalide. Soit i_R l'indice du premier d'entre-eux. Ainsi $\nVdash H_{i_R}^R$ et on indexe le fils correspondant à la règle $[R]$ par $H_{i_R}^R$. Alors le nœud f devient un nœud interne mais vérifie en même temps la condition imposée aux nœuds internes des arbres de réfutation partiels. $\mathbb{F}(\mathcal{A})$ est donc un arbre de réfutation partiel. De plus tous ses index sont invalides.

Soit P une proposition invalide ($\nVdash P$) et $\mathcal{A}_0$ l'arbre de refutation partiel constitué d'une unique feuille indexée par P. On définit la suite $\mathcal{A}_n \triangleq \mathbb{F}^n(\mathcal{A}_0)$ et sa limite $\mathcal{A} \triangleq \bigcup_n \mathcal{A}_n$. On montre que $\mathcal{A}$ est un arbre de réfutation. En effet, si f est un nœud interne de $\mathcal{A}$ alors c'est un nœud interne

[7] Cette notation est expliquée au chapitre 2 section 2.1.

de l'un des $\mathcal{A}_n$ et par conséquent, ses fils correspondent aux règles qui peuvent appliquées « à l'envers » à son index. Si par contre f est une feuille d'index C, soit n_0 un index pour lequel $f \in \mathcal{A}_{n_0}$, alors par construction de $\mathcal{A}_{n_0+1}$, si f n'a pas de fils dans cet arbre, c'est parce qu'aucune règle ne possède C pour conclusion. Ainsi, f peut légitimement être une feuille d'un arbre de réfutation. Pour finir, $\mathcal{A}$ est donc un arbre de réfutation et P indexe sa racine. □

Le lecteur pourra remarquer que cette démonstration ne définit pas d'algorithme de construction d'arbres de réfutation dans la mesure où d'une part, il peut très bien arriver que l'on obtienne un arbre infini, soit en largeur, soit en hauteur, et d'autre part, qu'on ne possède pas forcément d'algorithme de décision d'(in)validité.

1.4.3 Notion de co-validité

Dans une preuve, les index sont tous des propositions valides, à condition bien sûr que les règles logiques soient valides — voir la proposition 1.3.1. Que peut-on dire des arbres de réfutations : ne contiennent-ils que de propositions invalides ?

Proposition 1.4.3 *Si $\mathcal{S}$ est un système complet et P admet une réfutation alors P est invalide ($\nVdash P$).*

Démonstration. Si P admet une réfutation $\mathcal{R}$ alors comme $P \in \mathcal{R}$ et $\vdash_{\mathcal{S}} = \Vdash \subseteq \succ_{\mathcal{R}} = (\cdot) \notin \mathcal{R}$ d'après la proposition 1.4.1, on a $\nVdash P$. □

Mais que se passe-t-il dans le cas d'un système où l'on suppose seulement la validité, pas la complétude ?

Définition 1.4.3 (Co-validité) *Le système $\mathcal{S}$ est dit **co-valide** si pour toute proposition C la condition suivante est vérifiée : si (pour toute instance de règle $H_1, \ldots, H_n \blacktriangleright C$ $[R]$ ayant comme conclusion C, l'une des prémisses H_i est invalide ($\nVdash H_i$)) alors C est invalide ($\nVdash C$).*

Proposition 1.4.4 *Un système complet est co-valide.*

Démonstration. Dans un système complet, l'hypothèse de la condition dans la définition 1.4.3 correspond au fait qu'aucune règle ne peut achever une preuve de C. C n'ayant pas de preuve est invalide. □

La co-validité est donc, comme la validité, une condition plus faible que la complétude. Nous verrons en section 1.5.4 comment la conjonction des deux peut, dans certains cas, suffire à montrer la complétude.

Proposition 1.4.5 *Si $\mathcal{S}$ est co-valide et $\mathcal{R}$ un arbre de réfutation **fini** alors toute proposition P qui est un index dans $\mathcal{R}$ est invalide ($\nVdash P$).*

Démonstration. Par l'absurde, on considère une proposition C, index de $\mathcal{R}$, valide ($\Vdash C$) de hauteur maximale. Dans ce cas, par maximalité, les index de tous ses fils sont invalides. Par définition d'un arbre de réfutation, les index des fils correspondent chacun à une prémisse d'une des règles ayant C comme conclusion, et tous ses index sont invalides. Donc par co-validité, C est invalide, ce qui est absurde d'où le résultat. □

Les réfutations finies permettent donc de construire des propositions invalides. Mais peut-on toutes les construire ainsi. Qu'en est-il des réfutations infinies ?

$$\frac{}{2n}\ [R_1] \qquad \frac{n+2}{n}\ [R_2] \qquad \cfrac{\cfrac{\cfrac{\cdots}{5}}{3}}{1}$$

FIG. 1.6 – Arbre de réfutation infini.

Proposition 1.4.6 *Il existe des arbres de réfutations infinis où toutes les propositions (les index) sont valides.*

Démonstration. On considère le système de déduction $(\mathbb{N}, \Vdash)$ où $\Vdash n$ est vraie pour n'importe quel entier n. On considère le système $\mathcal{S}$ constitué des deux règles en figure 1.6 où n est un entier arbitraire. Dans ce cas, on a l'équivalence

$$\vdash_{\mathcal{S}} n \text{ est vraie si et seulement si } n \text{ est pair}$$

et donc, $\mathcal{S}$ n'est pas complet. Par contre les deux règles sont à la fois valides et co-valides. L'arbre de réfutation infini décrit en figure 1.6 ne contient que des propositions valides. □

La condition de co-validité ne peut donc suffire quand il s'agit de fabriquer des propositions invalides. Nous verrons en section 1.5.3 que la condition d'analycité qui impose aux réfutations d'être des arbres finis permet d'assurer la complétude.

1.5 Algorithme de recherche de preuves

L'idée qui consiste à essayer d'appliquer toutes les règles possibles à l'envers, c'est-à-dire à partir d'une proposition C rechercher toutes les instances de règles de la forme $H_1, \ldots, H_n \blacktriangleright C$ et recommencer de manière récursive sur les propositions H_i permet de définir un algorithme de construction de preuves ou de réfutations. Bien sûr, cet algorithme ne termine que si des conditions supplémentaires sont vérifiées par le système de règles $\mathcal{S}$, conditions que nous détaillerons plus loin.

1.5.1 Description de l'algorithme

On définit une fonction récursive φ qui prend en paramètre une proposition C (valide ou invalide) et renvoie en sortie, soit une preuve de C, soit une réfutation de C, si la fonction φ termine. Pour chaque instance de règle $H_1, \ldots, H_n \blacktriangleright C\ [R]$, on applique récursivement φ sur chacun des H_i et on pose $\mathcal{A}_i^R \triangleq \varphi(H_i)$. Deux cas se présentent :

1. soit il existe une règle $[R]$ telle que chaque $\mathcal{A}_1^R, \ldots, \mathcal{A}_n^R$ soit une preuve de respectivement $H_1, \ldots, H_n$ et dans ce cas on renvoie la preuve de C suivante :

$$\frac{\mathcal{A}_1^R \quad \ldots \quad \mathcal{A}_n^R}{C}\ [R]$$

2. soit pour chaque règle $[R]$, l'un au moins des $\mathcal{A}_i^R$ est une réfutation de H_i et soit i_R l'index de l'un d'entre eux (par exemple le premier.) Dans ce cas, on renvoie la réfutation de C suivante :

$$\frac{\mathcal{A}_{i_0}^{R_0} \quad \dots \quad \mathcal{A}_{i_k}^{R_k}}{C}$$

Théorème 1.5.1 *Si l'algorithme φ termine sur le paramètre C alors son résultat est soit une preuve, soit une réfutation finie de C.*

1.5.2 Le problème de la terminaison

L'algorithme φ décrit en section 1.5.1 peut très bien ne pas se terminer et ce pour deux raisons. Tout d'abord, il est possible que pour une proposition donnée C, un nombre infini d'instances de règles de la forme $H_1, \dots, H_n \blacktriangleright C$ existe. C'est par exemple le cas pour la règle du modus ponens $A \to C, A \blacktriangleright C$ [mp] où A peut prendre une valeur arbitraire. Dans le cas d'un système à la Hilbert, l'algorithme φ ne termine pas car l'arbre de recherche est à **branchement infini**, entre autres.

Mais il est aussi possible que cet algorithme ne termine pas parce que la fonction φ est appelée un nombre infini de fois de façon récursive. Considérons par exemple l'instance suivante de la règle $[\to_L]$ du calcul des séquents G2i décrit en figure 1.3 :

$$\frac{A \to B \vdash A \qquad B \vdash A}{A \to B \vdash A}\ [\to_L]$$

Dans le cas d'un appel $\varphi(A \to B \vdash A)$, la fonction φ va se rappeler un nombre infini de fois sur ce même argument et ne jamais terminer.[8] Dans le cas de ce calcul des séquents, l'algorithme peut ne pas se terminer car l'arbre de recherche admet **une branche infinie**.

1.5.3 Notion de système analytique

Nous introduisons maintenant deux conditions qui, combinées, suffisent à montrer la terminaison de l'algorithme φ de recherche de preuve. En fait elles garantissent que l'espace de recherche de preuve est fini.

Définition 1.5.1 (Système de règles analytique) *Un système de règles est dit **analytique** s'il vérifie les deux conditions suivantes :*

1. *Pour toute proposition C, il n'existe qu'un nombre fini (et calculable) d'instances de règles $H_1, \dots, H_n \blacktriangleright C$ ayant C pour conclusion.*
2. *Il existe un ordre strict bien fondé $<$ sur $\mathcal{P}$ tel que pour toute instance de règle $H_1, \dots, H_n \blacktriangleright C$ et chaque prémisse H_i, on ait $H_i < C$.*

Théorème 1.5.2 *Si $\mathcal{S}$ est un système analytique alors l'algorithme φ termine.*

Démonstration. La preuve se fait par induction sur C pour l'ordre strict bien fondé $<$. Comme φ est appelée sur un nombre fini de prémisses d'un nombre fini (calculable) d'instances de règles, φ est appelée un nombre fini de fois. Chaque appel se fait sur une prémisse H qui est strictement plus petite que C, et donc, par hypothèse d'induction sur $<$, cet appel $\varphi(H)$ se termine. Cet ensemble fini d'appels effectué, l'appel $\varphi(C)$ peut renvoyer son résultat. □

[8] Dans ce cas précis, on peut imaginer « améliorer » l'algorithme car la deuxième prémisse est réfutable, mais si on prend $B \equiv A$ alors, on ne peut plus rien dire.

En guise d'exemple de système analytique, le calcul G4ip alias LJT [Dyc 92] peut-être cité. Des versions améliorées de ce calcul seront présentées au chapitre 3. De manière générale, les calculs de séquents relatifs à la logique linéaire (sans les exponentielles) sont aussi analytiques, voir chapitre 4 section 4.1.3.

Proposition 1.5.3 *Si $\mathcal{S}$ est un système analytique alors ses arbres de réfutations sont finis.*

Démonstration. Il s'agit la d'une application du lemme de König. D'après la première condition, les arbres de réfutations sont à branchement fini. D'après la deuxième condition, ils ne possèdent pas de branche infinie. Donc par le lemme de König, ils sont finis. □

1.5.4 Analycité et complétude

Nous introduisons maintenant un résultat intéressant qui permet de montrer la complétude d'un système de règles sans passer par le processus usuel de traduction mutuelle vers un autre système dont la complétude est connue. Il suffit de montrer la co-validité d'un système analytique valide pour en obtenir la complétude.

Théorème 1.5.4 *Un système analytique, valide et co-valide est complet.*

Démonstration. Soit $\mathcal{S}$ un système analytique. On a déjà $\vdash_{\mathcal{S}} \subseteq \Vdash$ par validité de $\mathcal{S}$. Soit C une proposition valide ($\Vdash C$) quelconque. Montrons l'algorithme φ de recherche de preuves renvoit une preuve de C. Soit $\mathcal{A} \triangleq \varphi(C)$ la réponse de l'algorithme φ sur l'entrée C, qui existe d'après le théorème 1.5.2. Comme le système $\mathcal{S}$ est co-valide, si $\mathcal{A}$ est une réfutation finie alors C est invalide, d'après le résultat 1.4.5. Donc comme C est valide, alors $\mathcal{A}$ n'est pas une réfutation finie, c'est donc une preuve de C. Ainsi on obtient $\vdash_{\mathcal{S}} C$. Ceci pour toute proposition valide C. On peut en conclure $\Vdash \subseteq \vdash_{\mathcal{S}}$. □

1.6 Preuves de co-validité

Dans cette section nous présentons une technique qui permet de simplifier la preuve de co-validité d'un système, en se concentrant uniquement sur les propositions irréductibles, c'est-à-dire celles auxquelles aucune règle inversible ne peut être appliquée. En combinaison avec le théorème 1.5.4, ils permettent d'établir plus facilement la preuve de complétude d'un système analytique. On suppose que l'on dispose d'un système de règles valides.

Définition 1.6.1 (Propositions atomiques, réductibles et irréductibles) *Un proposition C est dite **réductible** s'il existe une instance de règle inversible de la forme $H_1, \ldots, H_n \blacktriangleright C$ $[R]$. Sinon la proposition est dite **irréductible**. Parmi les propositions irréductibles, il y a des propositions qui ne sont la conclusion d'aucune règle : elles sont appelées **atomiques**.*

Par exemple, le séquent $\Gamma, A \vdash A$ est réductible car c'est la conclusion de l'axiome [Ax] qui est bien sûr inversible. Le séquent $\Gamma \vdash A \to B$ est lui aussi réductible car la règle $[\to_R]$ est inversible. Dans le calcul des séquents G2i, les séquents atomiques sont de la forme, $V_1, \ldots, V_n \vdash W$ où les V_i sont des variables et W est une variable distincte des V_i ou bien la constante F. Enfin, le séquent $A \to B \vdash C \vee B$ est irréductible car seules les règles $[\to_L]$ et $[\vee_R]$ peuvent lui être appliquées et aucune de ces deux règles n'est inversible.

Si on considère la condition de co-validité pour une proposition C, on peut déjà traiter le cas atomique et le cas réductible et simplifier le cas irréductible :

Si C est atomique alors la condition de co-validité signifie simplement que la proposition C est invalide $\not\Vdash C$;

Si C est réductible alors la condition de co-validité est forcément vérifiée. En effet, on prend l'une des règles inversibles qui s'applique à C, soit $H_1, \ldots, H_n \blacktriangleright C$ $[R]$. Par hypothèse, l'une des prémisses H_i est invalide $\not\Vdash H_i$. Comme la règle $[R]$ est inversible, C est aussi invalide $\not\Vdash C$;

Si C est irréductible alors il suffit d'être capable de montrer qu'à partir d'un choix quelconque d'une prémisse invalide par règle applicable, il est possible de montrer l'invalidité de C. Pour ce choix de prémisse, il est inutile de prendre une prémisse inversible car dans ce cas, l'invalidité de C est acquise.

Principe 1.6.1 *Pour vérifier la co-validité d'un système $\mathcal{S}$, outre l'invalidité des propositions atomiques, il suffit de vérifier que pour toute proposition irréductible C, on peut déduire l'invalidité de C à partir d'un choix quelconque d'une prémisse invalide par règle applicable à C.*

Si ce résultat paraît un peu abstrait, on se référera à la preuve de complétude du système G4ip en section 3.3.4 chapitre 3 et on comparera sa complexité à celle de la preuve initiale de complétude [Dyc 92].

Le cas de la logique classique est dégénéré mais on peut très bien l'analyser à travers de la notion de co-validité. En effet, le calcul des séquents classique sans coupure et sans règles structurelles est analytique et **toutes les règles sont inversibles**. Par conséquent, il n'y a pas de séquents irréductibles autres que les séquents atomiques. Ceci veut dire que l'invalidité d'un séquent peut se déduire de l'une des feuilles d'un arbre de réfutation. On peut d'ailleurs fabriquer un contre-modèle classique à partir de l'une de ces feuilles. Par exemple la réfutation suivante du séquent $A \vee B \vdash A \wedge B$:

$$\dfrac{\dfrac{\dfrac{}{A \vdash B}}{A \vdash A \wedge B} \quad \dfrac{\dfrac{}{B \vdash A}}{A \vee B \vdash A}}{A \vee B \vdash A \wedge B}$$

nous fournit le contre-modèle $(A = 1, B = 0)$ à la première feuille et $(A = 0, B = 1)$ pour la deuxième.

Dans les cas de la logique intuitionniste et de la logique intuitionniste linéaire, la non-inversibilité de certaines règles fait qu'il ne suffit pas de considérer uniquement une feuille d'une réfutation pour construire un contre-modèle, voir section 3.4.4 chapitre 3 et section 4.4.9 chapitre 4.

Conclusion

Après avoir rappelé les notions de base de la théorie de la preuve, nous introduisons le concept moins répandu de réfutation, dual syntaxique de celui de preuve. Nous montrons que dans un système analytique, il existe un algorithme qui construit soit une preuve, soit une réfutation d'une proposition donnée. En fait, l'analycité est le critère qui garantit la terminaison de l'algorithme. Prouver la complétude d'un système analytique se réduit alors à montrer la validité et la co-validité des règles d'inférences. Les concepts de propositions (ir-)réductibles et atomiques permettent une approche simplifiée de la preuve de co-validité.

Nous nous sommes limités ici aux aspects syntaxiques et combinatoires de la logique. Nous allons maintenant nous intéresser aux aspects sémantiques, c'est-à-dire aux structures algébriques qui permettent une interprétation des propositions dans des modèles. Dans les chapitres 3 et 4 nous pourrons alors expliquer comment le concept de réfutation se compare à celui de contre-modèle, ces notions étant toutes les deux des critères pour invalider les propositions logiques.

2 | Sémantique

Introduction

Dans cette partie, nous présentons les structures algébriques fondamentales qui permettent de construire des modèles des logiques intuitionniste et intuitionniste linéaire. Ceci nous donne également l'occasion de fixer le vocabulaire sémantique. Notre approche est fondée sur la théorie de l'ordre et la théorie des treillis. Pour une bonne introduction à ces notions, le lecteur pourra consulter les ouvrages suivants [Bir 67, Dav 90].

Nous abordons le problème de la construction des modèles par complétion d'un ensemble ordonné enrichi, par exemple, d'une structure complémentaire de monoïde. Par complétion, nous entendons le processus qui consiste à étendre la structure d'ordre en structure de treillis, c'est-à-dire ajouter des bornes supérieures et inférieures, en essayant de préserver le plus possible les bornes qui préexistaient car elles représentent des informations sémantiques que nous ne voulons pas perdre. Nous verrons aux chapitres 3 et 4 que ces bornes supérieures (resp. inférieures) sont le pendant sémantique des opérateurs logiques additifs de disjonction (resp. conjonction.)

La notion de clôture est centrale dans notre travail. Nous verrons qu'elle caractérise le processus de complétion. Adaptée au cas des monoïdes ordonnés, on obtient la notion de prétopologie [Sam 93]. Nous montrons qu'il existe une plus grande prétopologie qui correspond à la manière « la plus fidèle[9] » de compléter un monoïde ordonné en quantale. Les quantales [Ros 90, Yet 90] correspondent à la sémantique algébrique de la logique intuitionniste linéaire et les algèbres de Heyting complètes correspondent à la sémantique algébrique de la logique intuitionniste. C'est pourquoi les outils de fabrication de quantales que nous introduisons seront si importants lorsque nous voudrons construire des contre-modèles de formules invalides dans ces logiques.

2.1 Ensembles, relations et notations

Nous noterons les ensembles munis de structures algébriques additionnelles (monoïdes, ordres, treillis...) par des lettres calligraphiques $\mathcal{X}, \mathcal{K}, \mathcal{M}$. Par contre, leurs sous-ensembles, ou parties, seront généralement notés par des lettres capitales X, Y, Z. Ces ensembles ne seront généralement pas munis d'une structure additionnelle. La relation d'inclusion ensembliste sera notée $\subseteq$ comme par exemple dans les relations suivantes : $X \subseteq \mathcal{X}$ et $X \subseteq Y$. Les éléments des ensembles ou des sous-ensembles seront notés avec des lettres minuscules x, y, z. La relation d'appartenance sera dénotée par $\in$ comme dans $x \in \mathcal{X}$ ou $x \in X$. L'ensemble des parties sera noté par $\mathbb{P}(\mathcal{X})$. Ainsi les deux relations $X \subseteq \mathcal{X}$ et $X \in \mathbb{P}(\mathcal{X})$ sont équivalentes. Les paires seront notées (x, y) pour deux éléments quelconques x et y. $\mathcal{X} \times \mathcal{Y}$ dénote l'ensemble des paires (x, y) d'éléments de $x \in \mathcal{X}$ et $y \in \mathcal{Y}$.

[9] Le sens du mot « fidèle » sera précisé plus loin dans ce chapitre.

Soit $\mathcal{X}$ et $\mathcal{Y}$ deux ensembles. On notera $f : \mathcal{X} \to \mathcal{Y}$ pour dire que f est une fonction de $\mathcal{X}$ dans $\mathcal{Y}$. L'image d'un élément $x \in \mathcal{X}$ par f sera notée $f(x)$. L'image directe d'une partie $X \subseteq \mathcal{X}$ sera notée $f(X)$ et vaut par définition $f(X) \triangleq \{f(x) \mid x \in X\}$. L'image réciproque $f^{-1}(Y)$ vaut quand à elle $f^{-1}(Y) \triangleq \{x \in \mathcal{X} \mid f(x) \in Y\}$. Nous écrirons $f : \mathcal{X} \rightarrowtail \mathcal{Y}$ si la fonction f est injective (ou injective à équivalence près, voir section 2.2.1.0), $f : \mathcal{X} \twoheadrightarrow \mathcal{Y}$ si elle est surjective et $f : \mathcal{X} \rightleftharpoons \mathcal{Y}$ si elle est bijective.

Soit $\mathcal{X}$ et $\mathcal{Y}$ deux ensembles. Une relation binaire $\succ$ entre $\mathcal{X}$ et $\mathcal{Y}$ est un sous-ensemble de $\mathcal{X} \times \mathcal{Y}$. On notera donc $\succ \subseteq \mathcal{X} \times \mathcal{Y}$. Si $x \in \mathcal{X}$ et $y \in \mathcal{Y}$ sont deux éléments, on dira que x est en relation avec y et on notera $x \succ y$ lorsque l'appartenance $(x, y) \in \succ$ sera vérifiée. Par exemple, les relations d'ordre définies en section 2.2.1 sont des exemples de relations binaires. Pour un élément $y \in \mathcal{X}$ et une partie $X \subseteq \mathcal{X}$ de $\mathcal{X}$, on notera $X \succ y$ si $\forall x \in X$, $x \succ y$. Dans le cas d'une relation $\leqslant$ d'ordre, $X \leqslant y$ veut dire que y est un majorant de X. De même, on notera $y \succ X$ si $\forall x \in X$, $y \succ x$ est vérifiée.

Enfin nous utiliserons souvent la notation abrégée $(\cdot)$ comme opérateur d'abstraction, quand il n'y a pas d'ambiguïté possible. Ainsi par exemple, la notation $(\cdot) + 1$ dénote la fonction d'incrémentation $x \mapsto x + 1$. La notation $(\cdot)^\star$ dénote la fonction $X \mapsto X^\star$ et la notation $(\cdot) \in \mathcal{X}$ dénote la relation d'appartenance à l'ensemble $\mathcal{X}$.

2.2 Complétion des ensembles ordonnés

Dans cette section, nous rappelons certaines définitions et quelques résultats de base sur les ordres et la façon usuelle de les étendre en treillis complets par ajout des bornes supérieures. La complétion se fait par clôture.

2.2.1 Ensembles ordonnés

Notion d'ordre

Définition 2.2.1 (Ensemble (pré)ordonné) *La paire $(\mathcal{P}, \leqslant)$ est un **ensemble préordonné** si $\mathcal{P}$ est un ensemble et $\leqslant \subseteq \mathcal{P} \times \mathcal{P}$ une relation binaire sur $\mathcal{P}$ à la fois **réflexive** et **transitive**. $(\mathcal{P}, \leqslant)$ est un **ensemble ordonné** si la relation $\leqslant$ est aussi **antisymétrique**.*

Par exemple l'ensemble des entiers naturels $(\mathbb{N}, \leqslant)$ est un ensemble ordonné. Un autre exemple fondamental est l'ensemble des parties, $(\mathbb{P}(X), \subseteq)$ ordonné par la relation d'inclusion entre parties. Les systèmes de transitions comme les réseaux de Petri introduisent aussi souvent une relation de transition $\rightsquigarrow$ qui est un préordre, voir section 4.6.1 chapitre 4. Par la suite, nous écrirons parfois seulement $\mathcal{P}$ pour parler de l'ensemble (pré)ordonné $(\mathcal{P}, \leqslant)$ dans la mesure où le (pré)ordre $\leqslant$ sera implicitement connu. D'autre part, nous utiliserons souvent le même symbole $\leqslant$ pour les (pré)ordres dans la mesure où ils sont le plus souvent distingués par le contexte.

À tout préordre $\leqslant$, on peut associer un **ordre strict** noté $<$, défini par $x < y \Leftrightarrow (x \leqslant y$ et $y \not\leqslant x)$. C'est une relation **irréflexive** $(x \not< x)$ et **transitive** incluse dans l'ordre $\leqslant$, i.e. $< \subseteq \leqslant$. Nous introduisons la notation $x \simeq y$ et nous dirons que x et y sont **équivalents** quand à la fois $x \leqslant y$ et $y \leqslant x$ sont vraies pour deux éléments x, y de $\mathcal{P}$. Nous avons donc la relation $x \leqslant y \Leftrightarrow (x < y$ ou $x \simeq y)$. Dans le cas d'un ordre, le fait que $\leqslant$ soit antisymétrique signifie que la relation $\simeq$ se confond avec l'égalité. La relation $\simeq$ est une relation d'équivalence et nous noterons $[x] \triangleq \{y \in \mathcal{P} \mid y \simeq x\}$ la **classe d'équivalence** de x.

Nous introduisons les notions de **borne supérieure** et de **borne inférieure**. Dans le contexte des ensembles préordonnés, i.e. quand la relation $\leqslant$ n'est pas nécessairement antisymétrique, une borne supérieure n'est pas unique. Elle est seulement unique à équivalence près.

On utilisera la notation $a \simeq \bigvee_i b_i$ qui signifiera que la condition $\forall x \in \mathcal{P}(a \leqslant x \Leftrightarrow \forall i,\, b_i \leqslant x)$ est vérifiée. Nous insistons sur le fait qu'il s'agit là d'une notation et qu'aucun élément $\bigvee_i b_i$ n'a été défini.[10] La borne inférieure $a \simeq \bigwedge_i b_i$ est définie par la condition $\forall x \in \mathcal{P}(x \leqslant a \Leftrightarrow \forall i,\, x \leqslant b_i)$. Le **plus grand élément**, s'il existe, correspond à la borne inférieure de la famille vide et sera noté $\top$. Le **plus petit élément** sera noté $\bot$. Le lecteur aura noté un léger abus de langage. En fait, il peut y avoir plusieurs plus grands éléments mais ils sont tous équivalents. Les éléments extrémaux sont uniques à équivalence près.

Plongements, équivalences

Un **morphisme** d'ensembles préordonnés est une fonction $\varphi : \mathcal{P} \to \mathcal{Q}$ croissante, c'est-à-dire qu'elle vérifie $\varphi(x) \leqslant \varphi(y)$ si $x \leqslant y$. Les morphismes préservent l'ordre mais pas nécessairement l'ordre strict. Ainsi, on peut très bien avoir $\varphi(x) = \varphi(y)$ et $x < y$. Par exemple, la fonction $x \mapsto \max(x, 2)$ de $\mathbb{N} \to \mathbb{N}$ bien que croissante, ne l'est pas strictement. Cette propriété n'est pas souhaitée lorsque l'on fait de l'interprétation des formules logiques, car on veut préserver les propriétés sémantiques, voir partie 4. Par conséquent, on utilise plutôt une autre notion plus forte :

Définition 2.2.2 (Plongement) *Soient $(\mathcal{P}, \leqslant)$ et $(\mathcal{Q}, \leqslant)$ deux ensembles préordonnés. Une fonction $i : \mathcal{P} \rightarrowtail \mathcal{Q}$ est un **plongement** de $\mathcal{P}$ dans $\mathcal{Q}$ si la condition $\forall x, y \in \mathcal{P},\ x \leqslant y \Leftrightarrow i(x) \leqslant i(y)$ est vérifiée.*

Un plongement est donc a fortiori un morphisme mais il est clair que le fonction $x \mapsto \max(x, 2)$ n'est pas un plongement. Dans le contexte des ensembles ordonnés, il s'agit en fait d'une fonction croissante injective. Une autre notion importante est celle d'équivalence d'ensembles ordonnés.

Définition 2.2.3 (Équivalence) *Un plongement $i : \mathcal{P} \rightleftharpoons \mathcal{Q}$ est une **équivalence** si i est surjective à équivalence près, i.e. la condition $\forall y \in \mathcal{Q},\ \exists x \in \mathcal{P},\ y \simeq i(x)$ est vérifiée.*

Dans le cas restreint des ordres, on parle plutôt d'**isomorphisme**. La notion d'équivalence de préordre est une « isomorphie à équivalence $\simeq$ près. » Le lecteur aura noté que l'on utilise le même terme d'équivalence pour une relation entre préordre et la relation $\simeq$ définie à partir de $\leqslant$. Ces deux notions ne pourront pas être confondues ici, car le contexte les distinguera toujours.[11]

Ordre quotient

Nous introduisons maintenant une construction élémentaire mais fondamentale qui fait le lien entre les deux notions d'ordre et de préordre et qui montre que l'on peut, dans de nombreux cas, substituer l'une à l'autre. Soit $(\mathcal{P}, \leqslant)$ un ensemble préordonné. On note de manière standard

$$\mathcal{P}/\!\simeq \triangleq \{[x] \mid x \in \mathcal{P}\}$$

l'ensemble des classes pour la relation d'équivalence $\simeq$. Cet ensemble se trouve muni d'un ordre défini par $[x] \leqslant [y] \Leftrightarrow x \leqslant y$.

[10] Cependant, s'il en existe une borne supérieure, alors toutes les autres lui sont équivalentes et de même, tout élément équivalent à une borne supérieure est une borne supérieure. Ceci est donc tout à fait cohérent avec la notation $\simeq$.

[11] Le lecteur averti pourra noter que la relation de plongement $\rightarrowtail$ est un préordre défini sur les ensembles ordonnés mais que la notion d'équivalence d'ensembles ordonnés $\rightleftharpoons$ n'est pas la symétrisée de ce préordre $\rightarrowtail$. Cette dernière est une notion plus faible, dans la catégorie des ordres [Mac 71] en tous cas.

Proposition 2.2.1 *Soit $(\mathcal{P}, \leqslant)$ un ensemble préordonné, alors $(\mathcal{P}/\simeq, \leqslant)$ est un ensemble ordonné et la fonction $i : \mathcal{P} \to \mathcal{P}/\simeq$ définie par $i(x) \triangleq [x]$ est une équivalence d'ensembles préordonnés.*

Démonstration. La relation $\leqslant$ qui est bien définie sur $\mathcal{P}/\simeq$ est réflexive, transitive. Elle hérite ces propriétés de la relation $\leqslant$ définie sur $\mathcal{P}$. Mais sur $\mathcal{P}/\simeq$, nous obtenons une relation antisymétrique car $([x] \leqslant [y]$ et $[y] \leqslant [x]) \Leftrightarrow (x \leqslant y$ et $y \leqslant x) \Leftrightarrow x \simeq y \Leftrightarrow [x] = [y]$. Par définition de $\leqslant$ sur $\mathcal{P}/\simeq$, i est un plongement et il est surjectif, toute classe étant de la forme $[x] = i(x)$. □

Il est important de garder cette construction en tête et de pouvoir passer de la notion d'ordre à celle de préordre au besoin. Dans le cadre de l'interprétation des formules logiques, les deux notions sont tout à fait interchangeables.

2.2.2 Treillis complets, clôtures

Dans un ensemble ordonné, l'existence des bornes supérieures et inférieures n'est pas garantie. Pour pouvoir interpréter les opérateurs logiques (additifs) de conjonction ($\wedge$ ou &) et de disjonction ($\vee$ ou $\oplus$), nous avons besoin de structures additionnelles, que nous introduisons maintenant.

Définition 2.2.4 (Treillis complet) *Un ensemble ordonné $(\mathcal{K}, \leqslant)$ est un **treillis complet** si pour toute famille $(x_i)_{i \in I}$ d'éléments de $\mathcal{K}$, la borne supérieure $\bigvee_{i \in I} x_i$ existe.*

L'exemple fondamental de treillis complet est l'ensemble des parties $(\mathbb{P}(X), \subseteq)$ d'un ensemble X, la borne supérieure étant l'union ensembliste $\bigcup_i X_i$ pour une famille de parties X_i. Un autre exemple est celui de l'ensemble des nombres réels complété $\{-\infty\} \cup \mathbb{R} \cup \{+\infty\}$. D'autres exemples vont être introduits par la suite à l'aide de la notion de clôture. L'existence de toutes les bornes supérieures assure l'existence de n'importe quelle borne inférieure par l'égalité

$$\bigwedge_{i \in I} x_i = \bigvee \{y \mid \forall i \in I \; y \leqslant x_i\}$$

Par symétrie, on a également l'identité $\bigvee_i x_i = \bigwedge \{y \mid \forall i \; y \leqslant x_i\}$ ce qui fait que l'on aurait très bien pu supposer l'existence des bornes inférieures dans la définition : les deux sont interchangeables.

Une méthode générique pour fabriquer des treillis complets consiste à utiliser un opérateur de clôture. Nous en ferons un usage important dans les parties de cette thèse qui se rapportent à la sémantique.

Définition 2.2.5 (Clôture) *Soit $\mathcal{X}$ un ensemble. On appelle **opérateur de clôture**, ou plus simplement **clôture**, toute fonction $(\cdot)^\star : \mathbb{P}(\mathcal{X}) \to \mathbb{P}(\mathcal{X})$ vérifiant la condition $X \subseteq Y^\star \Leftrightarrow X^\star \subseteq Y^\star$ pour toutes parties X, Y de $\mathcal{X}$. $X^\star$ est **la clôture** d'une partie X et une partie X est dite **$(\cdot)^\star$-close** si $X = X^\star$.*

La condition $X \subseteq Y^\star \Leftrightarrow X^\star \subseteq Y^\star$ peut être remplacée par les trois conditions $X \subseteq X^\star$, $X^{\star\star} \subseteq X^\star$ et $X \subseteq Y \Rightarrow X^\star \subseteq Y^\star$. De même une partie est $(\cdot)^\star$-close si et seulement si elle est de la forme $X^\star$. Le lecteur est invité à se rapporter à [Bir 67, Dav 90] pour tout complément sur la notion de clôture, qui est fondamentale. D'autre part, il est possible de construire des clôtures à partir de correspondances de Galois [Gal 92].[12]

[12] Cependant, nous n'aurons pas besoin de cet outil très général dans le cadre de cette thèse.

L'opérateur de clôture le plus simple que l'on puisse imaginer est la fonction identité. C'est d'ailleurs la plus petite au sens de l'inclusion point à point $\sqsubseteq$.[13] La plus grande clôture sur un ensemble $\mathcal{X}$ est la fonction constante $X \mapsto \mathcal{X}$. Si $\mathcal{X}$ est muni d'un préordre $\leqslant$, on peut définir deux opérateurs de clôture supplémentaires.

L'opérateur $\downarrow(\cdot)$ défini par

$$\downarrow X \triangleq \{y \in \mathcal{X} \mid \exists x \in X,\, y \leqslant x\} \tag{2.1}$$

est une clôture appelée **clôture inférieure** ou **section initiale** de X. L'autre opérateur de clôture très important sur les ensembles préordonnés est la **clôture de MacNeille** définie par

$$X^{\dagger} \triangleq \{x \in \mathcal{X} \mid \forall m \in \mathcal{X}, (X \leqslant m \Rightarrow x \leqslant m)\} \tag{2.2}$$

Dans le cas où $\mathcal{X}$ possède plus de deux éléments, le lecteur pourra noter que si l'ordre $\leqslant$ est **plat** — i.e. il s'identifie à l'égalité $(x \leqslant y \Leftrightarrow x = y,)$ — alors $\downarrow(\cdot)$ et $(\cdot)^{\dagger}$ sont respectivement la plus petite $(X \mapsto X)$ et la plus grande clôture $(X \mapsto \mathcal{X})$.

Quelques remarques de notations : l'ensemble $\mathcal{X}$ peut être vu comme la partie pleine de $\mathcal{X}$ et dans ce cas, $\mathcal{X}^{\star} = \mathcal{X}$. Dans la mesure où $\mathcal{X}^{\star}$ est connue et est égale à $\mathcal{X}$, nous préférons utiliser la notation $\mathcal{X}^{\star}$ pour autre chose. Nous définissons $\mathcal{X}^{\star}$ comme étant $\mathcal{X}^{\star} \triangleq \{X^{\star} \mid X \subseteq \mathcal{X}\}$ l'ensemble des parties closes de $\mathcal{X}$. Par abus de notation, nous écrirons $x^{\star}$ pour $\{x\}^{\star}$ pour un élément x de $\mathcal{X}$ lorsque cela ne prête pas à confusion. Le lecteur pourra remarquer que $\downarrow x = x^{\dagger}$, cette égalité n'étant valable que pour les singletons dans le cas général.

Proposition 2.2.2 *Si $(\cdot)^{\star}$ est une clôture sur l'ensemble $\mathcal{X}$ alors $(\mathcal{X}^{\star}, \subseteq)$ est un treillis complet.*

Démonstration. $(\mathcal{X}^{\star}, \subseteq)$ est tout d'abord un ensemble ordonné. Pour la structure de treillis complet, elle est obtenue par les identités suivantes :

$$\textstyle\bigwedge_i X_i = \bigcap_i X_i \qquad \text{et} \qquad \bigvee_i X_i = \left(\bigcup_i X_i\right)^{\star} \tag{2.3}$$

Pour prouver la première, il suffit de montrer que l'intersection d'une famille de parties $(\cdot)^{\star}$-closes est $(\cdot)^{\star}$-close, ce qui se fait par le calcul suivant : $(\bigcap_i X_i^{\star})^{\star} \subseteq \bigcap_i X_i^{\star} \Leftrightarrow \forall i\ (\bigcap_i X_i^{\star})^{\star} \subseteq X_i^{\star} \Leftrightarrow \forall i\ \bigcap_i X_i^{\star} \subseteq X_i^{\star}$ dernière condition qui est vérifiée. Pour la borne supérieure, nous obtenons le calcul : $\forall i\ X_i \subseteq Y^{\star} \Leftrightarrow \bigcup_i X_i \subseteq Y^{\star} \Leftrightarrow (\bigcup_i X_i)^{\star} \subseteq Y^{\star}$. □

La notion de clôture permet donc de fabriquer des treillis complets et nous verrons que nous pouvons fabriquer n'importe quel treillis de cette manière en section 2.2.3. Ce qui nous intéresse maintenant, c'est de pouvoir étendre un ensemble ordonné en treillis, c'est-à-dire, à partir d'un préordre $\mathcal{P}$, fabriquer des plongements $i : \mathcal{P} \rightarrowtail \mathcal{K}$ où $\mathcal{K}$ est un treillis complet.

2.2.3 Clôtures compatibles

Nous introduisons maintenant une adaptation de la notion de clôture à la structure d'ensembles ordonnés dans le but de fabriquer des treillis qui prolongent ces ensembles ordonnés.

[13] Si f et g sont deux fonctions $\mathcal{E} \to \mathcal{P}$ où $\mathcal{P}$ est un ensemble préordonné, on écrit $f \sqsubseteq g$ si et seulement si la condition $\forall x \in \mathcal{E},\ f(x) \leqslant g(x)$ est vérifiée.

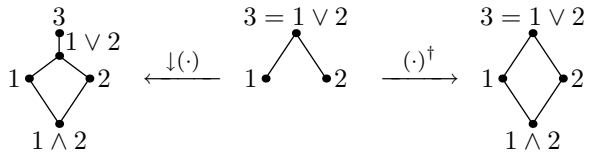


FIG. 2.1 – Deux exemples de complétion.

Compatibilité

Définition 2.2.6 (Clôture compatible) *Soit $(\mathcal{P}, \leqslant)$ un ensemble préordonné. Une clôture $(\cdot)^\star$ est dite* ***compatible*** *si pour tout $x \in \mathcal{P}$ on a $\downarrow x = x^\star$.*[14]

Bien sûr, la clôture $\downarrow(\cdot)$ est trivialement compatible. Si une clôture est compatible, alors $X^\star = \downarrow(X^\star)$ et donc $X^\star$ est une section initiale. La clôture de MacNeille $(\cdot)^\dagger$ est elle aussi compatible. Le résultat 2.2.6 précise la structure de l'ensemble des clôtures compatibles mais pour l'instant, intéressons-nous aux extensions de $\mathcal{P}$ par clôtures compatibles.

Proposition 2.2.3 *Soit $(\mathcal{P}, \leqslant)$ un ensemble préordonné et $(\cdot)^\star$ une clôture compatible alors $i : \mathcal{P} \rightarrowtail \mathcal{P}^\star$ définie par $i(x) \triangleq \downarrow x$ est un plongement. Ce plongement préserve les bornes inférieures et les plus grands éléments — s'ils existent dans $\mathcal{P}$.*

Démonstration. Pour x et y dans $\mathcal{P}$ on a $x \leqslant y \Leftrightarrow \downarrow x \subseteq \downarrow y$. Donc nous avons bien un plongement. Soit (x_k) une famille d'éléments de $\mathcal{P}$ et un élément a tel que $a \simeq \bigwedge_k x_k$ alors $\downarrow a = \bigcap_k \downarrow x_k$ d'où la préservation de la borne inférieure par les identités (2.3). □

Exemples de plongements

Prenons le cas d'un premier exemple simple. On considère l'ensemble ordonné des entiers naturels $(\mathbb{N}, \leqslant)$. Dans ce cas on obtient $\downarrow\mathbb{N} = \{[0, n[\mid 0 \leqslant n\} \cup \mathbb{N}$ et $\mathbb{N}^\dagger = \{[0, n[\mid 1 \leqslant n\} \cup \mathbb{N}$. On obtient un résultat différent : les deux treillis sont certes isomorphes mais le plongement n'est pas le même. Dans la complétion par $\downarrow(\cdot)$, le plus petit élément 0 n'est pas préservé puisqu'il vaut $\emptyset \neq \downarrow 0$ alors que dans le cas de $(\cdot)^\dagger$, il l'est et vaut $\{0\} = \downarrow 0$. Pour un contre-exemple en ce qui concerne la préservation de la borne supérieure, en figure 2.1, on peut voir que la borne supérieure $3 = 1 \vee 2$ n'est pas préservée dans le cas de $\downarrow(\cdot)$ alors qu'elle l'est dans le cas de $(\cdot)^\dagger$. En général, on a tout intérêt à utiliser la complétion de MacNeille $(\cdot)^\dagger$ si on veut préserver au maximum la structure initiale.

Préservation des bornes supérieures

Proposition 2.2.4 *Le plongement $i : \mathcal{P} \to \mathcal{P}^\dagger$ où $(\cdot)^\dagger$ est la clôture de MacNeille définie par l'identité (2.2) préserve les bornes supérieures et les plus petits éléments — s'ils existent dans $\mathcal{P}$.*

Démonstration. Pour une famille $(x_k)_{k \in \mathcal{K}}$ (éventuellement vide) et un élément a tel que $a \simeq \bigvee_k x_k$, on a $\downarrow a = \{x_k \mid k \in \mathcal{K}\}^\dagger = \left(\bigcup_k \downarrow x_k\right)^\dagger$. □

Ce dernier résultat possède un corollaire intéressant qui exprime le fait que tout treillis peut-être obtenu (à isomorphie près) par clôture (compatible d'un ensemble ordonné.)

[14] Attention, x est un élément de $\mathcal{P}$ et pas une partie de cet ensemble, sinon on aurait $\downarrow(\cdot) = (\cdot)^\star$ ce qui serait bien trop restrictif.

Corollaire 2.2.5 *Si $(\mathcal{K}, \leqslant)$ est un treillis complet alors la fonction $i : \mathcal{K} \to \mathcal{K}^\dagger$ définie par $i(x) \triangleq x^\dagger$ est un isomorphisme de treillis complets.*

Démonstration. i est un plongement qui préserve les bornes supérieures et les bornes inférieures. On a aussi $X^\dagger = \bigvee i(X) = i(\bigvee X)$. Donc i est surjectif. Ainsi, i est un isomorphisme d'ensembles ordonnés. Par conséquent, il préserve automatiquement la structure de treillis complet. □

Structure de l'ensemble des clôtures compatibles

Nous nous intéressons maintenant à la structure de l'ensemble des clôtures compatibles que nous noterons $\mathbb{C}(\mathcal{P})$, dont nous décrivons la structure d'ordre $\sqsubseteq$. Nous donnons également une caractérisation catégorique de cet ensemble ordonné à partir de la notion de $\vee$-complétion.

Proposition 2.2.6 *L'ensemble des clôtures compatibles définies sur $(\mathcal{P}, \leqslant)$ est noté $\mathbb{C}(\mathcal{P})$. Ordonné par l'inclusion point à point $\sqsubseteq$, il forme un treillis complet $(\mathbb{C}(\mathcal{P}), \sqsubseteq)$ dont le plus petit (resp. grand) élément est $\downarrow(\cdot)$ (resp. $(\cdot)^\dagger$).*

Démonstration. La structure de treillis est donnée par sa borne inférieure. Si $((\cdot)_k^\star)_{k\in K}$ est une famille (non vide) de clôtures compatibles alors on peut vérifier que $X \mapsto \bigcap_k (X)_k^\star$ est une clôture compatible. Pour le plus grand élément, vérifions tout d'abord que $(\cdot)^\dagger$ est compatible. Soit $y \leqslant x$ deux éléments de $\mathcal{P}$. Si on considère un élément m tel que $x \leqslant m$, alors on a aussi $y \leqslant m$. Ainsi $\downarrow x \subseteq x^\star$. Réciproquement, si $y \in x^\star$ alors en choisissant $m = x$, de $x \leqslant m$ on déduit $y \leqslant m$, i.e. $y \leqslant x$. Donc $x^\star \subseteq \downarrow x$. Vérifions maintenant que c'est le plus grand élément. Soit $(\cdot)^\star$ une clotûre compatible quelconque, X une partie quelconque de $\mathcal{P}$ et montrons $X^\star \subseteq X^\dagger$. Soit $x \in X^\star$. Si $X \leqslant m$ alors $X \subseteq \downarrow m = m^\star$ et donc $X^\star \subseteq m^\star = \downarrow m$ puis $X^\star \leqslant m$ et finalement $x \leqslant m$. □

Définition 2.2.7 (Complétion) *Soit $(\mathcal{P}, \leqslant)$ un ensemble préordonné. On appelle* ***$\vee$-complétion*** *de $\mathcal{P}$ un plongement $i : \mathcal{P} \rightarrowtail \mathcal{K}$ dans un treillis complet $(\mathcal{K}, \leqslant)$ qui soit $\vee$-dense, c'est-à-dire pour tout $k \in \mathcal{K}$ il existe une famille $(x_k)_{k\in I}$ de $\mathcal{P}$ telle que $k = \bigvee_k i(x_k)$.*

Par exemple, le plongement $i : \mathcal{P} \rightarrowtail \mathcal{P}^\star$ est $\vee$-dense car $X^\star = \left(\bigcup_{x\in X} \downarrow x\right)^\star = \bigvee_{x\in X} i(x)$ et par conséquent $\mathcal{P}^\star, \subseteq$ est une $\vee$-complétion de $(\mathcal{P}, \leqslant)$. En fait, il existe une correspondance entre la notion de clôture compatible et celle de $\vee$-complétion sous la forme d'une équivalence entre deux catégories. D'un coté nous avons le treillis des clôtures compatibles qui, en tant qu'ensemble ordonné, forme une catégorie, et de l'autre les $\vee$-complétions dont les morphismes $f : (i_1, \mathcal{K}_1) \to (i_2, \mathcal{K}_2)$ sont les fonctions $f : \mathcal{K}_1 \to \mathcal{K}_2$ qui préservent les bornes supérieures et commutent avec i_1 et i_2, c'est-à-dire $i_2 = f \circ i_1$.

Théorème 2.2.7 (Équivalence) *La catégorie des $\vee$-complétions de $\mathcal{P}$ est équivalente au treillis de clôtures compatibles sur $\mathcal{P}$.*

Cette caractérisation abstraite est certes très élégante et éclairante mais ne nous sera pas très utile par la suite. Aussi nous faisons le choix de citer le résultat, sans le démontrer, pour ne pas avoir à introduire un important vocabulaire catégorique. Le lecteur désireux d'approfondir ce point pourra se référer à la preuve du résultat dans [LW 00a].

2.3 Monoïdes ordonnés

2.3.1 Monoïdes

Définition 2.3.1 (Monoïde) *Un **monoïde (commutatif)** est une paire $(\mathcal{M}, \bullet)$ où $\mathcal{M}$ est un ensemble et $\bullet : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ une opération binaire sur $\mathcal{M}$, associative, commutative, et admettant un élément neutre noté $\mathtt{e}$. Un **morphisme** de monoïdes $f : \mathcal{M} \to \mathcal{Q}$ est une fonction vérifiant $f(\mathtt{e}_{\mathcal{M}}) = \mathtt{e}_{\mathcal{Q}}$ et $f(x) \bullet f(y) = f(x \bullet y)$ pour des éléments x et y quelconques de $\mathcal{M}$.*

Nous n'utiliserons que des monoïdes commutatifs, aussi nous omettrons le terme « commutatifs » par la suite, il sera sous-entendu sauf exception qui serait précisée. Les monoïdes sont des structures élémentaires très communes. Par exemple, l'ensemble des entiers naturels muni de l'addition $(\mathbb{N}, +)$ est un monoïde dont l'élément neutre est l'entier 0. Les groupes (commutatifs) sont a fortiori des monoïdes comme par exemple les groupes cycliques $\mathbb{Z}/p\mathbb{Z}$. Un exemple très important est celui des monoïdes libres.

Définition 2.3.2 (Multi-ensemble) *Un **multi-ensemble** (fini) est une fonction $M : \mathcal{X} \to \mathbb{N}^+$ où $\mathcal{X}$ est un ensemble fini, appelé **ensemble de définition** de M.*

Ici $\mathbb{N}^+$ désigne les entiers non nuls. Par exemple, on pourra noter $\{1, 1, 4\}$ le multi-ensemble décrit par la fonction $(1 \mapsto 2, 4 \mapsto 1)$ définie sur $\{1, 4\}$. L'ordre ne compte pas et on aurait pu écrire $\{1, 4, 1\}$. Par contre, le nombre d'occurences compte et $\{1, 4\} \neq \{1, 4, 1\}$. Le multi-ensemble défini sur l'ensemble vide est noté $\emptyset$. On peut faire la somme, notée $M + N$ de deux multi-ensembles. Par exemple $\{1, 4, 1\} + \{4, 1, 8\} = \{1, 1, 1, 4, 4, 8\}$. Si $\mathcal{K}$ est un ensemble, on appelle multi-ensemble sur $\mathcal{K}$ tout multi-ensemble défini sur une partie (finie) de $\mathcal{K}$.

Proposition 2.3.1 *L'ensemble des multi-ensembles sur $\mathcal{X}$, noté $\mathbb{M}_{\mathrm{f}}(\mathcal{X})$ muni de $+$ est un monoïde commutatif dont l'élément neutre est le multi-ensemble vide $\emptyset$. Il est isomorphe au monoïde libre, c'est-à-dire le monoïde commutatif engendré par $\mathcal{X}$.*

On considère un monoïde $(\mathcal{M}, \bullet)$ quelconque. L'opération monoïdale $\bullet$ peut s'étendre sur l'ensemble des parties de $\mathcal{M}$. Si X et Y sont deux parties de $\mathcal{M}$ alors on définit une opération binaire sur l'ensemble des parties $\mathbb{P}(\mathcal{M})$ notée $\cdot \bullet \cdot$ par

$$X \bullet Y \triangleq \{x \bullet y \mid x \in X \text{ et } y \in Y\} \tag{2.4}$$

Par abus de notation, pour m élément de $\mathcal{M}$, nous écrirons $x \bullet X$ au lieu de $\{x\} \bullet X$ s'il n'y a pas d'ambiguïté. Cette nouvelle opération sur $\mathbb{P}(\mathcal{M})$ est associative et admet un élément neutre $\{\mathtt{e}\}$ où $\mathtt{e}$ est l'élément neutre de $\mathcal{M}$. Par contre, $\emptyset$ est un élément absorbant.

Proposition 2.3.2 *L'ensemble des parties $(\mathbb{P}(\mathcal{M}), \cdot \bullet \cdot)$ du monoïde $\mathcal{M}$ est un monoïde dont l'élément neutre est $\{\mathtt{e}\}$.*

Ce dernier monoïde est bien connu en informatique puisqu'on l'utilise par exemple (dans le cas non-commutatif certes) pour les équations de point fixe sur des ensembles des mots, dans le but de définir des langages à partir de grammaires.[15] On se sert aussi de sa structure d'ordre (de treillis complet en fait) pour résoudre ces fameuses équations.

[15] Par exemple l'équation au point fixe $X = a.X \cup b$ a pour solution le langage $\{a^n b \mid n \in \mathbb{N}\}$.

2.3.2 Monoïdes préordonnés

Nous ajoutons maintenant une structure de préordre. Pour des besoins ultérieurs, nous affaiblissons la définition de monoïde à celle de **quasi-monoïde**. Que veut dire le terme quasi monoïde ? Il signifie simplement que la structure de monoïde est à équivalence $\simeq$ près, autrement dit les équations suivantes sont vérifiées :

Quasi unité	$a \bullet \mathsf{e} \simeq a$
Quasi associatif	$a \bullet (b \bullet c) \simeq (a \bullet b) \bullet c$
Quasi commutatif	$a \bullet b \simeq b \bullet a$

Cela peut sembler un peu lourd. De toutes façons, dans le cadre le plus couramment utilisé des ensembles ordonnés, l'équivalence $\simeq$ se confond avec l'égalité $=$ et on a alors un « vrai » monoïde. Dans ce cas, e est défini de manière unique par $\bullet$, et pas seulement à équivalence $\simeq$ près, et on ne le précise pas dans la structure $(\mathcal{M}, \bullet, \leqslant)$. La croissance de $\bullet$ veut dire que pour tous x, a, b éléments de $\mathcal{M}$ tels que $a \leqslant b$ on a $x \bullet a \leqslant x \bullet b$.

Définition 2.3.3 (Monoïde (pré)ordonné) *Un **monoïde (pré)ordonné** est un quadruplet $(\mathcal{M}, \bullet, \leqslant, \mathsf{e})$ tel que $(\mathcal{M}, \bullet, \mathsf{e})$ soit un quasi-monoïde, $(\mathcal{M}, \leqslant)$ soit un ensemble (pré)ordonné et que $\bullet$ soit croissante en ses deux arguments. Un **plongement** de monoïdes préordonnés $f : \mathcal{M} \to \mathcal{N}$ est une fonction qui est à la fois un plongement d'ensembles (pré)ordonnés et un morphisme de monoïdes (à équivalence près), i.e. $f(\mathsf{e}_\mathcal{M}) \simeq \mathsf{e}_\mathcal{Q}$ et $f(x \bullet y) \simeq f(x) \bullet f(y)$.*

Définition 2.3.4 (Équivalence) *Une **équivalence** $i : \mathcal{M} \rightleftharpoons \mathcal{N}$ est un plongement de monoïdes préordonnés qui est aussi une équivalence d'ensembles préordonnés.*

Proposition 2.3.3 *L'opération monoïdale $\bullet$ passe au quotient par l'équivalence $\simeq$ et le quotient $(\mathcal{M}/\simeq, \bullet, \leqslant)$ est un monoïde ordonné. De plus, la fonction $i : \mathcal{M} \rightleftharpoons \mathcal{M}/\simeq$ définie par $i(x) = [x]$ est une équivalence de monoïdes préordonnés.*

Démonstration. D'après la proposition 2.2.1, nous avons déjà une structure d'ensemble ordonné sur $\mathcal{M}/\simeq$ définie par $[x] \leqslant [y] \Leftrightarrow x \leqslant y$. Comme l'opération monoïdale $\bullet$ est croissante, $\simeq$ est une congruence pour cette opération — on dit aussi que $\bullet$ passe aux classes — et on peut définir $[x] \bullet [y] \triangleq [x \bullet y]$. Par quasi unité — $a \bullet \mathsf{e} \simeq a$, — $[\mathsf{e}]$ est un élément neutre. De plus, sur les classes, $\bullet$ devient associative et commutative. Toujours d'après 2.2.1, i est déjà une équivalence d'ensembles préordonnés. □

Le monoïde libre de multi-ensembles $\mathbb{M}_\mathrm{f}(\mathcal{X})$ muni de la relation d'inclusion[16] $\sqsubseteq$ entre multi-ensembles est un monoïde ordonné. L'ensemble des parties $(\mathbb{P}(\mathcal{M}), \cdot \bullet \cdot, \subseteq)$ d'un monoïde $\mathcal{M}$ muni de l'extension de $\cdot \bullet \cdot$ aux parties de $\mathcal{M}$ et ordonné par l'inclusion $\subseteq$ est aussi un monoïde ordonné.

$(\mathbb{N}, +, \leqslant)$ est un autre exemple de monoïde ordonné et on pourra noter qu'il est isomorphe à $(\mathbb{M}_\mathrm{f}(\{\blacklozenge\}), +, \sqsubseteq)$ où $\blacklozenge$ est un élément arbitraire. D'autres exemples seront présentés dans le chapitre 4. En section 4.6.3 plus particulièrement, nous verrons comment représenter n'importe quel monoïde préordonné (resp. fini) par un réseau de Petri (resp. fini.)

Proposition 2.3.4 *Si $(\mathcal{K}, \leqslant)$ est un treillis complet alors $(\mathcal{K}, \wedge, \leqslant)$ et $(\mathcal{K}, \vee, \leqslant)$ sont des monoïdes ordonnés.*

[16] $M \sqsubseteq N$ si et seulement si il y a moins d'occurences de chaque élément dans M que dans N.

Démonstration. L'opération de borne inférieure binaire $\vee$ est associative et commutative, son élément neutre est $\bot$. Elle est aussi croissante. En fait, on a seulement besoin de l'existence des bornes supérieures de familles finies sur l'ensemble ordonné $(\mathcal{K}, \leqslant)$ et de même dans le cas des bornes inférieures. □

Définition 2.3.5 (Résidu) *Soit $(\mathcal{M}, \bullet, \leqslant, \mathsf{e})$ un monoïde préordonné. Un **résidu** est un opérateur binaire $\multimap\!\!\bullet : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$ tel que pour tous x, a, b dans $\mathcal{M}$, on a $a \bullet x \leqslant b \Leftrightarrow x \leqslant a \multimap\!\!\bullet\, b$. Un monoïde préordonné sera dit **résidué** s'il possède au moins un résidu.*

En termes catégoriques, cela veut dire que $a \bullet (\cdot)$ et $a \multimap\!\!\bullet\, (\cdot)$ sont adjoints : ils forment une correspondance de Galois [Mac 71]. On peut avoir un résidu partiellement défini et on écrira par exemple $z \simeq a \multimap\!\!\bullet b$ si la condition $\forall x, x \leqslant z \Leftrightarrow a \bullet x \leqslant b$ est vérifiée. Comme les bornes supérieures et inférieures, le résidu est unique à équivalence près, ce qui justifie la notation précédente. Par exemple, sur le monoïde $(\mathbb{N}, +, \leqslant)$, la fonction $(a, b) \mapsto \max(b - a, 0)$ est un résidu.

Considérons le monoïde ordonné $(\mathbb{P}(\mathcal{M}), \bullet, \subseteq)$ des parties de $\mathcal{M}$ et le monoïde ordonné des multi-ensembles $(\mathbb{M}_{\mathrm{f}}(\mathcal{X}), +, \sqsubseteq)$:

Proposition 2.3.5 *L'opérateur défini par $X \multimap\!\!\bullet Y \triangleq \{z \mid z \bullet X \subseteq Y\}$ est un résidu sur le monoïde ordonné des parties $(\mathbb{P}(\mathcal{M}), \bullet, \subseteq)$.*

Démonstration. Pour X, A et B des parties quelconques de $\mathcal{M}$, on a

$$X \bullet A \subseteq B \Leftrightarrow \forall x \in X,\ x \bullet A \subseteq B \Leftrightarrow \forall x \in X,\ x \in A \multimap\!\!\bullet\, B \Leftrightarrow X \subseteq A \multimap\!\!\bullet\, B$$

□

Nous faisons remarquer que l'opérateur défini par $x \multimap\!\!\bullet\, y \triangleq \max(y - x, 0)$ n'est pas un résidu sur $(\mathbb{N}, +, \leqslant)$. En effet, comme on a $0 \leqslant 2 \multimap\!\!\bullet\, 1$, il s'en suivrait $2 + 0 \leqslant 1$ ce qui est faux. Par conséquent, le monoïde ordonné $(\mathbb{N}, +, \leqslant)$ et plus généralement les monoïdes ordonnés libres $\mathbb{M}_{\mathrm{f}}(\mathcal{X})$ **ne sont pas résidués**.

2.4 Espaces de phases, clôtures stables

Nous venons de présenter quelques exemples de monoïdes ordonnés. Tout comme nous avons montré comment il est possible de donner une structure de treillis complet à un ensemble quelconque avec le résultat 2.2.2, nous expliquons maintenant comment la notion de clôture stable permet d'ajouter une structure de treillis complet à un monoïde.

2.4.1 Définition

Définition 2.4.1 (Clôture stable/espace de phases) *Soit $(\mathcal{M}, \bullet)$ un monoïde. On appelle **clôture stable** toute clôture $(\cdot)^\star$ qui vérifie la propriété $X^\star \bullet Y^\star \subseteq (X \bullet Y)^\star$ pour toutes parties X et Y de $\mathcal{M}$. Le triplet $(\mathcal{M}, \bullet, (\cdot)^\star)$ est alors appelé **espace de phases**.*

La condition de stabilité $X^\star \bullet Y^\star \subseteq (X \bullet Y)^\star$ peut aussi s'écrire — i.e. est équivalente à — $X \bullet Y^\star \subseteq (X \bullet Y)^\star$ qui peut-être plus « facile » à prouver car on a $X^\star \bullet Y^\star \subseteq (X^\star \bullet Y)^\star \subseteq (X \bullet Y)^{\star\star} = (X \bullet Y)^\star$.

L'identité $X \mapsto X$ est la plus petite clôture stable et la fonction constante $X \mapsto \mathcal{M}$ la plus grande. Si $\leqslant$ est un ordre pour lequel $\bullet$ est croissante — autrement dit, $\mathcal{M}$ est un monoïde

ordonné — alors $\downarrow(\cdot)$ est aussi une clôture stable mais nous en reparlerons dans la prochaine section.

On considère un espace de phases $(\mathcal{M}, \bullet, (\cdot)^\star)$ fixé. Nous nous intéressons à la structure de l'ensemble des parties $(\cdot)^\star$-closes, c'est-à-dire de la forme $X = X^\star$. Nous définissons une opération binaire notée $(\cdot \bullet \cdot)^\star$ sur $\mathbb{P}(\mathcal{M})$ par

$$(\cdot \bullet \cdot)^\star : (X, Y) \mapsto (X \bullet Y)^\star$$

Proposition 2.4.1 *L'ensemble $\mathcal{M}^\star$ des parties $(\cdot)^\star$-stables de $\mathcal{M}$ muni de l'opération $(\cdot \bullet \cdot)^\star$ est un monoïde dont l'élément neutre est $\mathsf{e}^\star$.*

Démonstration. Montrons tout d'abord l'identité $(X^\star \bullet Y^\star)^\star = (X \bullet Y)^\star$. Par stabilité, il vient $X^\star \bullet Y^\star \subseteq (X \bullet Y)^\star$ et donc l'inclusion $\subseteq$ est acquise d'après la propriété caractéristique des clôtures — voir définition 2.2.5. L'inclusion réciproque est vraie pour n'importe quelle clôture, pas seulement les clôtures stables, parce que $Z \subseteq Z^\star$ et $\bullet$ est croissante sur $\mathbb{P}(\mathcal{M})$. Il est clair que $(\cdot \bullet \cdot)^\star$ peut se restreindre à $\mathcal{M}^\star$ car ses images $(X \bullet Y)^\star$ sont $(\cdot)^\star$-closes. Montrons l'associativité. $(X^\star \bullet (Y^\star \bullet Z^\star)^\star)^\star = (X^\star \bullet (Y \bullet Z)^\star)^\star = (X \bullet (Y \bullet Z))^\star = (X \bullet Y \bullet Z)^\star$. Par symétrie, on a aussi $((X^\star \bullet Y^\star)^\star \bullet Z^\star)^\star = (X \bullet Y \bullet Z)^\star$. D'où l'associativité. La commutativité est triviale. Enfin pour l'élément neutre, on rappelle que l'on note $\mathsf{e}^\star = \{\mathsf{e}\}^\star$. Alors il vient $(\mathsf{e}^\star \bullet X^\star)^\star = (\mathsf{e} \bullet X)^\star = X^\star$. □

D'autre part, $\mathcal{M}^\star$ hérite de $\mathbb{P}(\mathcal{M})$ une structure d'ensemble ordonné et possède même une structure de treillis d'après le résultat 2.2.2. On définit l'opération binaire notée $\multimap$ sur $\mathbb{P}(\mathcal{M})$ par

$$\multimap : (X, Y) \mapsto \{z \in \mathcal{M} \mid z \bullet X \subseteq Y\}$$

Il est facile de constater que $\multimap$ est croissante (ou co-variante) en son deuxième argument (Y dans la définition) et décroissante (ou contra-variante) en son premier (X dans la définition.) On pourra remarquer que $X \bullet (X \multimap Y) \subseteq Y$.

Proposition 2.4.2 *Pour toutes parties X et Y de $\mathcal{M}$, on a $X \multimap Y^\star = X^\star \multimap Y^\star$ et cette partie est $(\cdot)^\star$-close.*

Démonstration. Clairement on a déjà $X^\star \multimap Y^\star \subseteq X \multimap Y^\star$. Soit donc $z \in X \multimap Y^\star$. Alors on a $z \bullet X^\star \subseteq z^\star \bullet X^\star \subseteq (z \bullet X)^\star \subseteq Y^\star$ par stabilité de $(\cdot)^\star$. Donc on a $z \in X^\star \multimap Y^\star$. Montrons que $(X \multimap Y^\star)^\star \subseteq X \multimap Y^\star$. Soit $z \in (X \multimap Y^\star)^\star$. Ainsi il vient $z^\star \subseteq (X \multimap Y^\star)^\star$. On obtient le calcul $z \bullet X \subseteq z^\star \bullet X^\star \subseteq (X \multimap Y^\star)^\star \bullet X^\star \subseteq ((X \multimap Y^\star) \bullet X)^\star \subseteq (Y^\star)^\star = Y^\star$. Donc $z \in X \multimap Y^\star$. Donc $X \multimap Y^\star$ est $(\cdot)^\star$-close. □

Proposition 2.4.3 *$(\mathcal{M}^\star, (\cdot \bullet \cdot)^\star, \subseteq)$ est un monoïde ordonné et l'opération binaire $\multimap$ est un résidu sur $\mathcal{M}^\star$.*

Démonstration. Soient X, Y, Z des parties de $\mathcal{M}^\star$ telles que $X \subseteq Y$. Alors $Z \bullet X \subseteq Z \bullet Y$ et donc aussi $(Z \bullet X)^\star \subseteq (Z \bullet Y)^\star$. Nous avons bien un monoïde ordonné. Soient X, A, B trois parties $(\cdot)^\star$-closes de $\mathcal{M}$ alors $X \subseteq A \multimap B \Leftrightarrow X \bullet A \subseteq B \Leftrightarrow (X \bullet A)^\star \subseteq B$. □

2.4.2 Espaces quotients

On s'intéresse maintenant à la réduction des espaces de phases par passage au quotient. On se donne pour cela une projection (morphisme surjectif) notée $\lfloor\cdot\rfloor$ d'un monoïde $\mathcal{M}$ vers un monoïde $\mathcal{Q}$.

Définition 2.4.2 (Espace de phases quotient) *Soit $(\mathcal{M}, \bullet, (\cdot)^\star)$ un espace de phase. Le quadruplet $(\mathcal{Q}, \bullet, (\cdot)^\diamond, \lfloor\cdot\rfloor)$ est un* ***quotient*** *de $\mathcal{M}$ si $(\mathcal{Q}, \bullet, (\cdot)^\diamond)$ est un espace de phase et $\lfloor\cdot\rfloor : \mathcal{M} \twoheadrightarrow \mathcal{Q}$ est un morphisme de monoïdes surjectif — encore appelé* ***projection****. On suppose de plus que l'on a l'équivalence*

$$\lfloor x \rfloor \in \lfloor X \rfloor^\diamond \Leftrightarrow x \in X^\star \text{ pour tous } x \in \mathcal{M} \text{ et } X \subseteq \mathcal{M} \tag{2.5}$$

Nous rappelons que pour X une partie de $\mathcal{M}$, la notation $\lfloor X \rfloor$ est l'image extensionnelle de X par $\lfloor\cdot\rfloor$, c'est-à-dire $\lfloor X \rfloor \triangleq \{\lfloor x \rfloor \mid x \in X\}$. De même, on note l'image inverse pour Y partie de $\mathcal{Q}$, $\lfloor Y \rfloor^{-1} \triangleq \{x \in \mathcal{M} \mid \lfloor x \rfloor \in Y\}$. On pourra noter que pour $Y \subseteq \mathcal{Q}$ on a alors l'identité $Y^\diamond = \lfloor (\lfloor Y \rfloor^{-1})^\star \rfloor$. Ainsi $(\cdot)^\diamond$ est définie de manière unique par $\lfloor\cdot\rfloor$ mais cette dernière identité est plus faible que la condition (2.5). Nous rappelons les identités $\lfloor x \bullet y \rfloor = \lfloor x \rfloor \bullet \lfloor y \rfloor$ et $\lfloor \mathsf{e}_\mathcal{M} \rfloor = \mathsf{e}_\mathcal{Q}$ qui caractérisent un morphisme de monoïde. Enfin, tout élément y de $\mathcal{Q}$ est de la forme $\lfloor x \rfloor$ pour au moins un élément x de $\mathcal{M}$.

De l'identité $\lfloor X \rfloor^\diamond = \lfloor X^\star \rfloor$ pour toute partie X de $\mathcal{M}$, nous pouvons déduire que pour toute partie $(\cdot)^\star$-close X de $\mathcal{M}$, on a $\lfloor X^\star \rfloor = \lfloor X \rfloor = \lfloor X \rfloor^\diamond$. Ceci nous autorise à définir une fonction $\tau : \mathcal{M}^\star \twoheadrightarrow \mathcal{Q}^\diamond$ par $\tau(X) \triangleq \lfloor X \rfloor$ qui est bien sûr surjective. Nous nous intéressons maintenant aux propriétés de préservation de τ, dans la mesure où $\mathcal{M}^\star$ et $\mathcal{Q}^\diamond$ possèdent tous deux des structures de monoïdes résidués et des structures de treillis.

Lemme 2.4.4 *Soit X, Y, A_i des parties $(\cdot)^\star$-closes de $\mathcal{M}$ alors on a*

$$\begin{aligned}
\tau(\mathsf{e}^\star) &= \mathsf{e}^\diamond \\
\tau\big((X \bullet Y)^\star\big) &= \big(\tau(X) \bullet \tau(Y)\big)^\diamond \\
\tau(X \multimap Y) &= \tau(X) \multimap \tau(Y) \\
\tau(\textstyle\bigcap_i A_i) &= \textstyle\bigcap_i \tau(A_i) \\
\tau\big((\textstyle\bigcup_i A_i)^\star\big) &= \big(\textstyle\bigcup_i \tau(A_i)\big)^\diamond
\end{aligned}$$

Démonstration. Les deux premières identités sont triviales, la cinquième aussi. Pour la troisième et la quatrième, soit $x \in \mathcal{M}$. Nous obtenons les deux calculs suivants, présentés côte à côte :

$$\begin{array}{|ll|ll|}
 & \lfloor x \rfloor \in \tau(X) \multimap \tau(Y) & & \lfloor x \rfloor \in \tau(\bigcap_i A_i) \\
\Leftrightarrow & \lfloor x \rfloor \bullet \lfloor X \rfloor \subseteq \lfloor Y \rfloor & \Leftrightarrow & \lfloor x \rfloor \in \lfloor \bigcap_i A_i \rfloor^\diamond \\
\Leftrightarrow & \lfloor x \bullet X \rfloor \subseteq \lfloor Y \rfloor^\diamond & \Leftrightarrow & x \in (\bigcap_i A_i)^\star \\
\Leftrightarrow & x \bullet X \subseteq Y^\star & \Leftrightarrow & x \in \bigcap_i A_i \\
\Leftrightarrow & x \bullet X \subseteq Y & \Leftrightarrow & \forall i,\ x \in A_i \\
\Leftrightarrow & x \in (X \multimap Y)^\star & \Leftrightarrow & \forall i,\ x \in A_i^\star \\
\Leftrightarrow & \lfloor x \rfloor \in \lfloor X \multimap Y \rfloor^\diamond & \Leftrightarrow & \forall i,\ \lfloor x \rfloor \in \lfloor A_i \rfloor^\diamond \\
\Leftrightarrow & \lfloor x \rfloor \in \tau(X \multimap Y) & \Leftrightarrow & \lfloor x \rfloor \in \bigcap_i \tau(A_i)
\end{array}$$

□

Ainsi, à la fois la structure monoïdale résiduée et la structure de treillis complet sont préservées dans le quotient.

2.5 Quantales et prétopologies

Dans la section précédente 2.4, nous avons construit des monoïdes ordonnés résidués qui sont aussi des treillis complets, à partir d'un espace de phases. Ces structures sont appelées des quantales et leurs propriétés ont déjà été étudiées indépendamment des espaces de phases [Yet 90]. Nous nous intéressons surtout à leurs propriétés se rapportant à la sémantique de la logique intuitionniste linéaire et nous verrons que les quantales constituent la sémantique algébrique de cette logique, voir section 4.2.1 chapitre 4.

2.5.1 Définitions

Définition 2.5.1 (Quantale) *Une quantale est un triplet $(\mathcal{Q}, \bullet, \leqslant)$ où $(\mathcal{Q}, \bullet)$ est un monoïde commutatif et $(\mathcal{Q}, \leqslant)$ est un treillis complet et tel que pour tous éléments a, $(b_i)_i$ de $\mathcal{Q}$, on a $a \bullet \bigvee_i b_i = \bigvee_i (a \bullet b_i)$. On appelle cette relation* ***distributivité infinie****.*

La distributivité infinie s'instancie sur la famille vide par l'identité $x \bullet \bot = \bot$. Par conséquent, $\bot$ est un élément absorbant dans une quantale. Par exemple, $(\{\bot\} \cup \mathbb{N} \cup \{\infty\}, +, \leqslant)$ est une quantale, dans la mesure où on pose $\bot + \infty \triangleq \bot$. $([0, n], \max, \leqslant)$ est aussi une quantale.

Proposition 2.5.1 *Si $(\mathcal{K}, \leqslant)$ est un treillis complet alors $(\mathcal{K}, \vee, \leqslant)$ est une quantale. Si $\mathcal{K}$ est fini et distributif*[17] *alors $(\mathcal{K}, \wedge, \leqslant)$ est aussi une quantale.*

Ce dernier cas est intéressant puisque le treillis $(\mathcal{K}, \leqslant)$ est une **algèbre de Heyting** [Tro 88] — voir aussi section 3.2.1 chapitre 3 — si et seulement si $(\mathcal{K}, \wedge, \leqslant)$ est une quantale. D'autres exemples de quantales seront également présentés au chapitre 4.

Proposition 2.5.2 *Si $(\mathcal{M}, \bullet, \leqslant)$ est un monoïde ordonné et un treillis complet alors $\mathcal{M}$ est une quantale si et seulement si $\mathcal{M}$ est résidué.*

Démonstration. En supposant que $\mathcal{M}$ est une quantale alors il suffit de poser $a \multimap b \triangleq \bigvee\{x \mid x \bullet a \leqslant b\}$ et de vérifier que ceci définit bien un résidu. En effet, on a alors $a \bullet (a \multimap b) = \bigvee\{x \bullet a \mid x \bullet a \leqslant b\} \leqslant b$. Et par conséquent, $a \multimap b = \max\{x \mid x \bullet a \leqslant b\}$. D'où l'équivalence $x \bullet a \leqslant b \Leftrightarrow x \leqslant a \multimap b$. Réciproquement, si $\multimap$ est un résidu alors on obtient le calcul $x \bullet \bigvee b_i \leqslant m \Leftrightarrow \bigvee b_i \leqslant x \multimap m \Leftrightarrow \forall i,\ b_i \leqslant x \multimap m \Leftrightarrow \forall i,\ x \bullet b_i \leqslant m \Leftrightarrow \bigvee(x \bullet b_i) \leqslant m$. Par conséquent, on obtient la distributivité infinie. □

Proposition 2.5.3 *Si $(\mathcal{M}, \bullet, (\cdot)^\star)$ est un espace de phases alors $(\mathcal{M}^\star, (\cdot \bullet \cdot)^\star, \subseteq)$ est une quantale. L'élément neutre est $\mathsf{e}^\star$, le plus petit $\emptyset^\star$ et le plus grand $\mathcal{M}$. Pour le résidu, on a $X \multimap Y = \{z \in \mathcal{M} \mid z \bullet X \subseteq Y\}$ et pour les bornes sups. et infs, $\bigvee_i X_i = (\bigcup_i X_i)^\star$ et $\bigwedge_i X_i = \bigcap_i X_i$.*

Démonstration. Il suffit d'appliquer les résultats 2.4.3, 2.2.2 et 2.5.2. □

Nous savons donc fabriquer facilement des quantales. Nous nous intéressons maintenant à l'extension des monoïdes ordonnés en quantales, c'est-à-dire aux plongements $i : \mathcal{M} \rightarrowtail \mathcal{Q}$ d'un monoïde ordonné $\mathcal{M}$ dans une quantale $\mathcal{Q}$.

[17] Ici, on ne suppose que la distributivité finie des bornes supérieures sur les bornes inférieures.

2.5.2 Complétions, prétopologies

On se donne un monoïde préordonné $(\mathcal{M}, \bullet, \leqslant)$ fixé et on considère maintenant l'ensemble des clôtures qui sont à la fois compatibles avec $\leqslant$ et stables pour $\bullet$, ensemble que l'on note $\mathbb{C}(\mathcal{M})$.

Définition 2.5.2 (Prétopologie) *Soit $(\mathcal{M}, \bullet, \leqslant)$ un monoïde préordonné. On appelle **prétopologie** toute clôture $(\cdot)^\star$ sur $\mathcal{M}$ qui est à la fois compatible sur $(\mathcal{M}, \leqslant)$ et stable sur $(\mathcal{M}, \bullet)$. Nous rappelons les conditions :*

$$\begin{array}{ll} \textbf{\textit{Compatibilité}} & \forall x \in \mathcal{M},\ x^\star = {\downarrow}x \\ \textbf{\textit{Stabilité}} & \forall X, Y \subseteq \mathcal{M},\ X^\star \bullet Y^\star \subseteq (X \bullet Y)^\star \end{array}$$

Par exemple, la clôture inférieure $\downarrow(\cdot)$ est une prétopologie. Par contre, la clôture $(\cdot)^\dagger$ de MacNeille n'est pas, dans le cas général, une prétopologie. On peut en effet vérifier qu'elle n'est pas stable sur $([0,3], \max, \text{flat})$ où max est le maximum pour l'ordre naturel $0 < 1 < 2 < 3$ et flat est l'ordre plat sur cet ensemble.[18] La plus grande clôture compatible et stable est donnée par le résultat suivant :

Théorème 2.5.4 *La plus grande prétopologie, notée $(\cdot)^\circ$, est définie par l'équation suivante, pour $x \in \mathcal{M}$ et $X \subseteq \mathcal{M}$:*

$$x \in X^\circ \Leftrightarrow \forall a, m\ \ a \bullet X \leqslant m \Rightarrow a \bullet x \leqslant m \tag{2.6}$$

Démonstration. On voit tout d'abord que $a \bullet X \leqslant m \Rightarrow a \bullet X^\circ \leqslant m$ et par conséquent $X^{\circ\circ} \subseteq X^\circ$ et il est clair que $(\cdot)^\circ$ est une clôture. Il est aussi clair que cette clôture est compatible, i.e. ${\downarrow}x = x^\circ$. Montrons maintenant que $(\cdot)^\circ$ est stable. Soient X et Y deux parties de $\mathcal{M}$ et montrons que $X^\circ \bullet Y^\circ \subseteq (X \bullet Y)^\circ$. Soit $x \in X^\circ$ et $y \in Y^\circ$ et montrons $x \bullet y \in (X \bullet Y)^\circ$. Soient donc a et m fixés tels que $a \bullet (X \bullet Y) \leqslant m$. En fixant $x' \in X$, on a donc $a \bullet x' \bullet Y \leqslant m$ et donc, comme $y \in Y^\circ$, on a $a \bullet x' \bullet y \leqslant m$, ceci quel que soit $x' \in X$. Par conséquent, on obtient $a \bullet X \bullet y \leqslant m$ ou encore $a \bullet y \bullet X \leqslant m$. Comme $x \in X^\circ$, on obtient donc $a \bullet y \bullet x \leqslant m$ ou encore $a \bullet (x \bullet y) \leqslant m$. Donc $x \bullet y \in (X \bullet Y)^\circ$. $(\cdot)^\circ$ est une clôture compatible et stable et donc une prétopologie.

Soit maintenant une prétopologie quelconque $(\cdot)^\star$. Nous voulons montrer que $X^\star \subseteq X^\circ$ pour toute partie X de $\mathcal{M}$. Il suffit pour cela de montrer que pour deux éléments a et m quelconques de $\mathcal{M}$, on a $a \bullet X \leqslant m \Rightarrow a \bullet X^\star \leqslant m$. Or, en supposant $a \bullet X \leqslant m$ — i.e. $a \bullet X \subseteq {\downarrow}m$, — on obtient le calcul suivant : $a \bullet X^\star \subseteq a^\star \bullet X^\star \subseteq (a \bullet X)^\star \subseteq ({\downarrow}m)^\star = {\downarrow}m$. Donc pour toute prétopologie $(\cdot)^\star$, on a $(\cdot)^\star \sqsubseteq (\cdot)^\circ$. $(\cdot)^\circ$ est bien la plus grande prétopologie. □

Corollaire 2.5.5 *L'ensemble $\mathbb{C}(\mathcal{M})$ des prétopologies sur $\mathcal{M}$ muni de la relation d'ordre $\sqsubseteq$ forme un treillis complet dont le plus petit élément est $\downarrow(\cdot)$ et le plus grand $(\cdot)^\circ$.*

Démonstration. Nous avons déjà vu en proposition 2.2.6 que l'intersection (point à point) de clôtures compatibles est compatible. Il est aussi trivial de vérifier que l'intersection de clôtures stables est stable. Donc l'intersection point à point est l'opération de borne inférieure sur $\mathbb{C}(\mathcal{M})$, qui est par conséquent un treillis complet. Les éléments extrêmaux viennent d'être précisé dans ce qui précède. □

[18] Par exemple, si on prend $X \triangleq \{0,1\}$ et $Y \triangleq \{2\}$, on obtient $X^\dagger = \{0,1,2,3\}$ et $Y^\dagger = \{2\}$ d'où $X^\dagger \bullet Y^\dagger = \{2,3\}$. Par contre, $X \bullet Y = \{2\}$ et $(X \bullet Y)^\dagger = \{2\}$. Par conséquent, $X^\dagger \bullet Y^\dagger \not\subseteq (X \bullet Y)^\dagger$.

Proposition 2.5.6 *Soit $(\mathcal{M}, \bullet, \leqslant)$ un monoïde préordonné et $(\cdot)^\star$ une prétopologie sur $\mathcal{M}$. Alors $(\mathcal{M}^\star, (\cdot \bullet \cdot)^\star, \subseteq)$ est une quantale et $i : \mathcal{M} \rightarrowtail \mathcal{M}^\star$ définie par $i(x) \triangleq \downarrow x$ est un plongement de monoïdes préordonnés qui préservent les résidus (qui existent) et les bornes inférieures (qui existent.)*

Démonstration. D'après les résultats 2.2.2, 2.2.3, 2.4.3 et 2.5.2, la seule chose qui reste à prouver est que i préserve les résidus. Supposons donc que $z \simeq a \multimap b$, autrement dit, la condition suivante est vérifiée : $\forall x \;\; a \bullet x \leqslant b \Leftrightarrow x \leqslant z$. Alors il s'agit de montrer que $i(z) = i(a) \multimap i(b)$. Or $i(a) \multimap i(b) = \{a\} \multimap \downarrow b$ d'après la proposition 2.4.2. Un premier calcul donne : $z \leqslant z \Rightarrow z \bullet a \leqslant b \Rightarrow z \in \{a\} \multimap \downarrow b \Rightarrow z \in i(a) \multimap i(b) \Rightarrow i(z) \subseteq i(a) \multimap i(b)$ car $i(a) \multimap i(b)$ est $(\cdot)^\star$-close d'après 2.4.2. Un autre calcul pour x quelconque dans $\mathcal{M}$ donne : $x \in i(a) \multimap i(b) \Rightarrow x \in \{a\} \multimap \downarrow b \Rightarrow x \bullet a \leqslant b \Rightarrow x \leqslant z \Rightarrow x \in i(z)$. Donc $i(a) \multimap i(b) \subseteq i(z)$. □

On s'intéresse maintenant à la seule opération qui n'est pas préservée, c'est-à-dire la borne supérieure. Nous avions vu que pour les treillis, la clôture de MacNeille $(\cdot)^\dagger$ préserve les bornes supérieures, voir résultat 2.2.4. Dans le cas des quantales, $(\cdot)^\dagger$ n'est pas en général une prétopologie et d'autre part, il n'est pas forcément possible de préserver toutes les bornes supérieures. En effet, la condition de distributivité (infinie) n'est pas nécessairement vérifiée dans un monoïde ordonné complet quelconque, comme par exemple dans $(\{\emptyset, \{0\}, \{1\}, \{2\}, \{0,1,2\}\}, \cap, \subseteq)$. Les bornes supérieures qui sont préservées doivent au moins vérifier la distributivité infinie, et cette condition est suffisante. En particulier, si le monoïde ordonné est résidué alors les bornes supérieures sont préservées.

Proposition 2.5.7 *Le plongement $i : \mathcal{M} \rightarrowtail \mathcal{M}^\circ$ défini par $i(x) \triangleq \downarrow x$ — où $(\cdot)^\circ$ est la plus grande prétopologie, — préserve aussi les bornes supérieures $z \simeq \bigvee_k b_k$ qui existent, et qui vérifient la propriété de distributivité infinie, i.e. $\forall a,\; a \bullet z \simeq \bigvee_k (a \bullet b_k)$.*

Démonstration. On suppose $z \simeq \bigvee_k b_k$, i.e. $\forall x,\; (z \leqslant x \Leftrightarrow \forall k \; b_k \leqslant x)$. Pour toute partie X de $\mathcal{M}$, on a

$$\begin{aligned} z \in X^\circ &\Leftrightarrow \forall a, m,\; (a \bullet X \leqslant m \Rightarrow a \bullet z \leqslant m) \\ &\Leftrightarrow \forall a, m,\; \big(a \bullet X \leqslant m \Rightarrow (\forall k \; a \bullet b_k \leqslant m)\big) \\ &\Leftrightarrow \forall k, a, m,\; (a \bullet X \leqslant m \Rightarrow a \bullet b_k \leqslant m) \\ &\Leftrightarrow \forall k,\; b_k \in X^\circ \end{aligned}$$

Ainsi $i(z) \subseteq X^\circ \Leftrightarrow \forall k,\; i(b_k) \subseteq X^\circ$ est vérifiée et par conséquent, $i(z) = \bigvee_k i(b_k)$. □

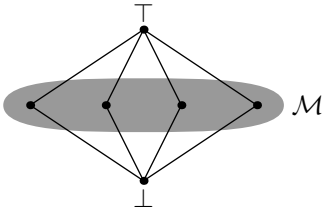
Corollaire 2.5.8 *Si $(\mathcal{M}, \bullet, \leqslant)$ possède déjà une structure de quantale alors $i : \mathcal{M} \rightleftharpoons \mathcal{M}^\circ$ défini par $i(x) \triangleq \downarrow x$ est un isomorphisme de quantales.*

Démonstration. Toutes les opérations algébriques sont préservées d'après les deux résultats qui précèdent, et d'autre part, i est surjective car on a $X^\circ = i(\bigvee X)$ pour toute partie X de $\mathcal{M}$. □

2.5.3 Exemples de complétions

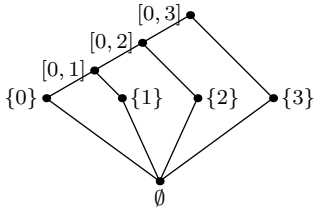
Nous présentons maintenant quelques exemples de complétions en utilisant la prétopologie $(\cdot)^\circ$. Tout d'abord deux exemples simples de monoïdes ordonnés fondamentaux :

$$\begin{aligned} (\mathbb{N}, +, \leqslant) &\xrightarrow{(\cdot)^\circ} \big(\{[0,n] \mid n \in \mathbb{N}\} \cup \{\emptyset, \mathbb{N}\}, +, \subseteq\big) \\ (\mathbb{Q}, +, \leqslant) &\xrightarrow{(\cdot)^\circ} \big(\mathbb{R} \cup \{-\infty, +\infty\}, +, \leqslant\big) \end{aligned}$$



$$\begin{array}{lcl} \bot \bullet \top & = & \bot \\ m \bullet \top & = & \top \\ m \bullet \bot & = & \bot \end{array}$$

FIG. 2.2 – Complétion d'un monoïde plat régulier.



$([0,3], \max, \text{flat})^\circ$

FIG. 2.3 – Complétion d'un monoïde plat non régulier.

Dans le cas de $\mathbb{Q}$, on remarque que la $(\cdot)^\circ$-complétion s'identifie à celle de Dedekind-MacNeille. On pourra noter que $(x, y) \mapsto y - x$ est un résidu dans la mesure où $\mathbb{Q}$ est un groupe ordonné. Le cas de $\mathbb{N}$ est intéressant car on constate que si 0 est le plus petit élément dans $(\mathbb{N}, \leqslant)$, son image $0^\circ = \{0\}$ ne l'est pas dans $(\mathbb{N}^\circ, \subseteq)$ puisque le plus petit élément est $\emptyset$. Voici donc un exemple $\bot$ n'est pas conservé. Il ne peut donc pas être distributif, et en effet, on constate par exemple que dans $(\mathbb{N}, +, \leqslant)$ on a $1 + \bigvee \emptyset = 1 + 0 = 1 \neq 0 = \bigvee \emptyset$.

Nous considérons maintenant un autre type d'exemple : la complétion d'un monoïde $\mathcal{M}$ plat, c'est-à-dire que $<$ est la relation vide. Dans le cas de $\downarrow(\cdot)$, on obtient toujours $\mathbb{P}(\mathcal{M})$ car cette complétion ne tient compte que de l'ordre. Par contre, dans le cas de $(\cdot)^\circ$, la structure monoïdale peut avoir une influence :

Proposition 2.5.9 *Si le monoïde plat $\mathcal{M}$ est régulier et contient plus de deux éléments alors $M^\circ = \mathcal{M} \cup \{\bot, \top\}$ avec $\bot \bullet x = \bot$ et pour $x \neq \bot$, $\top \bullet x = \top$.*

Bien sûr, lorsqu'on écrit $M^\circ = \mathcal{M} \cup \{\bot, \top\}$, il s'agit d'une isomorphie. D'autre part, nous précisons le sens de régulier. Cela veut dire que l'opérateur $\bullet$ est simplifiable, i.e. $\forall a, b, x,\ a \bullet x = b \bullet x \Rightarrow a = b$. C'est par exemple le cas des monoïdes libres (multi-ensembles) tels que $(\mathbb{N}, +)$ ou des groupes comme $\mathbb{Z}/p\mathbb{Z}$. La structure obtenue est représentée en figure 2.2. On remarque aussi que l'on obtient un treillis non-distributif pourvu que $\mathcal{M}$ soit de taille supérieure à trois. C'est une façon d'obtenir un contre-exemple à la distributivité de $\oplus$ sur $\&$, voir section 4.2.2 chapitre 4.

Cependant tous les monoïdes ne sont évidement pas réguliers. Considérons par exemple le monoïde $([0, n], \max)$, pour $n \geqslant 1$. Il n'est pas régulier parce que $\max(0, n) = \max(1, n)$. Nous obtenons la structure de treillis décrite en figure 2.3 qui présente le cas $n = 3$. On peut montrer que $\emptyset^\circ = \emptyset$, $x^\circ = \{x\}$ et si X a plus de deux éléments alors $X^\circ = [0, \max X]$.

2.5.4 Correspondance catégorique

Définition 2.5.3 (Complétion) *Soit $(\mathcal{M}, \bullet, \leqslant)$ un monoïde préordonné. On appelle* ***$\vee$-complétion*** *de $\mathcal{M}$ un plongement de monoïde préordonné $i : \mathcal{M} \rightarrowtail \mathcal{Q}$ dans une quantale $(\mathcal{Q}, \bullet, \leqslant)$ qui est $\vee$-dense. Les morphismes $f : (i_1, \mathcal{Q}_1) \to (i_2, \mathcal{Q}_2)$ sont les morphismes de quasi-monoïdes $f : \mathcal{Q}_1 \to \mathcal{Q}_2$ qui préservent les bornes supérieures et commutent avec i_1 et i_2, i.e. $i_2 = f \circ i_1$.*

Théorème 2.5.10 (Équivalence) *La catégorie des $\vee$-complétions du monoïde préordonné $\mathcal{M}$ est équivalente au treillis des prétopologies sur $\mathcal{M}$.*

Nous rappelons au lecteur qu'il pourra trouver une démonstration de ce résultat[19] dans l'article [LW 00a]. Cette démonstration fait uniquement appel aux notions de bases de la théorie des catégories et est donc relativement accessible au néophyte.

2.6 Interprétation des séquents intuitionnistes

Dans cette partie, on suppose donné un ensemble Form de formules logiques, qui dépend bien sûr de la logique utilisée. Par exemple, en figure 1.1, nous avons défini les formules de la logique intuitionniste. Le lecteur trouvera les formules de la logique intuitionniste linéaire en section 4.1 chapitre 4. Nous ne nous intéressons pas ici à la structure des formules mais seulement à leur combinaison dans ce que l'on appelle un séquent, et à l'interprétation sémantique de ces séquents, à partir d'une interprétation des formules de Form.

Définition 2.6.1 (Séquent intuitionniste) *Soit Form un ensemble de formules. On appelle **séquent intuitionniste** une paire (Γ, A) notée $\Gamma \vdash A$ où Γ est un multi-ensemble de formules et A est une formule. Γ est appelé **contexte** ou (multi-)ensemble d'**hypothèses** et A est appelé **conclusion**.*

Les séquents sont les briques de bases (éléments) des systèmes déductifs appelés **calculs des séquents** comme par exemple le système de règles décrit en figure 1.3 mais peuvent aussi servir dans certaines présentations de la déduction naturelle [Tro 96]. Nous informons le lecteur qu'il est d'usage d'écrire $A_1, A_2, \ldots, A_n$ au lieu de $\{A_1, A_2, \ldots, A_n\}$ pour un contexte. De la même manière, nous écrirons Γ, Δ au lieu de $\Gamma \bullet \Delta$ ou bien $\Gamma + \Delta$ pour la somme (ou concaténation ou superposition) des contextes Γ et Δ. Nous respecterons cet usage sauf quand un autre choix nous semblera plus expressif.

Si un système déductif a été défini pour les séquents, à partir de règles par exemples, nous écrirons $\Gamma \Vdash A$ au lieu de $\Vdash \Gamma \vdash A$ lorsque le séquent $\Gamma \vdash A$ est valide.

Définition 2.6.2 (Interprétation) *Soit Form un ensemble de formules et $(\mathcal{M}, \bullet, \leqslant)$ un monoïde ordonné. Une **interprétation** des formules de Form est une fonction $[\![\cdot]\!] : \mathsf{Form} \to \mathcal{M}$.*

En général, on interprète les formules à partir de leurs briques de bases, les formules atomiques et on a besoin d'une structure additionnelle pour interpréter les connecteurs logiques, comme par exemple une structure de treillis. Mais la structure de monoïde préordonné suffit pour interpréter les séquents intuitionnistes.

Étant donnée une interprétation $[\![\cdot]\!] : \mathsf{Form} \to \mathcal{M}$ des formules, on peut interpréter les contextes par

$$[\![\Gamma]\!] \triangleq [\![A_1]\!] \bullet \cdots \bullet [\![A_n]\!] \text{ pour } \Gamma \equiv A_1, \ldots, A_n$$

et de manière naturelle, $[\![\emptyset]\!] = \mathsf{e}$ pour le cas $n = 0$. Dans le cas de la logique intuitionniste, nous prendrons $\wedge$ comme opération monoïdale $\bullet$ et nous retrouvons $[\![A_1, \ldots, A_n]\!] = [\![A_1]\!] \wedge \cdots \wedge [\![A_n]\!]$.

Définition 2.6.3 (Interprétation fidèle) *Une interprétation $[\![\cdot]\!]$ est dite **fidèle (ou valide)** si pour tout séquent $\Gamma \vdash A$ tel que $\Gamma \Vdash A$, on a $[\![\Gamma]\!] \leqslant [\![A]\!]$. En particulier, si une formule A est valide alors $\mathsf{e} \leqslant [\![A]\!]$.*

[19] S'il éclaire les rapports entre clôtures et complétions, il ne sera en revanche pas utilisé par la suite.

Définition 2.6.4 (Contre-modèle) *On appelle* ***contre-modèle*** *d'une formule A une interprétation fidèle telle que* $\mathsf{e} \not\leqslant [\![A]\!]$.

Ainsi, si A admet un contre-modèle, A est invalide $\not\Vdash A$. Un contre-modèle est donc une « preuve » de l'invalidité de A. La construction de contre-modèle permet une description extensionnelle de la relation $\not\Vdash$, elle permet d'exhiber des exemples de formules invalides.

Définition 2.6.5 (Complétude sémantique) *Une classe de monoïdes préordonnés est dite* ***complète*** *pour une logique donnée si toute formule invalide admet un contre-modèle dans cette classe.*

Définition 2.6.6 (Propriété des modèles finis) *On dit qu'une classe de monoïdes préordonnés à la* ***propriété des modèles finis*** *si toute formule invalide admet un contre-modèle dans un monoïde préordonné fini de cette classe.*

La propriété des modèles finis est en fait une propriété de complétude par rapport à une classe de modèles qui sont tous finis.

Conclusion

Nous rappelons les bases de la complétion des ensembles ordonnés en treillis complets. Nous définissons la notion de clôture qui permet de fabriquer des treillis complets en considérant l'ensemble des parties closes. Grâce à une condition de compatibilité, la structure d'ordre initiale se plonge naturellement dans le treillis des parties closes. Nous caractérisons la clôture inférieure (resp. de MacNeille) comme la plus petite (resp. grande) clôture compatible.

Nous définissons ensuite les notions de monoïde ordonné et de résidu. Nous rappelons les concepts d'espace de phases et de clôture stable. La notion de quantale qui est la base de la sémantique algébrique de la logique intuitionniste linéaire est introduite comme la combinaison d'un treillis et d'un monoïde ordonné. Nous étudions les rapports entre quantales et espaces de phases. Ceci nous permet de fonder une théorie de la complétion des monoïdes ordonnés en quantales. Si on considère une clôture à la fois compatible et stable, autrement dit une prétopologie, alors l'ensemble des parties closes forme une quantale. Nous caractérisons les prétopologies extrémales ce qui nous donne une généralisation de la complétion de MacNeille au cas des monoïdes ordonnés.

Enfin, nous rappelons les bases de l'interprétation sémantique des séquents intuitionnistes dans une structure de monoïde ordonné ainsi que la signification des résultats de complétude (sémantique) et de propriété des modèles finis.

Dans le premier chapitre, la notion de réfutation est utilisée comme témoin de l'invalidité d'une proposition. C'est un critère syntaxique dans la mesure où les réfutations sont des objets syntaxiques. Les constructions sémantiques que nous venons de présenter permettent de définir des modèles. Ils peuvent servir de critères sémantiques à l'invalidité des propositions : si une proposition admet un contre-modèle, alors elle est invalide.

Dans les deux chapitres qui vont suivre, nous nous attacherons à montrer comment ces deux critères, réfutations d'une part et contre-modèles d'autre part, sont liés l'un à l'autre. Nous montrerons qu'une réfutation contient suffisamment d'information pour construire un contre-modèle et nous en déduirons des procédures pour construire automatiquement des contre-modèles finis.

3 | Logique intuitionniste

Introduction

La logique intuitionniste LI est entre autres la logique des types des langages fonctionnels [Gir 89]. Elle est au centre des relations entre preuves et programmes via l'isomorphisme de Curry-Howard [How 80]. Ainsi de nombreux travaux ont été consacrés à la recherche de preuves dans cette logique ou dans des méta-logiques — « logical frameworks » — fondés sur ses extensions. Dans ce contexte, la recherche de preuve [Dow 93] est utilisée comme base de la programmation fonctionnelle sûre [Nor 90, PM 93].

Comparé au cas de la logique classique propositionnelle, le problème de décision de la validité est plus complexe [Sta 79]. Un certain nombre de méthodes de preuve dans LI sont fondées sur la recherche de preuve en logique classique à partir de laquelle on étudie la prouvabilité intuitionniste [Rit 00, Bal 00, Wal 89]. D'autres mettent en œuvre des calculs des séquents [Vor 96, Dyc 92] ou techniques particulières (skolemization [Sha 92], résolution [Tam 96], BDD [GL 96] et preuves uniformes [Mil 91].)

La preuve par le calcul des séquents a fait de grand progrès lorsque Roy Dyckhoff a montré [Dyc 92] qu'il était possible de se passer de la règle de contraction. Avec LJT, il propose un système où il n'est pas nécessaire de détecter les boucles à l'intérieur de l'espace de recherche de preuves, car il n'y en a pas. Il se distingue en ce sens du calcul original LJ de Gentzen. D'autres travaux ont proposés des résultats similaires sous forme de calculs des séquents ou de méthodes de tableaux pour des logiques propositionnelles intermédiaires entre logique intuitionniste et logique classique [Mig 97, Ave 99, Dyc 99].

L'un des problèmes de la recherche de preuve dans LI est celui de la duplication de formules lors de l'application inverse des règles. Une gestion fine des formules vues comme des ressources rend nécessaire la recherche de solutions à ce problème de duplication. Le partage est une solution possible comme l'illustrent les travaux autour de structures telles que les BDDs [Gou 94].

Nos premiers travaux ont cherché à optimiser l'implantation de LJT par une gestion plus fine des ressources [Gal 98a]. D'autres travaux ont montré qu'il est possible de construire des contre-modèles à partir d'un échec de la recherche de preuve dans LJT [Pin 95, Wei 98]. Les travaux de Hudelmaier [Hud 93] ont montré comment gérer la duplication des sous-formules au niveau logique. Nous combinons ces différentes approches de construction de contre-modèles et de gestion des ressources, ici les formules logiques, pour obtenir un système appelé STRIP.

Dans un premier temps, nous rappelons la syntaxe de la logique intuitionniste propositionnelle définie par le biais de son calcul des séquents originel LJ. Puis nous rappelons la sémantique algébrique et introduisons la sémantique de Kripke comme cas particulier de la sémantique algébrique en nous servant des notions sémantique introduites au chapitre 2.

Dans un deuxième temps, nous mettons en œuvre les notions de réfutation et de co-validité introduites au chapitre 1 pour fournir une preuve de complétude du système LJT par rapport à la sémantique des arbres de Kripke finis ce qui prouve à la fois la complétude de ce système

par rapport à la logique intuitionniste et la propriété des modèles finis. D'autre part, la preuve contient un algorithme de construction de preuves ou de contre-modèles de Kripke.

Dans un troisième temps, nous proposons un nouveau système SLJ. C'est une étape intermédiaire qui nous permet de supprimer une forme de duplication par le partage d'une sous-formule entre une hypothèse et la conclusion. À la manière de [Wei 98], nous proposons un algorithme de construction de preuves ou de contre-modèles extrait de la démonstration du théorème de complétude de SLJ que nous donnons — voir aussi [Gal 99].

Enfin, nous présentons le système STRIP qui est une implantation du système SLJ dans laquelle il n'y a plus aucune duplication. L'algorithme de construction de preuves ou contre-modèles basé sur la recherche de preuve se traduit dans un programme. Nous discutons les conséquences de l'absence de duplication sur la gestion efficace des ressources lors de la recherche de preuve et des choix d'implantation au sein de STRIP.

3.1 Syntaxe

3.1.1 Les formules de la logique intuitionniste

L'ensemble Form des formules de la **logique intuitionniste propositionnelle**, notée LI, construites à partir d'un ensemble de variables Var est défini par induction :

$$\mathsf{Form}: \quad A, B ::= V \mid \mathsf{F} \mid \mathsf{T} \mid A \wedge B \mid A \vee B \mid A \to B \text{ pour } V \in \mathsf{Var}$$

Nous noterons usuellement les variables par les lettres V ou W. Les variables sont des formules **atomiques**, tout comme les symboles de constantes F (qui représente la valeur « faux ») et T (qui représente la valeur « vrai »).[20] Le symbole $\wedge$ représente la **conjonction** logique et le symbole $\vee$ la **disjonction** logique. Quand à $\to$, il représente l'**implication** logique. D'après la définition récursive, si A, B et C sont des formules alors $(A \vee B) \to (A \wedge C)$ est une formule. Par rapport à la logique classique, on constate que l'ensemble des formules est le même. Cependant la notion de validité est différente comme nous allons le voir par la suite. Pour une introduction plus détaillée à la logique intuitionniste, du point de vue formel et du point de vue philosophique, le lecteur est invité à consulter les deux ouvrages [Tro 88, Bee 84].

Un **contexte** est un multi-ensemble fini de formules, généralement dénoté par les lettres Γ ou Δ. Ainsi $A_1, \ldots, A_n$ dénote le multi-ensemble composé de n formules. L'ordre des formules n'a pas d'importance. Ainsi les contextes A, A, C et C, A, A sont identiques. On note par Γ, Δ la juxtaposition des contextes Γ et Δ et par Context l'ensemble des contextes.

3.1.2 Le calcul des séquents

Un **séquent intuitionniste** est une paire $(\Gamma, A) \in \mathsf{Context} \times \mathsf{Form}$ où Γ est un contexte et A une formule. Les formules de Γ sont appelées **hypothèses** et A est la **conclusion** de séquent. On écrira plutôt $\Gamma \vdash A$ pour dénoter un séquent. Si Γ est le contexte vide on écrira $\vdash A$ plutôt que $\emptyset \vdash A$. Le calcul des séquents pour LI est un système de règles décrit en figure 3.1. Ce calcul est appelé G2i est une version additive[21] du calcul des séquents de Gentzen. La règle de coupure [cut] permet l'utilisation du lemme intermédiaire A pour prouver B. Cette règle est particulière parce que l'on peut montrer qu'elle est redondante.

[20] Cependant, « vrai » et « faux » n'ont pas le même sens qu'en logique classique. En particulier, ils ne sont pas duaux l'un de l'autre.

[21] Pour une explication des notions d'additif et de multiplicatif, voir section 4.1.2.

Le système G2i

$$\frac{}{\Gamma, A \vdash A}\ [\mathrm{id}] \qquad \frac{\Gamma \vdash A \qquad \Gamma, A \vdash B}{\Gamma \vdash B}\ [\mathrm{cut}]$$

$$\frac{}{\Gamma, \mathsf{F} \vdash C}\ [\mathsf{F}_L] \qquad \frac{}{\Gamma \vdash \mathsf{T}}\ [\mathsf{T}_R]$$

$$\frac{\Gamma \vdash C}{\Gamma, \mathsf{T} \vdash C}\ [\mathsf{T}_L] \qquad \frac{\Gamma \vdash A}{\Gamma \vdash A \vee B}\ [\vee^1_R]$$

$$\frac{\Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma, A \vee B \vdash C}\ [\vee_L] \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B}\ [\vee^2_R]$$

$$\frac{\Gamma, A, B \vdash C}{\Gamma, A \wedge B \vdash C}\ [\wedge_L] \qquad \frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ [\wedge_R]$$

$$\frac{\Gamma, A \to B \vdash A \qquad \Gamma, B \vdash C}{\Gamma, A \to B \vdash C}\ [\to_L] \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}\ [\to_R]$$

FIG. 3.1 – Calcul des séquents G2i pour la logique intuitionniste propositionnelle, LI.

Théorème 3.1.1 (Élimination des coupures) *Tout séquent qui admet une preuve dans* G2i *admet aussi une preuve dans ce système privé de la règle de coupure* [*cut*].

D'autres règles redondantes importantes sont les règles structurelles de d'**affaiblissement** et de **contraction** :

$$\frac{\Gamma \vdash B}{\Gamma, A \vdash B}\ [\mathrm{weak}] \qquad \frac{\Gamma, A, A \vdash B}{\Gamma, A \vdash B}\ [\mathrm{contraction}]$$

Le choix de l'axiome $\Gamma, A \vdash A$ plutôt que $A \vdash A$ et d'une version additive des règles $[\vee_L]$, $[\wedge_R]$ et $[\to_L]$ permet de rendre ces deux règles redondantes. Pour des informations plus précises sur le système G2i, le lecteur est invité à consulter [Tro 96].

Nous définissons la notion de **validité intuitionniste** par le système G2i en posant $\Vdash \triangleq \vdash_{\mathsf{G2i}}$. Lorsqu'un séquent est valide — c'est-à-dire admet une preuve dans G2i — nous écrirons $\Gamma \Vdash A$ ou $\Gamma \vdash_{\mathsf{G2i}} A$ plutôt que $\Vdash \Gamma \vdash A$.[22] Nous faisons remarquer que nous serons amenés à utiliser plusieurs notions de validité différentes — en tous cas définies de manière différente, — en particulier lorsque nous introduisons une interprétation des séquents. C'est pourquoi nous préservons la notation $\vdash_{\mathsf{G2i}}$ définie par la prouvabilité dans G2i alors que la notation $\Vdash$ pourra recouvrir des sens différents suivant le contexte.

3.2 Sémantique

La logique intuitionniste peut être considérée comme une formalisation des mathématiques constructives de Brouwer ou Heyting [Bee 84]. Dans les mathématiques constructives, une proposition est considérée comme vraie si l'on peut en exhiber une preuve et comme fausse s'il est possible de montrer que l'assertion ne possède pas de preuve. La notion de vérité n'est donc pas

[22] Nous faisons simplement le choix de noter la relation binaire $\Vdash$ sur $\mathsf{Context} \times \mathsf{Form}$ de manière infixe plutôt que préfixe comme cela était fait auparavant dans la partie 1.

indépendante de l'étendue de notre connaissance. Ce qui n'est pas le cas des mathématiques classiques où l'on considère qu'une proposition A est soit vraie, soit fausse, même si l'on ne dispose ni d'une preuve de A, ni d'une preuve de $\neg A$: $A \vee \neg A$ est un axiome.

En logique intuitionniste, la vérité de $A \vee \neg A$ n'est acquise qu'à partir du moment où l'on dispose d'une preuve de cette disjonction, c'est-à-dire soit d'une preuve de A, soit d'une preuve de $\neg A$. Si A représente l'axiome du choix par exemple, alors la formule $A \vee \neg A$ ne possède pas de preuve.

Cette différence de point de vue se traduit au niveau de la sémantique de la manière suivante. En logique classique propositionnelle, l'interprétation usuelle se fait dans des algèbres de Boole, typiquement l'algèbre des fonctions $2^{\mathsf{Var}} \to 2$. On associe à chaque variable V sa projection canonique. Ce procédé s'étend aux formules (composées) par les opérations booléennes usuelles.

En logique intuitionniste propositionnelle, les modèles sont plus complexes. Par exemple, dans la sémantique de Kripke décrite en section 3.2.2, nous partons d'un univers de mondes dans lesquels chaque formule atomique possède une valeur de vérité. Cette valeur peut très bien différer d'un monde à l'autre. Cependant, cela doit se faire dans le respect de la condition de monotonie 3.2.8 en page 43.

3.2.1 Sémantique algébrique

Dans cette section, nous présentons les propriétés caractéristiques de l'algèbre des formules, considérées à équivalence logique près, aussi appelée algèbre de Lindenbaum [Ras 63].

Définition 3.2.1 (Algèbre de Heyting complète) *Une **algèbre de Heyting complète** est un treillis complet $(\mathcal{H}, \leqslant)$ tel que $(\mathcal{H}, \wedge, \leqslant)$ soit une quantale.*

On rappelle que l'on note par $\bigwedge$ (resp. $\wedge$) l'opération de borne inférieure infinie (resp. binaire) et par $\top = \bigwedge \emptyset$ le plus grand élément de $\mathcal{H}$. Pour les bornes supérieures, on utilise les notations $\bigvee$, $\vee$ et $\bot$.

Proposition 3.2.1 *Soit $(\mathcal{H}, \leqslant)$ un treillis complet, $\mathcal{H}$ est une quantale si et seulement si l'une des deux conditions suivantes est vérifiée :*

1. *le monoïde ordonné $(\mathcal{H}, \wedge, \leqslant)$ est résidué ;*
2. *la distributivité infinie $a \wedge \bigvee_i b_i = \bigvee_i (a \wedge b_i)$ est vérifiée.*

Ce résultat est une conséquence directe de la proposition 2.5.2. On notera $\twoheadrightarrow$ le résidu de $\wedge$ et on a par conséquent l'équivalence

$$a \wedge b \leqslant c \Leftrightarrow a \leqslant b \twoheadrightarrow c \tag{3.1}$$

pour tous éléments a, b et c de $\mathcal{H}$. Toujours d'après la proposition 2.5.2, on a l'égalité $a \twoheadrightarrow b = \bigvee \{x \mid x \wedge a \leqslant b\}$. Une autre propriété fondamentale qui est vérifiée dans les algèbres de Heyting est l'équation

$$a \wedge (a \twoheadrightarrow b) = a \wedge b \tag{3.2}$$

qui se prouve très simplement à l'aide de (3.1). Il est clair que $b \leqslant a \twoheadrightarrow b$ et donc $a \wedge b \leqslant a \wedge (a \twoheadrightarrow b)$. Réciproquement $a \wedge (a \twoheadrightarrow b) \leqslant b$, car $a \twoheadrightarrow b \leqslant a \twoheadrightarrow b$ et (3.1), et donc aussi $a \wedge (a \twoheadrightarrow b) \leqslant a \wedge b$.

Nous donnons les exemples fondamentaux d'algèbres de Heyting complètes. Une topologie en est un. Une topologie $(\tau, \subseteq)$ est un ensemble de parties stable par unions quelconques et par intersections finies. C'est donc un treillis complet. L'opération de borne supérieure quelconque étant

l'union ensembliste et l'opération de borne inférieure binaire étant l'intersection ensembliste. Comme l'union distribue sur l'intersection, c'est donc une algèbre de Heyting complète.

Un treillis distributif fini est aussi une algèbre de Heyting complète. La correspondance très importante entre les treillis distributifs finis (c'est-à-dire algèbres de Heyting finies) et les modèles de Kripke finis dont la présentation est à venir est présentée de manière très claire dans [Dav 90].

Dans le cadre de la sémantique algébrique, l'interprétation des formules logiques et des séquents consiste à associer un opérateur algébrique à chaque opérateur logique. À la conjonction logique $\wedge$ on associe la borne inférieure $\wedge$, à la disjonction $\vee$ on associe la borne supérieure $\vee$ et à l'implication logique $\rightarrow$ on associe le résidu $\twoheadrightarrow$. Soit $\sigma : \mathsf{Var} \rightarrow \mathcal{H}$ une interprétation des variables logiques. Alors on interprète les formules de la manière suivante :

Sémantique algébrique

$$\begin{array}{ll} [\![V]\!] \triangleq \sigma(V) & [\![A \rightarrow B]\!] \triangleq [\![A]\!] \twoheadrightarrow [\![B]\!] \\ [\![\mathsf{T}]\!] \triangleq \top & [\![A \wedge B]\!] \triangleq [\![A]\!] \wedge [\![B]\!] \\ [\![\mathsf{F}]\!] \triangleq \bot & [\![A \vee B]\!] \triangleq [\![A]\!] \vee [\![B]\!] \end{array}$$

Les séquents intuitionnistes sont interprétés dans le monoïde ordonné $(\mathcal{H}, \wedge, \leqslant)$ comme cela a été expliqué en section 2.6. Autrement dit, si $\Gamma \equiv A_1, \ldots, A_n$ est un multi-ensemble alors $[\![\Gamma]\!] \triangleq [\![A_1]\!] \wedge \cdots \wedge [\![A_n]\!]$.

Proposition 3.2.2 (Adéquation) *Soit $(\mathcal{H}, \leqslant)$ une algèbre de Heyting complète et $\sigma : \mathsf{Var} \rightarrow \mathcal{H}$ une interprétation des variables. Si $\Gamma \vdash A$ est prouvable dans le système* G2i*, autrement dit $\Gamma \vdash_{\mathsf{G2i}} A$ est vrai, alors $[\![\Gamma]\!] \leqslant [\![A]\!]$.*

Démonstration. La preuve de ce résultat se fait par application de la proposition 1.2.2. Soit $\succ$ la relation sur les séquents $\succ \subseteq \mathsf{Context} \times \mathsf{Form}$ définie par $\Gamma \succ A \Leftrightarrow [\![\Gamma]\!] \leqslant [\![A]\!]$. Il suffit de montrer que la relation $\succ$ est close par rapport aux règles du système G2i et on obtient alors $\vdash_{\mathsf{G2i}} \subseteq \succ$. Considérons par exemple la règle

$$\frac{\Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma, A \vee B \vdash C}\ [\vee_L]$$

et montrons que cette règle est valide par rapport à $\succ$. On suppose que $\Gamma, A \succ C$ et $\Gamma, B \succ C$ et on cherche à montrer que $\Gamma, A \vee B \succ C$. On a donc $[\![\Gamma]\!] \wedge [\![A]\!] \leqslant [\![C]\!]$ et $[\![\Gamma]\!] \wedge [\![B]\!] \leqslant [\![C]\!]$. Par conséquent, on obtient $[\![\Gamma]\!] \wedge [\![A \vee B]\!] = [\![\Gamma]\!] \wedge ([\![A]\!] \vee [\![B]\!]) = ([\![\Gamma]\!] \wedge [\![A]\!]) \vee ([\![\Gamma]\!] \wedge [\![B]\!]) \leqslant [\![C]\!] \vee [\![C]\!] = [\![C]\!]$ par distributivité. On obtient donc bien $\Gamma, A \vee B \succ C$.

Pour les autres règles, nous indiquons simplement les principaux arguments de la preuve sans entrer dans les détails :

- les cas des axiomes [id], $[\mathsf{T}_R]$, $[\mathsf{F}_L]$ sont triviaux ;
- pour la règle $[\mathsf{T}_L]$, il suffit de constater que $\top$ est l'élément neutre de $\wedge$ dans un treillis ;
- pour la règle de coupure [cut], il suffit de voir que si $[\![\Gamma]\!] \leqslant [\![A]\!]$ alors $[\![\Gamma]\!] = [\![\Gamma]\!] \wedge [\![A]\!]$;
- pour $[\vee^1_R]$ par exemple, on a $[\![A]\!] \leqslant [\![A]\!] \wedge [\![B]\!]$;
- pour $[\wedge_L]$, la preuve est également triviale ;
- pour $[\wedge_R]$, $[\![A]\!] \wedge [\![B]\!]$ est le plus grand des minorants de $[\![A]\!]$ et $[\![B]\!]$, donc plus grand que Γ ;
- pour $[\rightarrow_R]$, c'est l'équation (3.1) qui intervient ;
- enfin pour $[\rightarrow_L]$, on constate tout d'abord $[\![A]\!] \wedge ([\![A]\!] \twoheadrightarrow [\![B]\!]) = [\![A]\!] \wedge [\![B]\!]$ d'après l'équation (3.2). On obtient donc le calcul $[\![\Gamma]\!] \wedge [\![A \rightarrow B]\!] \leqslant [\![\Gamma]\!] \wedge [\![A]\!] \wedge ([\![A]\!] \rightarrow [\![B]\!]) \leqslant [\![\Gamma]\!] \wedge [\![A]\!] \wedge [\![B]\!] \leqslant [\![\Gamma]\!] \wedge [\![B]\!] \leqslant [\![C]\!]$.

□

On pourra aussi noter que les règles structurelles sont elles aussi valides. Par exemple pour la règle d'affaiblissement [weak], on obtient $[\![\Gamma, A]\!] = [\![\Gamma]\!] \wedge [\![A]\!] \leqslant [\![\Gamma]\!] \leqslant [\![B]\!]$.

Corollaire 3.2.3 *Soit $\mathcal{H}$ une algèbre de Heyting complète et σ une interprétation des variables. Si une formule A est une formule valide (dans* G2i*) alors $[\![A]\!] = \top$.*

Démonstration. Si A est valide alors par définition $\vdash_{\mathsf{G2i}} A$ est vrai et donc $\top = [\![\emptyset]\!] \leqslant [\![A]\!]$. □

Proposition 3.2.4 (Complétude et propriété de modèle finis) *Si le séquent $\Gamma \vdash A$ n'est pas prouvable dans* G2i *alors il existe une algèbre de Heyting complète finie $\mathcal{H}$ et une interprétation des variables telle que $\top \not\leqslant [\![A]\!]$.*

Ces propriétés seront formellement établies par la suite. Pour l'instant nous donnons seulement quelques indications pour des approches possibles. On peut utiliser la construction de Lindenbaum comme par exemple dans [Tro 88]. Cette technique est également présentée en partie 4 section 4.2.2 pour la logique intuitionniste linéaire. On peut aussi construire le contre-modèle de manière effective sous la forme d'un contre-modèle de Kripke fini, obtenu à partir d'une réfutation dans le système LJT, voir section 3.3.4.

3.2.2 Sémantique de Kripke

Une autre interprétation bien connue pour la logique intuitionniste est la sémantique de Kripke [Kri 65]. Nous montrons qu'elle peut être vue comme un cas particulier de la sémantique algébrique. En fait, il est possible d'associer une algèbre de Heyting complète à un modèle de Kripke d'une manière telle que les interprétations des formules soient identiques dans les deux modèles.

Définition 3.2.2 (Modèle de Kripke) *Un **modèle de Kripke** est un préordre $(\mathcal{K}, \leqslant)$ et une interprétation $\sigma : \mathsf{Var} \to \mathbb{P}(\mathcal{K})$ telle que pour toute variable V et $l \leqslant k$ dans $\mathcal{K}$ on ait $k \in \sigma(V) \Rightarrow l \in \sigma(V)$.*

Les éléments de $\mathcal{K}$ sont souvent appelés **mondes** et la relation $k \in \sigma(V)$ se lit « V est vraie dans le monde k. » La condition sur σ exprime simplement le fait que $\sigma(V)$ est une section initiale dans $(\mathcal{K}, \leqslant)$: $\downarrow\sigma(V) = \sigma(V)$. Elle se lit : « si V est vrai dans un monde, elle l'est aussi dans tous les mondes que le précèdent. » Du fait des résultats de la partie 2, $(\downarrow\mathcal{K}, \subseteq)$ forme alors un treillis complet. Mais nous obtenons mieux.

Proposition 3.2.5 *Si $(\mathcal{K}, \leqslant)$ est un préordre (resp. fini) alors $(\downarrow\mathcal{K}, \subseteq)$ est une algèbre de Heyting complète (resp. finie.) Les opérations algébriques sont données par :*

$$\bot = \emptyset \qquad \top = \mathcal{K} \qquad \bigvee_i A_i = \bigcup_i A_i \qquad \bigwedge_i A_i = \bigcap_i A_i$$

$$\boxed{A \twoheadrightarrow B = \{k \in \mathcal{K} \mid \forall l \leqslant k,\ l \in A \Rightarrow l \in B\}}$$

Démonstration. Nous connaissons déjà la structure de treillis complet de $\mathcal{K}$ par la proposition 2.2.2. Comme l'union quelconque de sections initiales est une section initiale, on a donc $\bigvee_i A_i = \downarrow\bigcup_i A_i = \bigcup_i A_i$. Il suffit de montrer que l'opérateur $\twoheadrightarrow$ défini par $A \twoheadrightarrow B \triangleq \{k \in \mathcal{K} \mid \forall l \leqslant k,\ l \in A \Rightarrow l \in B\}$ est un résidu de $\cap$. Tout d'abord, il est clair que $A \twoheadrightarrow B$ est bien une section initiale, c'est-à-dire $A \twoheadrightarrow B = \downarrow A \twoheadrightarrow B$. Il reste à montrer l'équivalence $X \cap A \subseteq B \Leftrightarrow X \subseteq A \twoheadrightarrow B$ pour toutes sections initiales X, A, B. Or $X \subseteq A \twoheadrightarrow B \Leftrightarrow \forall x \in X, \forall l \leqslant x,\ (l \in A \Rightarrow l \in B) \Leftrightarrow \forall x \in X,\ (x \in A \Rightarrow x \in B) \Leftrightarrow X \cap A \subseteq B$. □

L'interprétation σ dans un modèle de Kripke $(\mathcal{K}, \leqslant, \sigma)$ est tout simplement une valuation des variables dans l'algèbre de Heyting $\downarrow\mathcal{K}$ et donc un cas particulier de sémantique algébrique. Nous écrivons maintenant $\boldsymbol{k} \vDash \boldsymbol{A}$ pour $k \in [\![A]\!]$ et on dit $\boldsymbol{k}$ **force** $\boldsymbol{A}$. On peut alors réécrire la définition de la sémantique algébrique dans le cas particulier de l'algèbre de Heyting complète $(\downarrow\mathcal{K}, \leqslant)$:

Sémantique de Kripke

$k \vDash \mathsf{T}$	$k \vDash A \wedge B \Leftrightarrow k \vDash A$ et $k \vDash B$
$k \nvDash \mathsf{F}$	$k \vDash A \vee B \Leftrightarrow k \vDash A$ ou $k \vDash B$
$k \vDash V \Leftrightarrow k \in \sigma(V)$	$k \vDash A \to B \Leftrightarrow \forall l \leqslant k,\ l \vDash A \Rightarrow l \vDash B$

Proposition 3.2.6 (Adéquation) *Soit $(\mathcal{K}, \leqslant, \sigma)$ est un modèle de Kripke. Si A est une formule valide (dans* G2i*) alors pour tout monde $k \in \mathcal{K}$ on a $k \vDash A$.*

C'est une simple application du corollaire 3.2.3. Nous pouvons en déduire une définition de la notion de contre-modèle de Kripke.

Définition 3.2.3 (Contre-modèle de Kripke) *Soit A une formule intuitionniste. $(\mathcal{K}, \leqslant, \sigma)$ est un* ***contre-modèle*** *de A si il existe un monde $k \in \mathcal{K}$ tel que $k \nvDash A$.*

Proposition 3.2.7 *Si A admet un contre-modèle de Kripke alors A est invalide.*

C'est une application directe de l'adéquation. Un autre résultat fondamental dans le cadre de la sémantique de Kripke est la monotonie.

Proposition 3.2.8 (Monotonie) *Soit $(\mathcal{K}, \leqslant, \sigma)$ est un modèle de Kripke. Soit $l \leqslant k$ deux mondes de K. Si $k \vDash A$ alors $l \vDash A$.*

Il s'agit d'une simple application du fait que $[\![A]\!] = \{k \in \mathcal{K} \mid k \vDash A\}$ est une section initiale de $(\mathcal{K}, \leqslant)$. On voit donc que la sémantique de Kripke et les résultats associés sont des cas particuliers de la sémantique algébrique. Nous nous intéressons maintenant au cas particulier où la structure d'ordre de l'ensemble des mondes est celles d'un arbre fini.

3.2.3 Arbres de Kripke

Soit $\mathcal{T}$ un arbre fini dont les feuilles sont étiquetées par des parties de Var. On note $\mathcal{N}_\mathcal{T}$ l'ensemble des nœuds de l'arbre $\mathcal{T}$. Si n est un nœud de $\mathcal{T}$, on note $\mathcal{S}_n \subseteq \mathsf{Var}$ l'étiquette du nœud n. Ainsi $n \mapsto \mathcal{S}_n$ est une fonction $\mathcal{N}_\mathcal{T} \to \mathbb{P}(\mathsf{Var})$. On dénote par $\prec$ la relation de parenté. Ainsi on a $s \prec f$ lorsque le nœud s est le fils de f.

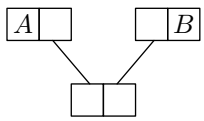


FIG. 3.2 – Un arbre de Kripke.

Par exemple, la formule $(A \to B) \vee (B \to A)$ est une formule valide en logique classique mais pas en logique intuitionniste. Un contre-modèle de Kripke pour cette formule est donné par la figure 3.2.[23] Il s'agit d'un arbre à trois nœuds dont la racine est étiquetée par l'ensemble vide $\emptyset$, la feuille de gauche par le singleton $\{A\}$ et la feuille de droite par le singleton $\{B\}$. On écrira $(\emptyset, \{(\{A\}, \emptyset), (\{B\}, \emptyset)\})$ pour une représentation linéaire de cet arbre.

[23] La justification du fait qu'il s'agit effectivement d'un contre-modèle est donnée plus loin.

Définition 3.2.4 (Arbre de Kripke) *Un **arbre de Kripke** est un arbre fini dont les feuilles sont étiquetées par des parties finies de l'ensemble* Var *des variables et tel que pour tous nœuds* $s \prec f$ *on a* $\mathcal{S}_f \subseteq \mathcal{S}_s$.

Soit $\mathcal{T}$ un arbre étiqueté par des parties finies de Var. Soit r la racine de cet arbre. On note $\mathcal{S}_\mathcal{T}$ l'ensemble $\mathcal{S}_r$, identifiant l'arbre à sa racine r. Si $\mathcal{T}_1, \ldots, \mathcal{T}_n$ sont les sous-arbres correspondant aux fils de la racine r, on écrira $\mathcal{T} \equiv (\mathcal{S}_\mathcal{T}, \{\mathcal{T}_1, \ldots, \mathcal{T}_n\})$. Si l'arbre $\mathcal{T}$ est juste une feuille, autrement dit r n'a aucun fils, alors on écrira $\mathcal{T} \equiv (\mathcal{S}_\mathcal{T}, \emptyset)$. $\{\mathcal{T}_1, \ldots, \mathcal{T}_n\}$ représente donc le multi-ensemble des sous-arbres directs de $\mathcal{T}$.

Proposition 3.2.9 *Soit* $\mathcal{T} \equiv (\mathcal{S}_\mathcal{T}, \{\mathcal{T}_1, \ldots, \mathcal{T}_n\})$. $\mathcal{T}$ *est un arbre de Kripke si et seulement si pour tout* i, $\mathcal{T}_i$ *est un arbre de Kripke et on a l'inclusion* $\mathcal{S}_\mathcal{T} \subseteq \mathcal{S}_{\mathcal{T}_i}$.

Cette caractérisation est une conséquence directe de la définition et nous donne une méthode de construction inductive des arbres de Kripke. Nous utiliserons cette méthode dans la construction des contre-modèles en section 3.3.4.

Nous montrons maintenant comment il est possible de voir l'ensemble des arbres de Kripke comme un modèle de LI. On note $\mathcal{K}$ l'ensemble des arbres de Kripke. Cet ensemble est muni de la relation de parenté $\prec \subseteq \mathcal{K} \times \mathcal{K}$. Ainsi $\mathcal{T}_i \prec \mathcal{T}$ si et seulement si $\mathcal{T}$ est de la forme $\mathcal{T} \equiv (\mathcal{S}_\mathcal{T}, \{\ldots, \mathcal{T}_i, \ldots\})$. Soit $\leqslant$ la clôture réflexive et transitive de la relation de parenté $\prec$, i.e. $\leqslant \triangleq \prec^\star$. Alors $\leqslant$ est une relation d'ordre entre les arbres de Kripke et $(\mathcal{K}, \leqslant)$ est un ensemble ordonné. On note $\downarrow\mathcal{T}$ la section initiale engendrée par un arbre $\mathcal{T}$ qui correspond à l'ensemble des sous-arbres (étiquetés) de $\mathcal{T}$. On peut définir une interprétation $\sigma : \mathsf{Var} \to \mathbb{P}(\mathcal{K})$ de la manière suivante : $\sigma(V) \triangleq \{\mathcal{T} \mid V \in \mathcal{S}_\mathcal{T}\}$. On note $\sigma_\mathcal{T}$ la restriction de σ à $\downarrow\mathcal{T}$, c'est-à-dire $\sigma_\mathcal{T}(V) = \{\mathcal{T}' \mid \mathcal{T}' \leqslant \mathcal{T}$ et $V \in \mathcal{S}_{\mathcal{T}'}\}$.

Proposition 3.2.10 *Pour tout arbre de Kripke* $\mathcal{T}$, *le triplet* $(\downarrow\mathcal{T}, \leqslant, \sigma_\mathcal{T})$ *est un modèle de Kripke.*

Démonstration. Soit $\mathcal{T}_i \prec \mathcal{T}$. Alors $\mathcal{S}_\mathcal{T} \subseteq \mathcal{S}_{\mathcal{T}_i}$ et par conséquent, pour toute variable V, on a $\mathcal{T} \in \sigma(V) \Rightarrow V \in \mathcal{S}_\mathcal{T} \Rightarrow V \in \mathcal{S}_{\mathcal{T}_i} \Rightarrow \mathcal{T}_i \in \sigma(V)$. Par transitivité, si on suppose $\mathcal{T} \prec^\star \mathcal{T}'$ alors $\mathcal{T}' \in \sigma(V) \Rightarrow \cdots \Rightarrow \mathcal{T} \in \sigma(V)$. □

Nous pouvons maintenant donner la sémantique des arbres de Kripke dans la mesure où $(\downarrow\mathcal{T}, \leqslant, \sigma_\mathcal{T})$ est un modèle de Kripke. Soit $\mathcal{T} \equiv (\mathcal{S}_\mathcal{T}, \{\mathcal{T}_1, \ldots, \mathcal{T}_n\})$. On obtient la définition :

Sémantique des arbres de Kripke

$\mathcal{T} \vDash \mathsf{T}$	$\mathcal{T} \vDash A \wedge B \Leftrightarrow \mathcal{T} \vDash A$ et $\mathcal{T} \vDash B$
$\mathcal{T} \nvDash \mathsf{F}$	$\mathcal{T} \vDash A \vee B \Leftrightarrow \mathcal{T} \vDash A$ ou $\mathcal{T} \vDash B$
$\mathcal{T} \vDash V \Leftrightarrow V \in \mathcal{S}_\mathcal{T}$	$\mathcal{T} \vDash A \to B \Leftrightarrow \mathcal{T} \vDash A \Rightarrow \mathcal{T} \vDash B$ et $\forall i$, $\mathcal{T}_i \vDash A \to B$

Démonstration. Mis à part la définition de $\mathcal{T} \vDash A \to B$ il ne s'agit que d'une réécriture de la sémantique de Kripke dans le cadre du modèle $(\downarrow\mathcal{T}, \leqslant, \sigma_\mathcal{T})$. Pour $\to$, nous aurions du traduire $\mathcal{T} \vDash A \to B \Leftrightarrow \forall \mathcal{T}' \leqslant \mathcal{T}$, $\mathcal{T}' \vDash A \Rightarrow \mathcal{T}' \vDash B$. Or si $\mathcal{T}' \leqslant \mathcal{T}$, on a soit $\mathcal{T}' = \mathcal{T}$, soit il existe i tel que $\mathcal{T}' \leqslant \mathcal{T}_i$ ce qui autorise la transformation.[24] □

[24] Une démonstration formelle consiste bien sûr à faire une preuve de l'équivalence par induction sur l'arbre $\mathcal{T}$.

Proposition 3.2.11 (Adéquation) *Si A est valide alors pour tout arbre de Kripke $\mathcal{T}$ on a $\mathcal{T} \vDash A$.*

Proposition 3.2.12 (Monotonie) *Soit $\mathcal{T} \equiv (\mathcal{S}_\mathcal{T}, \{\mathcal{T}_1, \ldots, \mathcal{T}_n\})$ un arbre de Kripke et A une formule. Alors si $\mathcal{T} \vDash A$ on a $\mathcal{T}_i \vDash A$ pour tout i.*

Démonstration. Dans le modèle de Kripke $(\downarrow\mathcal{T}, \leqslant, \sigma_\mathcal{T})$, il suffit de constater que $\mathcal{T}_i \leqslant \mathcal{T}$ et d'appliquer la propriété 3.2.8. □

Définition 3.2.5 (Modèle strict) *Soit $A_1, \ldots, A_n \vdash B$ un séquent et $\mathcal{T}$ un arbre de Kripke. On dit que $\mathcal{T}$ est un **modèle strict** de $A_1, \ldots, A_n \vdash B$ et on note $A_1, \ldots, A_n \Vdash_\mathcal{T} B$ si $(\mathcal{T} \vDash A_1, \ldots, \mathcal{T} \vDash A_n) \Rightarrow \mathcal{T} \vDash B$*

Proposition 3.2.13 *Si le séquent $A_1, \ldots, A_n \vdash B$ est prouvable dans* G2i *alors tout arbre de Kripke en est un modèle strict.*

Démonstration. En appliquant n fois la règle $[\rightarrow_R]$, on obtient une preuve du séquent $\vdash A_1 \rightarrow (A_2 \ldots \rightarrow (A_n \rightarrow B) \ldots)$ et donc d'après la proposition 3.2.11, on a $\mathcal{T} \vDash A_1 \rightarrow (A_2 \ldots \rightarrow (A_n \rightarrow B) \ldots)$. Si on suppose $\mathcal{T} \vDash A_1, \ldots, \mathcal{T} \vDash A_n$, alors on peut déduire successivement $\mathcal{T} \vDash A_2 \rightarrow (A_3 \ldots \rightarrow (A_n \rightarrow B) \ldots)$ puis $\mathcal{T} \vDash A_3 \rightarrow (A_4 \ldots \rightarrow (A_n \rightarrow B) \ldots)$ puis ... puis $\mathcal{T} \vDash A_n \rightarrow B$ puis $\mathcal{T} \vDash B$. □

Définition 3.2.6 (Contre-modèle strict) *Soit $A_1, \ldots, A_n \vdash B$ un séquent. Un **contre-modèle strict** de ce séquent est un arbre de Kripke $\mathcal{T}$ tel que $\mathcal{T} \vDash A_1, \ldots, \mathcal{T} \vDash A_n$ et $\mathcal{T} \nvDash B$.*

Le qualificatif strict n'a de sens que pour un séquent. En effet un contre-modèle de la formule $A_1 \rightarrow A_2 \rightarrow \cdot \rightarrow A_n \rightarrow B$ est un arbre qui contient un sous-arbre dans lequel toutes les A_i sont forcées alors que B ne l'est pas. Il contient donc un contre-modèle strict du séquent $A_1, \ldots, A_n \vdash B$ mais n'en est pas nécessairement un lui-même. Si A est une formule alors un contre-modèle strict $\mathcal{T}$ du séquent $\vdash A$ est simplement un contre-modèle de A, c'est-à-dire $\mathcal{T} \nvDash A$.

Proposition 3.2.14 *Si une formule admet un contre-modèle alors elle est invalide.*

Reprenons par exemple l'arbre de Kripke en figure 3.2 et montrons qu'il constitue un contre-modèle strict de la formule $(A \rightarrow B) \vee (B \rightarrow A)$. Les nœuds sont $\{0, 1, 2\}$ et on note $\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2$ les sous-arbres induits. On a donc $\mathcal{T}_0 \equiv (\emptyset, \{\mathcal{T}_1, \mathcal{T}_2\})$, $\mathcal{T}_1 \equiv (\{A\}, \emptyset)$ et $\mathcal{T}_2 \equiv (\{B\}, \emptyset)$. Il vient donc $\mathcal{T}_1 \nvDash B$ et $\mathcal{T}_1 \vDash A$. Pour la deuxième feuille, on a $\mathcal{T}_2 \nvDash A$ et $\mathcal{T}_2 \vDash B$. On en déduit donc $\mathcal{T}_1 \nvDash A \rightarrow B$ et $\mathcal{T}_2 \nvDash B \rightarrow A$.[25] D'après la définition on peut en déduire $\mathcal{T}_0 \nvDash A \rightarrow B$ et $\mathcal{T}_0 \nvDash B \rightarrow A$. Par conséquent $\mathcal{T}_0 \nvDash (A \rightarrow B) \vee (B \rightarrow A)$.

3.3 Le système LJT

Le système LJT ou G4ip introduit par Roy Dyckhoff [Dyc 92] et sa variante introduite par Jörg Hudelmair [Hud 93] est une variante de G2i qui possède la propriété d'analycité. Il se dérive donc de manière naturelle en un algorithme de recherche de preuves ou de réfutation, voir section 1.5 chapitre 1.

Le système G4ip

$$\frac{}{\Gamma, A \vdash A}\ [\mathrm{id}] \qquad \frac{}{\Gamma, \mathsf{F} \vdash G}\ [\mathsf{F}_L]$$

$$\frac{\Gamma \vdash G}{\Gamma, \mathsf{T} \vdash G}\ [\mathsf{T}_L] \qquad \frac{}{\Gamma \vdash \mathsf{T}}\ [\mathsf{T}_R]$$

$$\frac{\Gamma, A, B \vdash G}{\Gamma, A \wedge B \vdash G}\ [\wedge_L] \qquad \frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ [\wedge_R]$$

$$\frac{\Gamma, A \vdash G \qquad \Gamma, B \vdash G}{\Gamma, A \vee B \vdash G}\ [\vee_L] \qquad \frac{\Gamma \vdash A}{\Gamma \vdash A \vee B}\ [\vee^2_R]$$

$$\frac{\Gamma, X, C \vdash G}{\Gamma, X, X \to C \vdash G}\ [\to^1_L] \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B}\ [\vee^1_R]$$

$$\frac{\Gamma, A \to (B \to C) \vdash G}{\Gamma, (A \wedge B) \to C \vdash G}\ [\to^2_L] \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \to B}\ [\to_R]$$

$$\frac{\Gamma, A \to C, B \to C \vdash G}{\Gamma, (A \vee B) \to C \vdash G}\ [\to^3_L] \qquad \frac{\Gamma, A, B \to C \vdash B \qquad \Gamma, C \vdash G}{\Gamma, (A \to B) \to C \vdash G}\ [\to^4_L]$$

$$\frac{\Gamma \vdash G}{\Gamma, \mathsf{F} \to C \vdash G}\ [\to^5_L] \qquad \frac{\Gamma, C \vdash G}{\Gamma, \mathsf{T} \to C \vdash G}\ [\to^6_L]$$

FIG. 3.3 – Calcul des séquents G4ip pour la logique intuitionniste propositionnelle, LI.

3.3.1 Le système

Le calcul des séquents initialement nommé LJT mais que nous dénoterons G4ip par cohérence avec [Tro 96] est décrit en figure 3.3. Si on le compare à G2i, on constate la disparition de la règle de coupure [cut] — qui de toutes façons était redondante — mais aussi la disparition de la règle

$$\frac{\Gamma, A \to B \vdash A \qquad \Gamma, B \vdash G}{\Gamma, A \to B \vdash G}\ [\to_L]$$

qui est remplacée par six règles $[\to^\star_L]$ qui correspondent aux différents cas pour la formule A dans la règle ci-dessus du système G2i. On notera que dans le cas de la règle $[\to^1_L]$, X représente une variable propositionnelle.[26]

3.3.2 Validité intuitionniste

Nous montrons maintenant que les règles du systèmes G4ip sont valides par rapport à la notion de validité définie en section 3.1.2 à partir du système G2i. Nous donnons la preuve complète modulo le fait que la validité de la règle d'affaiblissement [weak] soit déjà démontrée.[27]

Proposition 3.3.1 *La validité intuitionniste* $\vdash_{\mathsf{G2i}}$ *est close pour les règles du système* G4ip.

Démonstration. Les règles communes aux systèmes G4ip et G2i sont évidemment valides. Il reste donc à montrer la validité des six règles $[\to^\star_L]$. Nous utilisons pour cela les règles de G2i, en particulier la règle de coupure, ainsi que la règle dérivée d'affaiblissement [weak]. Considérons par exemple la règle $[\to^1_L]$:

$$\dfrac{\dfrac{\dfrac{}{\Gamma, X, X \to B \vdash X}\,[\mathrm{id}] \quad \dfrac{}{\Gamma, X, B \vdash B}\,[\mathrm{id}]}{\Gamma, X, X \to B \vdash B}\,[\to_L] \quad \dfrac{\Gamma, X, B \vdash G}{\Gamma, X, X \to B, B \vdash G}\,[\mathrm{weak}]}{\Gamma, X, X \to B \vdash G}\,[\mathrm{cut}]$$

On constate donc que la validité de $\Gamma, X, B \vdash G$ implique celle que $\Gamma, X, X \to B \vdash G$, modulo le fait que la règle d'affaiblissement soit valide. On remarque également que le fait que X soit une variable n'intervient pas du tout ici, ce qui implique que la règle $[\to^1_L]$ reste valide même dans le cas où X n'est pas nécessairement une variable. Passons à la règle $[\to^2_L]$. On commence par obtenir une preuve du séquent $\Gamma, (A \wedge B) \to C \vdash A \to (B \to C)$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, A, B, (A \wedge B) \to C \vdash A}\,[\mathrm{id}] \quad \dfrac{}{\Gamma, A, B, (A \wedge B) \to C \vdash B}\,[\mathrm{id}]}{\Gamma, A, B, (A \wedge B) \to C \vdash A \wedge B}\,[\wedge_R] \quad \dfrac{}{\Gamma, A, B, C \vdash C}\,[\mathrm{id}]}{\Gamma, A, B, (A \wedge B) \to C \vdash C}\,[\to_L]}{\Gamma, A, (A \wedge B) \to C \vdash B \to C}\,[\to_R]}{\Gamma, (A \wedge B) \to C \vdash A \to (B \to C)}\,[\to_R]$$

[25] Par contre on a $\mathcal{T}_1 \vDash B \to A$ et $\mathcal{T}_2 \vDash A \to B$ mais cela n'a pas d'importance ici.

[26] Cependant, nous montrerons que l'extension de la règle $[\to^1_L]$ au cas où X représente n'importe quelle formule est aussi une règle valide.

[27] Nous n'avons pas donné la preuve de ce fait auparavant. Cependant la démonstration est triviale et se fait par induction sur les preuves mais voir aussi [Tro 96].

Il ne reste plus qu'à combiner cela par affaiblissement et coupure :

$$\dfrac{\dfrac{\vdots}{\Gamma, (A \wedge B) \to C \vdash A \to (B \to C)} \quad \dfrac{\Gamma, A \to (B \to C) \vdash G}{\Gamma, (A \wedge B) \to C, A \to (B \to C) \vdash G}\,[\text{weak}]}{\Gamma, (A \wedge B) \to C \vdash G}\,[\text{cut}]$$

Pour la règle $[\to^3_L]$, on commence par montrer que les séquents $\Gamma, (A \vee B) \to C \vdash B \to C$ et $\Gamma, (A \vee B) \to C, B \to C \vdash A \to C$ sont prouvables. Puis on combine de la façon suivante :

$$\dfrac{\dfrac{\vdots}{\Gamma, (A \vee B) \to C \vdash B \to C} \quad \dfrac{\dfrac{\vdots}{\Gamma, (A \vee B) \to C, B \to C \vdash A \to C} \quad \dfrac{\Gamma, A \to C, B \to C \vdash G}{\Gamma, (A \vee B) \to C, B \to C, A \to C \vdash G}\,[\text{weak}]}{\Gamma, (A \vee B) \to C, B \to C \vdash G}\,[\text{cut}]}{\Gamma, (A \vee B) \to C \vdash G}\,[\text{cut}]$$

Enfin pour la règle $[\to^4_L]$. Après avoir obtenu une preuve du séquent $\Gamma, (A \to B) \to C \vdash B \to C$ dans G2i, on combine ainsi :

$$\dfrac{\dfrac{\Gamma, (A \to B) \to C \vdash B \to C \quad \dfrac{\dfrac{\Gamma, A, B \to C \vdash B}{\Gamma, B \to C \vdash A \to B}\,[\to_R]}{\Gamma, (A \to B) \to C, B \to C \vdash A \to B}\,[\text{weak}]}{\Gamma, (A \to B) \to C \vdash A \to B}\,[\text{cut}] \quad \Gamma, C \vdash G}{\Gamma, (A \to B) \to C \vdash G}\,[\to_L]$$

La règle $[\to^5_L]$ apparaît comme un cas particulier de la règle [weak] d'affaiblissement. Enfin la règle $[\to^6_L]$ se traite de la manière suivante :

$$\dfrac{\dfrac{}{\Gamma, \mathsf{T} \to A \vdash \mathsf{T}}\,[\mathsf{T}_R] \quad \Gamma, A \vdash G}{\Gamma, \mathsf{T} \to A \vdash G}\,[\to_L]$$

□

Corollaire 3.3.2 *La prouvabilité dans* G4ip *est plus forte que dans* G2i*, autrement dit la relation* $\vdash_{\mathsf{G4ip}} \subseteq \vdash_{\mathsf{G2i}}$ *est vérifiée.*

3.3.3 Analycité

Dans cette section nous montrons que le système G4ip est analytique au sens précisé en section 1.5.3 chapitre 1 : nous rappelons qu'il s'agit de montrer que les réfutations ne peuvent être ni infiniment larges ni infiniment hautes.

Proposition 3.3.3 *Étant donné un séquent intuitionniste* $\Gamma \vdash G$*, il n'existe qu'un nombre fini d'instances de règles de* G4ip *ayant* $\Gamma \vdash G$ *comme conclusion.*

Démonstration. Pour une formule A donnée dans le multi-ensemble Γ, G, il existe au plus une instance de règle ayant A pour formule principale, d'où le résultat. □

Afin de prouver que les arbres de réfutation sont de hauteur finie, nous introduisons par contre des concepts plus complexes : on utilise une mesure de complexité dans l'ordinal ω^ω. On montre que l'application inverse d'une règle fait diminuer la complexité du séquent, autrement dit, la conclusion d'une règle est toujours plus complexe que ses hypothèses. Cette technique est exactement celle proposée par Roy Dyckhoff [Dyc 92].

On commence par mesurer les formules. Nous définissons :

$$\underline{A} \triangleq 1 \text{ si } A \in \mathsf{Var} \cup \{\mathsf{F}, \mathsf{T}\} \quad \underline{A \to B} \triangleq \underline{A} + \underline{B} + 1$$
$$\underline{A \wedge B} \triangleq \underline{A} + \underline{B} + 2 \qquad \underline{A \vee B} \triangleq \underline{A} + \underline{B} + 1$$

Puis nous définissons la complexité des multi-ensemble. Si $\Gamma \equiv A_1, \ldots, A_n$ est un multi-ensemble de formules, on donne la mesure[28] $\omega(\Gamma) \triangleq \Sigma_i \omega^{\underline{A_i}}$. Enfin, la mesure de complexité du séquent $\Gamma \vdash G$ est celle du multi-ensemble de ses formules Γ, G, c'est-à-dire $\omega(\Gamma \vdash G) \triangleq \omega(\Gamma) + \omega^{\underline{G}}$.

Lemme 3.3.4 *Soit Γ, A et $\Delta \equiv B_1, \ldots, B_k$ deux multi-ensembles de formules tels que pour tout i on ait $\underline{B_i} < \underline{A}$. Alors on a $\omega(\Gamma, B_1, \ldots, B_k) < \omega(\Gamma, A)$.*

Démonstration. Évidement, cette preuve fait appel à quelques notions de calcul ordinal. Le lecteur est invité à se référer à un cours de théorie des ensembles [Jec 97] pour plus d'informations sur les propriétés arithmétiques des ordinaux. Une preuve détaillée peut se trouver dans [Der 79]. Soit $m = \max\{\underline{B_1}, \ldots, \underline{B_k}\}$. On a $m < \underline{A}$ et donc $m + 1 \leqslant \underline{A}$. Alors $\Sigma_i \omega^{\underline{B_i}} \leqslant \Sigma_i \omega^m \leqslant \omega^m.k < \omega^m.\omega \leqslant \omega^{m+1} \leqslant \omega^{\underline{A}}$. D'où on en déduit $\omega(\Gamma) + \Sigma_i \omega^{\underline{B_i}} < \omega(\Gamma) + \omega^{\underline{A}}$. □

Ce lemme nous montre qu'avec notre mesure de complexité, il est possible de remplacer une formule par un nombre fini arbitraire de formules plus petites tout en diminuant strictement la complexité du multi-ensemble.

Proposition 3.3.5 *Pour toutes les instances de règles de* G4ip*, la conclusion est toujours de mesure $\omega(\cdot)$ strictement plus grande que les prémisses.*

Démonstration. L'application inverse d'une règle remplace la formule principale par un nombre fini (en fait $\leqslant 2$) de formules de mesure strictement plus petite. Voici un tableau des remplacements de la formule principale (Princ.) pour chaque règle (autre que les axiomes [id], $[\mathsf{F}_L]$ et $[\mathsf{T}_R]$) et chaque prémisse (P. 1 et P. 2) :

Règle	Princ.	P. 1	P. 2
$[\mathsf{T}]$	T		
$[\wedge_L]$	$A \wedge B$	A, B	
$[\wedge_R]$	$A \wedge B$	A	B
$[\vee_L]$	$A \vee B$	A	B
$[\vee_R^\star]$	$A \vee B$	A ou B	
$[\to_R]$	$A \to B$	A, B	

Règle	Princ.	P. 1	P. 2
$[\to_L^1]$	$X \to C$	C	
$[\to_L^2]$	$(A \wedge B) \to C$	$A \to (B \to C)$	
$[\to_L^3]$	$(A \vee B) \to C$	$A \to C, B \to C$	
$[\to_L^4]$	$(A \to B) \to C$	$A, B \to C, B$	C
$[\to_L^5]$	$\mathsf{F} \to C$		
$[\to_L^6]$	$\mathsf{T} \to C$	C	

et on pourra constater que le complexité diminue lorsque l'on passe de la colonne « Princ. » à la colonne « P. 1 » ou « P. 2. » Par exemple, $\underline{(A \wedge B) \to C} = \underline{A} + \underline{B} + \underline{C} + 3 > \underline{A} + \underline{B} + \underline{C} + 2 = \underline{A \to (B \to C)}$. □

Proposition 3.3.6 *Le système* G4ip *est analytique.*

[28] Les symboles + et Σ font ici référence à la somme naturelle sur les ordinaux.

Démonstration. On choisit l'ordre sur les séquents donné par la mesure de complexité $\omega(\cdot)$, autrement dit $(\Gamma \vdash G) < (\Gamma' \vdash G')$ si et seulement si $\omega(\Gamma \vdash G) < \omega(\Gamma' \vdash G')$. Nous obtenons un ordre strict bien fondé car ω^ω est un ordinal donc un ordre bien fondé. Il ne peut donc pas y avoir de branche infinie dans un arbre de réfutation. La proposition 3.3.3 complète la preuve. □

3.3.4 Complétude

Dans cette section, nous prouvons la complétude du système G4ip par rapport à la logique LI, qui a été définie par le système G2i. Pour cela, nous démontrons directement la complétude sémantique de G4ip par rapport à la classe des arbres de Kripke finis. Ce résultat combiné à l'équation (3.3) nous donne à la fois la complétude de G4ip par rapport à G2i et la propriété des modèles finis pour LI.

Soit $\mathcal{T}$ un arbre de Kripke. On peut définir une notion de validité dans cet arbre de Kripke par

$$A_1, \ldots, A_n \Vdash_{\mathcal{T}} B \quad \text{ssi} \quad (\mathcal{T} \vDash A_1, \ldots, \mathcal{T} \vDash A_n) \Rightarrow \mathcal{T} \vDash B$$

Autrement dit, le séquent $A, \ldots, A_n \vdash B$ est $\Vdash_{\mathcal{T}}$-valide si $\mathcal{T}$ est un modèle strict de ce séquent. D'après les propositions 3.3.2 et 3.2.13, nous avons les inégalités suivantes

$$\vdash_{\mathsf{G4ip}} \subseteq \vdash_{\mathsf{G2ip}} \subseteq \Vdash_{\mathcal{T}} \tag{3.3}$$

On pose $\Vdash \triangleq \bigcap_{\mathcal{T} \in \mathcal{K}} \Vdash_{\mathcal{T}}$. Pour prouver $\vdash_{\mathsf{G4ip}} = \vdash_{\mathsf{G2ip}}$, il nous suffit donc de montrer $\vdash_{\mathsf{G4ip}} = \Vdash$, autrement dit, G4ip est complet par rapport à la sémantique des arbres de Kripke. Dans la mesure où G4ip est analytique, notre approche consiste à prouver la validité et la co-validité de ce système par rapport à $\Vdash$.[29]

Proposition 3.3.7 *Pour tout arbre de Kripke $\mathcal{T} \in \mathcal{K}$, la validité sémantique $\Vdash_{\mathcal{T}}$ est close pour les règles de* G4ip.

Démonstration. Dans la preuve de la proposition 3.2.2, nous avons déjà montré que toutes les règles de G2i sont closes pour la sémantique algébrique. Or $\Vdash_{\mathcal{T}}$ est un cas particulier de sémantique algébrique dans l'algèbre de Heyting complète $(\downarrow\mathcal{T}, \prec^\star)$. Il ne reste donc que les règles $[\rightarrow^\star_L]$ à analyser. Nous pouvons prouver la validité sémantique de ces règles directement, mais nous pouvons aussi utiliser les constructions de la preuve de la proposition 3.3.1. Dans cette preuve, nous obtenons toutes les règles $[\rightarrow^\star_L]$ comme règles dérivées de règles de G2i et de la règle d'affaiblissement [weak] sémantiquement valide elle aussi. Or les règles dérivées de règles valides sont bien sûr valides. □

Nous étudions maintenant l'inversibilité sémantique des règles de G4ip. Nous nous plaçons dans un modèle particulier. Nous montrons que presque toutes les règles de G4ip sont strictement inversibles, autrement dit, un contre-modèle d'une prémisse est un contre-modèle de la conclusion.

Proposition 3.3.8 (Inversibilité stricte) *Soit $\mathcal{T} \in \mathcal{K}$ un arbre de Kripke. Alors toutes les règles de* G4ip *exceptées $[\vee^\star_R]$ et $[\rightarrow^4_L]$ sont inversibles par rapport à $\Vdash_{\mathcal{T}}$. Dans la règle $[\rightarrow^4_L]$, seule la première prémisse n'est pas inversible.*

[29] Nous tenons à insister sur le fait que dans cette section, le symbole $\Vdash$ ne représente pas la validité intuitionniste, i.e. $\vdash_{\mathsf{G2i}}$ mais la validité sémantique stricte dans les arbres de Kripke.

Démonstration. Étudions d'abord le cas des règles communes à G4ip et G2i. Les axiomes sont évidemment inversibles. Pour les autres, soit par exemple la règle $[\wedge_L]$. Il s'agit de montrer que de $\Gamma, A \wedge B \Vdash_{\mathcal{T}} G$, on peut déduire $\Gamma, A, B \Vdash_{\mathcal{T}} G$. Or $\mathcal{T} \vDash A \wedge B$ si et seulement si $\mathcal{T} \vDash A$ et $\mathcal{T} \vDash B$. Donc $\mathcal{T}$ est un modèle strict de $\Gamma, A \wedge B \vdash G$ si et seulement si c'est un modèle strict de $\Gamma, A, B \vdash G$. Pour la même raison, si $\mathcal{T}$ est un modèle strict $\Gamma \vdash A \wedge B$, alors c'est aussi un modèle strict de $\Gamma \vdash A$ et $\Gamma \vdash B$. Les autres règles $[\mathsf{T}_L]$, $[\vee_L]$ et $[\rightarrow_R]$ se traitent tout aussi simplement.

Nous étudions maintenant le cas des règles de G4ip qui ne sont pas des règles de G2i. Par exemple la règle $[\rightarrow^1_L]$. Or si on suppose $\mathcal{T} \vDash X$, alors $\mathcal{T} \vDash C \Leftrightarrow \mathcal{T} \vDash X \rightarrow C$. Donc $\Gamma, X, X \rightarrow C \Vdash_{\mathcal{T}} G$ si et seulement si $\Gamma, X, C \Vdash_{\mathcal{T}} G$.

Prenons maintenant la cas de la règle $[\rightarrow^2_L]$. Supposons $\mathcal{T} \vDash A \rightarrow (B \rightarrow C)$ et montrons $\mathcal{T} \vDash (A \wedge B) \rightarrow C$. Soit $\mathcal{T}' \leqslant \mathcal{T}$ un sous-arbre de $\mathcal{T}$ tel que $\mathcal{T}' \vDash A \wedge B$. Alors $\mathcal{T}' \vDash A$. Donc on peut en déduire $\mathcal{T}' \vDash B \rightarrow C$. De même, $\mathcal{T}' \vDash B$ et donc $\mathcal{T}' \vDash C$. Réciproquement, supposons maintenant $\mathcal{T} \vDash (A \wedge B) \rightarrow C$ et montrons $\mathcal{T} \vDash A \rightarrow (B \rightarrow C)$. Soit donc $\mathcal{T}' \leqslant \mathcal{T}$ tel que $\mathcal{T}' \vDash A$. Il s'agit de montrer $\mathcal{T}' \vDash B \rightarrow C$. Soit donc $\mathcal{T}'' \leqslant \mathcal{T}'$ tel que $\mathcal{T}'' \vDash B$, il s'agit maintenant de montrer $\mathcal{T}'' \vDash C$. Par monotonie (voir la proposition 3.2.12) comme $\mathcal{T}'' \leqslant \mathcal{T}'$ on a $\mathcal{T}'' \vDash A$. Donc $\mathcal{T}'' \vDash A \wedge B$. Or par transitivité, on a $\mathcal{T}'' \leqslant \mathcal{T}$ et donc $\mathcal{T}'' \vDash C$.

Nous ne présentons pas tous les cas qui mettent en œuvre des raisonnements similaires. Montrons pourquoi la première prémisse de $[\rightarrow^4_L]$ n'est pas inversible. Considérons l'instance

$$\frac{A, B \rightarrow A \vdash B \qquad \ldots}{(A \rightarrow B) \rightarrow A \vdash \mathsf{T}} \; [\rightarrow^4_L]$$

Il est clair que la conclusion est strictement valide dans n'importe quel arbre de Kripke alors que le séquent $A, B \rightarrow A \vdash B$ peut-être invalidé dans un arbre de Kripke à un seul nœud r où $r \nvDash B$ et $r \vDash A$, c'est-à-dire un contre-modèle au sens de la logique classique. □

Proposition 3.3.9 *Dans* $(\mathsf{G4ip}, \Vdash)$ *les séquents irréductibles sont de la forme*

$$X_1, \ldots, X_n, Y_1 \rightarrow D_1, \ldots, Y_p \rightarrow D_p, (A_1 \rightarrow B_1) \rightarrow C_1, \ldots, (A_k \rightarrow B_k) \rightarrow C_k \vdash K$$

avec $\{X_1, \ldots, X_n\} \cap \{Y_1, \ldots, Y_p, Z\} = \emptyset$, *et* $K \in \mathsf{Var} \cup \{\mathsf{F}\}$ *ou* $K \equiv A \vee B$.

Démonstration. Les séquents irréductibles sont ceux auxquels aucune règle inversible ne peut être appliquée à l'envers, autrement dit, ils ne sont la conclusion d'aucune des règles [id], $[\mathsf{F}_L]$, $[\mathsf{T}_\star]$ $[\wedge_\star]$, $[\vee_L]$, $[\rightarrow_R]$ et $[\rightarrow^i_L]$ pour $i \in \{1, 2, 3, 5, 6\}$ et $\star \in \{L, R\}$. Soit $\Gamma \vdash K$ un séquent irréductible :

- la **conclusion** K du séquent n'est ni une conjonction $\wedge$ ou T ni une implication $\rightarrow$ à cause des règles $[\wedge_R]$, $[\mathsf{T}_R]$ et $[\wedge_R]$ toutes les trois inversibles ;
- soit $M \in \Gamma$ une des **hypothèses** du séquent. Alors du fait des règles $[\mathsf{F}_L]$, $[\mathsf{T}_L]$, $[\wedge_L]$ et $[\vee_L]$, M est soit une variable, soit une implication. Si $M \equiv N \rightarrow C$ est une implication alors, N n'est ni une conjonction $\wedge$, ni une disjonction $\vee$, ni une constante T ou F à cause des règles $[\rightarrow^i_L]$ pour $i \in \{2, 3, 5, 6\}$. Si N est une variable alors elle n'apparaît pas dans Γ à cause de la règle $[\rightarrow^1_L]$. Sinon N est forcément une implication.

□

Proposition 3.3.10 *Dans* $(\mathsf{G4ip}, \Vdash)$ *les séquents atomiques sont de la forme*

$$X_1, \ldots, X_n, Y_1 \rightarrow C_1, \ldots, Y_p \rightarrow C_p \vdash K$$

avec $\{X_1, \ldots, X_n\} \cap \{Y_1, \ldots, Y_p, K\} = \emptyset$ *et* $K \in \mathsf{Var} \cup \{\mathsf{F}\}$.

Démonstration. Les séquents atomiques sont ceux auxquels aucune règle ne peut être appliquée à l'envers, autrement dit, ils ne sont la conclusion d'aucune règle. Donc ils sont aussi irréductibles et on peut se servir du résultat précédent. Les seules formules non atomiques qui peuvent apparaître dans un séquent atomique sont donc de la forme $Y \to C$ où Y est une variable qui n'apparaît pas dans les hypothèses à cause de la règle $[\to^1_L]$. Les formules atomiques pouvant apparaître dans le séquent sont forcément des variables à gauche du signe $\vdash$, à cause des règles $[\mathsf{F}_L]$ et $[\mathsf{T}_L]$ et, soit une variable soit F à droite de $\vdash$ à cause des règles $[\mathsf{T}_R]$ et $[\vee^\star_R]$. □

Proposition 3.3.11 (Co-validité) *Le système* G4ip *est co-valide par rapport à* $\Vdash$.

Démonstration. On applique le principe 1.6.1 en page 17 du chapitre 1. Montrons tout d'abord que les séquents atomiques sont invalides. Soit $X_1, \ldots, X_n, Y_1 \to C_1, \ldots, Y_p \to C_p \vdash K$ un séquent atomique. On considère l'arbre de Kripke à un seul nœud r tel que $r \not\Vdash K$, $r \not\Vdash Y_i$ et $r \Vdash X_j$ pour $i \in [1,p]$ et $j \in [1,n]$, c'est-à-dire $\mathcal{T} \triangleq (\{X_1, \ldots, X_n\}, \emptyset)$. Il apparaît clairement que c'est un contre-modèle strict du séquent atomique $X_1, \ldots, X_n, Y_1 \to C_1, \ldots, Y_p \to C_p \vdash K$ car $\mathcal{T} \not\Vdash K$.

On considère maintenant un séquent irréductible $X_1, \ldots, X_n, Y_1 \to D_1, \ldots, Y_p \to D_p, (A_1 \to B_1) \to C_1, \ldots, (A_k \to B_k) \to C_k \vdash K$. On définit $\Gamma \triangleq X_1, \ldots, X_n, Y_1 \to D_1, \ldots, Y_p \to D_p$ et $\Delta \triangleq (A_1 \to B_1) \to C_1, \ldots, (A_k \to B_k) \to C_k$. Notre séquent est donc $\Gamma, \Delta \vdash K$. On définit ensuite Δ_i pour $i \in [1,k]$ qui est Δ privé d'une instance de la formule $(A_i \to B_i) \to C_i$. La règle ayant $(A_i \to B_i) \to C_i$ comme formule principale s'écrit alors :

$$\frac{\Gamma, \Delta_i, A_i, B_i \to C_i \vdash B_i \qquad \ldots}{\Gamma, \Delta_i, (A_i \to B_i) \to C_i \vdash K} \; [\to^4_L]$$

Si K est atomique (donc dans $\mathsf{Var} \cup \{\mathsf{F}\}$) alors il n'y a pas d'autres règles applicables. Si par contre $K \equiv A \vee B$ alors on peut aussi appliquer les deux règles :

$$\frac{\Gamma, \Delta \vdash A}{\Gamma, \Delta \vdash A \vee B} \; [\vee^1_R] \qquad \frac{\Gamma, \Delta \vdash B}{\Gamma, \Delta \vdash A \vee B} \; [\vee^2_R]$$

C'est le cas que nous traitons dans cette preuve. Il s'agit de montrer que si chaque prémisse non-inversible provenant de ce séquent admet un contre-modèle strict alors le séquent lui aussi admet un contre-modèle strict. Soit donc par hypothèse $\mathcal{T}_i$ un contre-modèle strict de $\Gamma, \Delta_i, A_i, B_i \to C_i \vdash B_i$ pour chaque $i \in [1,k]$ et $\mathcal{T}_A$ et $\mathcal{T}_B$ deux contre-modèles stricts de $\Gamma, \Delta \vdash A$ et $\Gamma, \Delta \vdash B$ si K n'est pas atomique. On considère l'arbre de Kripke :

$$\mathcal{T} \triangleq (\{X_1, \ldots, X_n\}, \{\mathcal{T}_1, \ldots, \mathcal{T}_k, \mathcal{T}_A, \mathcal{T}_B\})$$

en faisant remarquer que nous considérons ici le cas où K n'est pas atomique. On vérifie tout d'abord que $\mathcal{T}$ est bien un arbre de Kripke. Il faut donc montrer $\mathcal{S}_\mathcal{T} \subseteq \mathcal{S}_{\mathcal{T}_i}$ pour chaque i, $\mathcal{S}_\mathcal{T} \subseteq \mathcal{S}_{\mathcal{T}_A}$ et $\mathcal{S}_\mathcal{T} \subseteq \mathcal{S}_{\mathcal{T}_B}$. Or comme $\mathcal{T}_i$ est un contre-modèle strict de $\Gamma, \Delta_i, A_i, B_i \to C_i \vdash B_i$, $\mathcal{T}_i$ force en particulier toutes les formules de Γ, qui contient les $X_1, \ldots, X_n$. De même pour $\mathcal{T}_A$ et $\mathcal{T}_B$. Nous avons bien un arbre de Kripke.

Il reste à montrer qu'il s'agit d'un contre-modèle strict du séquent initial $\Gamma, \Delta \vdash A \vee B$. Dans un premier temps, montrons que $\mathcal{T} \Vdash (A_i \to B_i) \to C_i$:

- Nous montrons d'abord que $\mathcal{T}_i \Vdash (A_i \to B_i) \to C_i$. Soit $\mathcal{T}' \leqslant \mathcal{T}_i$ tel que $\mathcal{T}' \Vdash A_i \to B_i$. Alors comme $\mathcal{T}_i \Vdash A_i$ on a par monotonie $\mathcal{T}' \Vdash A_i$ et donc $\mathcal{T}' \Vdash B_i$. Or on a aussi $\mathcal{T}_i \Vdash B_i \to C_i$ et par monotonie $\mathcal{T}' \Vdash B_i \to C_i$. On en déduit $\mathcal{T}' \Vdash C_i$;

- ensuite, si $i \neq j$, alors $\mathcal{T}_j$ force la formule $(A_i \to B_i) \to C_i$ car cette formule apparaît dans Δ_j ;
- de plus, il est clair qu'à la fois $\mathcal{T}_A$ et $\mathcal{T}_B$ forcent $(A_i \to B_i) \to C_i$;
- enfin, comme $\mathcal{T}_i \vDash A_i$ et $\mathcal{T}_i \nvDash B_i$, il vient $\mathcal{T}_i \nvDash A_i \to B_i$ et par monotonie $\mathcal{T} \nvDash A_i \to B_i$;
- on peut donc conclure que $\mathcal{T} \vDash (A_i \to B_i) \to C_i$.

Comme il est clair que $\mathcal{T}$ force toutes les formules de Γ, il s'en suit que $\mathcal{T}$ force toutes les formules de Γ, Δ. Comme $\mathcal{T}_A \nvDash A$ et $\mathcal{T}_B \nvDash B$, on obtient aussi $\mathcal{T} \nvDash A \vee B$. $\mathcal{T}$ est bien un contre-modèle strict de $\Gamma, \Delta \vdash A \vee B$.

On remarque que l'on a besoin de tous les sous-arbres $\mathcal{T}_i$ pour prouver $\mathcal{T} \vDash (A_i \to B_i) \to C_i$ car on a besoin de $\mathcal{T}_i$ pour montrer que $\mathcal{T} \nvDash A_i \to B_i$. Enfin, le cas où la conclusion K du séquent est atomique se traite de manière tout à fait similaire et on obtient un contre-modèle dans la mesure où $Z \notin \{X_1, \ldots, X_n\}$. □

Théorème 3.3.12 (Complétude et propriété des modèles finis) *Le système* G4ip *(aussi appelé* LJT*) est complet et la logique intuitionniste possède la propriété des modèles finis.*

Démonstration. Comme le système déductif $(\mathsf{G4ip}, \Vdash)$ est valide, co-valide et analytique, par application du théorème 1.5.4 en page 16, ce système est complet. On obtient donc

$$\Vdash \triangleq \bigcap_{\mathcal{T} \in \mathcal{K}} \Vdash_{\mathcal{T}} = \vdash_{\mathsf{G4ip}}$$

On combinant ce résultat à l'inégalité (3.3), on obtient la complétude de G4ip par rapport à la logique intuitionniste, i.e. $\vdash_{\mathsf{G4ip}} = \vdash_{\mathsf{G2i}}$, ainsi que la propriété des modèles finis pour la logique intuitionniste. En effet, la classe des modèles $\mathcal{K}$ des arbres de Kripke finis est une classe qui ne contient que des modèles finis et est complète pour LI car $\Vdash = \vdash_{\mathsf{G2i}}$. □

3.3.5 Extraction de contre-modèles

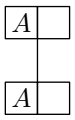
Nous rappelons que le principe 1.6.1 permet de simplifier la preuve de co-validité d'un système en élaguant les arbres de réfutations au niveau des nœuds où les propositions (des séquents ici) sont réductibles : on n'y conserve qu'une seule branche correspondant à l'une des prémisses d'une règle inversible.

On peut remarquer que la démonstration précédente consiste essentiellement à montrer comment à partir d'un tel arbre élagué on peut construire un contre-modèle de Kripke. La structure d'arbre du contre-modèle correspond en fait à celle de la réfutation, mais on n'y conserve que les nœuds à plusieurs fils (ceux qui correspondent aux séquents irréductibles.) Les autres nœuds sont identifiés à leur père.

Par exemple, considérons une réfutation de la formule $A \to ((A \to B) \to B) \to A \to B$. On lui associe, sur sa droite la version élaguée dans laquelle on applique uniquement les règles non-inversibles quand cela est absolument nécessaire, i.e. le séquent est irréductible. La branche de la règle $\boxed{[\to^4_L]}$ est donc supprimée :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A, A, B \to B \vdash B} \quad \dfrac{\dfrac{}{A, A, A, B \to B \vdash B}}{A, A, (A \to B) \to B \vdash B}\,[\to^4_L]}{A, (A \to B) \to B \vdash A \to B}\,\boxed{[\to^4_L]}}{A \vdash ((A \to B) \to B) \to A \to B}\,[\to_R]}{\vdash A \to ((A \to B) \to B) \to A \to B}\,[\to_R]}\ \text{et}\ [\to_R] \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{A, A, A, B \to B \vdash B}}{A, A, (A \to B) \to B \vdash B}\,[\to^4_L]}{A, (A \to B) \to B \vdash A \to B}\,[\to_R]}{A \vdash ((A \to B) \to B) \to A \to B}\,[\to_R]}{\vdash A \to ((A \to B) \to B) \to A \to B}\,[\to_R]$$

À partir de cette version élaguée, on obtient le contre-modèle. L'arbre de Kripke à un nœud $\mathcal{T}_0 \triangleq (\{A\}, \emptyset)$ est un contre-modèle de la feuille $A, A, A, B \to B \vdash B$. On prolonge cet arbre lorsque l'on rencontre la règle $[\to^4_L]$. Ce qui donne le contre-modèle $\mathcal{T}_1 \triangleq (\{A\}, \{\mathcal{T}_0\})$ représenté dans la figure ci-contre. On remarque que le contre-exemple n'est pas minimal.



3.4 Le partage de formules

Un des problèmes liés à l'implantation de G4ip est la duplication des sous-formules, comme par exemple B dans la règle $[\to^4_L]$ ou C dans la règle $[\to^3_L]$:

$$\frac{\Gamma, A, B \to C \vdash B \quad \ldots}{\Gamma, (A \to B) \to C \vdash G}\ [\to^4_L] \qquad \frac{\Gamma, A \to C, B \to C \vdash G}{\Gamma, (A \vee B) \to C \vdash G}\ [\to^3_L]$$

Dans cette section, nous cherchons à supprimer la première forme de duplication. Pour cela, nous étudions les séquents de la forme $\Gamma, \alpha \to C \vdash \alpha$ où α est une formule qui est « partagée » par une hypothèse et la conclusion. Cette idée est apparue pour la première fois dans [Hud 93] sous une forme légèrement différente. Nous la reprenons et fournissons une preuve sémantique de la complétude. Nous étudions un système de règles adapté aux séquents de cette forme qui préserve le lien sur la formule α. Nous montrons qu'il est possible de représenter LI par ce système en obtenant un résultat de complétude.

3.4.1 Les séquents avec partage

Nous développons ici un système logique spécialement adapté aux séquents avec partage qui permet, par conservation du lien sur la formule α, de supprimer la première forme de duplication. La proposition 3.4.4 montre que la validité intuitionniste est réductible au cas des séquents avec partage.

Définition 3.4.1 (Séquent avec partage) *Un **séquent avec partage** est une écriture de la forme*

$$\Gamma, \alpha \to C \vdash \alpha$$

*La **sémantique** d'un séquent avec partage est celle du séquent sous-jacent, c'est-à-dire $\Gamma, \alpha \to C \vdash \alpha$. La **complexité** d'un séquent avec partage est celle du multi-ensemble Γ, α, C, autrement dit, on ne compte α qu'une seule fois.*

Explicitons cette définition. La différence entre un séquent et un séquent avec partage est que l'on explicite le lien entre la formule α sous-formule de l'hypothèse $\alpha \to C$ et la conclusion α. Au niveau de la validité, ce lien n'a pas d'importance. Le séquent avec partage est valide si et seulement si le séquent « normal » l'est, c'est-à-dire :

$$\Gamma, \alpha \to C \Vdash \alpha \text{ si et seulement si } \Gamma, \alpha \to C \Vdash \alpha \tag{3.4}$$

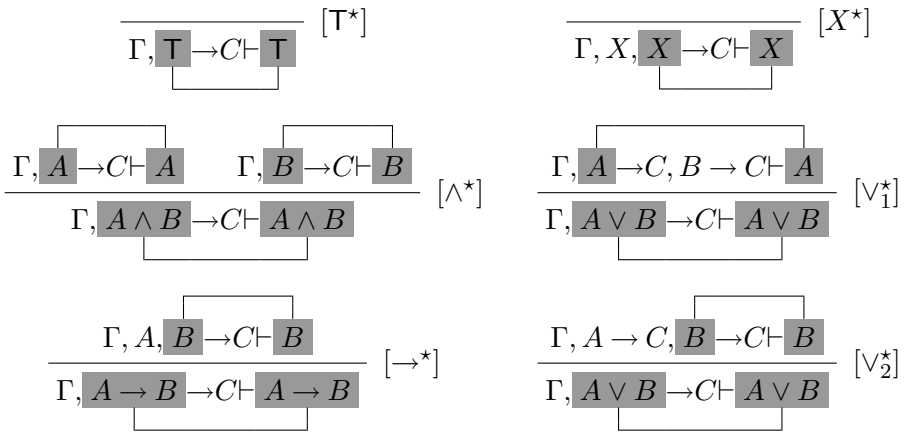


FIG. 3.4 – Des règles logiques pour les séquents avec partage.

Décomposition des séquents avec partage

En figure 3.4, nous donnons des règles de déductions pour les séquents avec partage. Ces règles sont dérivées de règles de G4ip.

Proposition 3.4.1 *Les règles de la figure 3.4 sont valides. De plus, mis à part les règles* $[\vee_1^\star]$ *et* $[\vee_2^\star]$*, elles sont aussi inversibles.*

Démonstration. La validité étant définie par effacement du lien, il suffit d'effacer ces liens dans les règles et de les étudier dans G4ip par exemple. Les règles $[\mathsf{T}^\star]$ et $[X^\star]$ apparaissent comme des cas particuliers d'axiomes de G4ip donc sont à la fois valides et inversibles.

Les deux règles $[\vee_1^\star]$ et $[\vee_2^\star]$ s'obtiennent par une simple combinaison des règles $[\to_L^3]$ et $[\vee_R^\star]$. Comme elles ne sont pas inversibles,[30] il n'y a rien d'autre à prouver les concernant.

Afin de simplifier les preuves qui vont suivre, nous introduisons deux nouvelles règles inverses l'une de l'autre et notées $[\to_L^r]$ et $[\to_L^i]$. Elles permettent la simplification d'hypothèses :

$$\frac{\Gamma, A, B \to C \vdash G}{\Gamma, A, (A \to B) \to C \vdash G}\ [\to_L^r] \qquad \frac{\Gamma, A, (A \to B) \to C \vdash G}{\Gamma, A, B \to C \vdash G}\ [\to_L^i]$$

Sous l'hypothèse A, nous pouvons simplifier $A \to B$ en B. Nous montrons la validité de ces deux règles, et par voie de conséquence leur inversibilité. Commençons par $[\to_L^r]$:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, A, B, A, B \to C \vdash B} \quad \dfrac{}{\Gamma, A, B, C \vdash C}}{\Gamma, A, B, (A \to B) \to C \vdash C}\,[\to_L^4]}{\Gamma, A, (A \to B) \to C \vdash B \to C}\,[\to_R] \qquad \dfrac{\Gamma, A, B \to C \vdash G}{\Gamma, A, (A \to B) \to C, B \to C \vdash G}\,[\text{weak}]}{\Gamma, A, (A \to B) \to C \vdash G}\,[\text{cut}]$$

Pour la validité de $[\to_L^i]$ (ou encore l'inversibilité de $[\to_L^r]$,) nous proposons la preuve :

[30] Nous ne montrons pas ici que les règles $[\vee_1^\star]$ et $[\vee_2^\star]$ ne sont pas inversibles. Cela n'est pas nécessaire pour le résultat qui nous intéresse. Bien sûr ce point est fondamental lorsqu'il s'agit de choisir un algorithme de recherche de preuve, voir section 3.4.4. On pourra facilement construire des contre-exemples à l'inversibilité de ces règles à l'aide du logiciel STRIP, voir section 3.5.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, A, B, C \vdash C}}{\Gamma, A, B, B \to C \vdash C}\,[\to^1_L]}{\Gamma, A, A \to B, B \to C \vdash C}\,[\to^1_L]}{\Gamma, A, B \to C \vdash (A \to B) \to C}\,[\to_R] \quad \dfrac{\Gamma, A, (A \to B) \to C \vdash G}{\Gamma, A, B \to C, (A \to B) \to C \vdash G}\,[\text{weak}]}{\Gamma, A, B \to C \vdash G}\,[\text{cut}]$$

On remarquera que nous utilisons la version généralisée (où X n'est pas forcément une variable) de la règle $[\to^1_L]$. Nous en avons déjà montré la validité. Appliquons maintenant la règle $[\to^r_L]$ et son inverse $[\to^i_L]$ pour prouver la validité de la règle $[\to^\star]$.

$$\dfrac{\dfrac{\Gamma, A, B \to C \vdash B}{\Gamma, A, (A \to B) \to C \vdash B}\,[\to^r_L]}{\Gamma, (A \to B) \to C \vdash A \to B}\,[\to_R]$$

Mais la règle $[\to^\star]$ est aussi inversible :

$$\dfrac{\dfrac{\dfrac{\Gamma, (A \to B) \to C \vdash A \to B}{\Gamma, A, (A \to B) \to C \vdash A \to B}\,[\text{weak}]}{\Gamma, A, B \to C \vdash A \to B}\,[\to^i_L] \quad \dfrac{\dfrac{}{\Gamma, A, B \to C, B \vdash B}}{\Gamma, A, B \to C, A \to B \vdash B}\,[\to^1_L]}{\Gamma, A, B \to C \vdash B}\,[\text{cut}]$$

Nous terminons avec la règle $[\wedge^\star]$. Nous commençons par fournir une preuve du séquent suivant $\Gamma, (A \to C) \to A, (B \to C) \to B, (A \wedge B) \to C \vdash A$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, A, C \to A, B, C \vdash C}}{\Gamma, A, C \to A, B, B \to C \vdash C}\,[\to^1_L] \text{ pour } B}{\Gamma, A, C \to A, (B \to C) \to B, B \to C \vdash C}\,[\to^1_L] \text{ pour } B \to C}{\Gamma, A, C \to A, (B \to C) \to B, A \to B \to C \vdash C}\,[\to^1_L] \quad \dfrac{}{\ldots, A \vdash A}}{\Gamma, (A \to C) \to A, (B \to C) \to B, A \to B \to C \vdash A}\,[\to^4_L]}{\Gamma, (A \to C) \to A, (B \to C) \to B, (A \wedge B) \to C \vdash A}\,[\to^2_L]$$

À partir du dernier séquent et deux preuves qui suivent :

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma, A \to C \vdash A}{\Gamma \vdash (A \to C) \to A}\,[\to_R]}{\Gamma, (B \to C) \to B \vdash (A \to C) \to A}\,[\text{weak}]}{\Gamma, (B \to C) \to B, (A \wedge B) \to C \vdash (A \to C) \to A}\,[\text{weak}] \quad \dfrac{\dfrac{\Gamma, B \to C \vdash B}{\Gamma \vdash (B \to C) \to B}\,[\to_R]}{\Gamma, (A \wedge B) \to C \vdash (B \to C) \to B}\,[\text{weak}]$$

on arrive, par double application de la règle de coupure [cut] à une preuve de $\Gamma, (A \wedge B) \to C \vdash A$. De la même manière, on peut obtenir une preuve de $\Gamma, (A \wedge B) \to C \vdash B$ ce qui nous donne la validité de la règle $[\wedge^\star]$.

On termine par l'inversibilité de la première prémisse de $[\wedge^\star]$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma, A, B, C \vdash C}}{\Gamma, A, B, A \to C \vdash C}\,[\to^1_L]}{\Gamma, A \wedge B, A \to C \vdash C}\,[\wedge_L]}{\Gamma, A \to C \vdash (A \wedge B) \to C}\,[\to_R] \qquad \dfrac{\Gamma, (A \wedge B) \to C \vdash A \wedge B \quad \dfrac{\dfrac{}{\Gamma, (A \wedge B) \to C, A, B \vdash A}}{\Gamma, (A \wedge B) \to C, A \wedge B \vdash A}\,[\wedge_L]}{\Gamma, (A \wedge B) \to C \vdash A}\,[\text{cut}]}{\dfrac{\Gamma, A \to C, (A \wedge B) \to C \vdash A}{\Gamma, A \to C \vdash A}\,[\text{weak}]}\,[\text{cut}]$$

L'inversibilité de la deuxième prémisse se prouve de façon analogue. □

Proposition 3.4.2 (Inversibilité stricte) *Les règles* $[\mathsf{T}^\star]$, $[X^\star]$, $[\wedge^\star]$, $[\to^\star]$ *sont strictement inversibles.*

Démonstration. Le cas des règles $[\mathsf{T}^\star]$ et $[X^\star]$ trivial. Le cas de la règle $[\to^\star]$ est plus intéressant. Soit $\mathcal{T}$ un contre-modèle strict de $\Gamma, A, B \to C \vdash B$ alors $\mathcal{T} \vDash A$ et $\mathcal{T} \nvDash B$ donc $\mathcal{T} \nvDash A \to B$. Montrons que $\mathcal{T} \vDash (A \to B) \to C$. Soit $\mathcal{T}' \leqslant \mathcal{T}$ un sous-arbre de $\mathcal{T}$. Alors si $\mathcal{T}' \vDash A \to B$, comme $\mathcal{T} \vDash A$ on a aussi $\mathcal{T}' \vDash A$ et donc $\mathcal{T}' \vDash B$. Comme on a aussi $\mathcal{T} \vDash B \to C$, il vient $\mathcal{T}' \vDash C$. Donc $\mathcal{T} \vDash (A \to B) \to C$. Nous avons donc un contre-modèle strict de $\Gamma, (A \to B) \to C \vdash A \to B$.

Considérons enfin le cas de la règle $[\wedge^\star]$. Soit $\mathcal{T}$ un contre-modèle strict de $\Gamma, A \to C \vdash A$. Alors on a $\mathcal{T} \nvDash A \wedge B$. Montrons que $\mathcal{T} \vDash (A \wedge B) \to C$. Soit $\mathcal{T}' \leqslant \mathcal{T}$ un sous-arbre de $\mathcal{T}$ et supposons $\mathcal{T}' \vDash A \wedge B$. Alors $\mathcal{T}' \vDash A$ et comme $\mathcal{T} \vDash A \to C$, il vient $\mathcal{T}' \vDash C$. Donc $\mathcal{T} \vDash (A \wedge B) \to C$. Nous avons bien en $\mathcal{T}$ un contre-modèle strict de $\Gamma, (A \wedge B) \to C \vdash A \wedge B$. □

3.4.2 Le système SLJ

Nous proposons maintenant un système de règles dénommé SLJ complet pour les séquents avec partage, voir figure 3.5 mais attention à la nouvelle notation : on n'écrit plus la conclusion α du séquent avec partage. Et pour conserver l'information du lien, nous marquons α d'une étoile :

$\Gamma, \alpha^\star \to C \vdash \blacksquare$ représente $\Gamma, \alpha \to C \vdash \alpha$

Lorsque l'on essaye de décomposer un séquent partagé en choisissant une formule principale, deux cas peuvent se présenter : soit la formule active est $\alpha^\star$ que ce soit parce que l'on essaye de décomposer la conclusion ou que l'on essaye de décomposer $\alpha^\star \to C$, soit $\alpha^\star$ n'est pas modifiée. On notera $\Gamma^\star, A \vdash \blacksquare$ un séquent avec partage lorsque que la formule marquée $\alpha^\star \to C$ fait partie du contexte Γ, par opposition au cas $\Gamma, \alpha^\star \to C \vdash \blacksquare$.

En figure 3.5 est présenté le système de règles SLJ pour les séquents avec partage. Sur la colonne de gauche ainsi que sur les deux premières lignes de la colonne de droite ($[\mathsf{T}_L]$ et $[\vee_L]$) on trouve exclusivement des règles pour lesquelles $\alpha^\star$ n'est pas active. Ces règles sont une simple réécriture des règles de G4ip pour le cas particulier d'un séquent avec partage.[31] Elles sont toutes valides et sémantiquement valides comme règles dérivées de règles sémantiquement valides. Par ailleurs toutes ces règles sont strictement inversibles[32] à l'exception notable de la règle $[\to^4_L]$

[31] Nous avons donc choisi de leur donner le même nom que les règles correspondantes de G4ip. Les noms des autres règles sont tous marqués d'une étoile pour mettre en valeur le fait que c'est la formule étoilée qui est active.

[32] Nous rappelons que cela signifie qu'un contre-modèle d'une prémisse quelconque est un contre-modèle de la conclusion.

Le système SLJ

$$\frac{}{\Gamma^\star, \mathsf{F} \vdash \blacksquare}\ [\mathsf{F}_L] \qquad \frac{\Gamma^\star \vdash \blacksquare}{\Gamma^\star, \mathsf{T} \vdash \blacksquare}\ [\mathsf{T}_L]$$

$$\frac{\Gamma^\star, A, B \vdash \blacksquare}{\Gamma^\star, A \wedge B \vdash \blacksquare}\ [\wedge_L] \qquad \frac{\Gamma^\star, A \vdash \blacksquare \qquad \Gamma^\star, A \vdash \blacksquare}{\Gamma^\star, A \vee B \vdash \blacksquare}\ [\vee_L]$$

$$\frac{\Gamma^\star, X, C \vdash \blacksquare}{\Gamma^\star, X, X \to C \vdash \blacksquare}\ [\to^1_L] \qquad \frac{}{\Gamma, X, X^\star \to C \vdash \blacksquare}\ [X^\star]$$

$$\frac{\Gamma^\star, A \to B \to C \vdash \blacksquare}{\Gamma^\star, (A \wedge B) \to C \vdash \blacksquare}\ [\to^2_L] \qquad \frac{\Gamma, A^\star \to C \vdash \blacksquare \qquad \Gamma, B^\star \to C \vdash \blacksquare}{\Gamma, (A \wedge B)^\star \to C \vdash \blacksquare}\ [\wedge^\star]$$

$$\frac{\Gamma, A, B^\star \to C \vdash \blacksquare \qquad \Gamma^\star, C \vdash \blacksquare}{\Gamma^\star, (A \to B) \to C \vdash \blacksquare}\ [\to^4_L] \qquad \frac{\Gamma, A, B^\star \to C \vdash \blacksquare}{\Gamma, (A \to B)^\star \to C \vdash \blacksquare}\ [\to^\star]$$

$$\frac{\Gamma^\star, A \to C, B \to C \vdash \blacksquare}{\Gamma^\star, (A \vee B) \to C \vdash \blacksquare}\ [\to^3_L] \qquad \frac{\Gamma, A^\star \to C, B \to C \vdash \blacksquare}{\Gamma, (A \vee B)^\star \to C \vdash \blacksquare}\ [\vee^\star_1]$$

$$\frac{\Gamma^\star \vdash \blacksquare}{\Gamma^\star, \mathsf{F} \to C \vdash \blacksquare}\ [\to^5_L] \qquad \frac{\Gamma, A \to C, B^\star \to C \vdash \blacksquare}{\Gamma, (A \vee B)^\star \to C \vdash \blacksquare}\ [\vee^\star_2]$$

$$\frac{\Gamma^\star, C \vdash \blacksquare}{\Gamma^\star, \mathsf{T} \to C \vdash \blacksquare}\ [\to^6_L] \qquad \frac{}{\Gamma, \mathsf{T}^\star \to C \vdash \blacksquare}\ [\mathsf{T}^\star]$$

FIG. 3.5 – Calcul des séquents avec partage SLJ pour la logique intuitionniste, LI.

bien sûr. Les autres règles sont les six règles de la figure 3.4 dont seules $[\vee_1^\star]$ et $[\vee_2^\star]$ ne sont pas strictement inversibles.

Un contre-modèle strict du séquent avec partage $\Gamma, \alpha^\star \to C \vdash \blacksquare$ est un contre-modèle strict du séquent $\Gamma, \alpha \to C \vdash \alpha$, c'est-à-dire un arbre de Kripke qui force les formules de Γ ainsi que $\alpha \to C$, mais par contre, ne force pas α. Nous prouvons la complétude de SLJ en nous basant sur la sémantique des arbres de Kripke.

Théorème 3.4.3 (Complétude et propriété des modèles finis) *Le système de règles* SLJ *pour les séquents avec partage est complet pour la sémantique des arbres de Kripke finis.*

Démonstration. Nous utilisons la même technique que pour G4ip en mettant en valeur les points de la démonstration qui changent. Le système SLJ est analytique, la preuve faite pour G4ip étant tout à fait adaptable. Nous rappelons que l'on ne mesure qu'une seule fois $\alpha^\star$ dans la complexité d'un séquent avec partage.

La validité sémantique de SLJ a été rappelé dans le paragraphe précédent. Il nous reste à en montrer la co-validité. La seule véritable différence entre G4ip et SLJ, par rapport à la preuve de co-validité, est que les séquents irréductibles sont plus nombreux. Il faut en effet ajouter $\Gamma, (A \vee B)^\star \to C \vdash \blacksquare$ à la liste des séquents irréductibles alors que dans le cas de G4ip, le séquent $\Gamma, (A \vee B) \to C \vdash A \vee B$ aurait été réductible par application de la règle $[\to^3_L]$. On note que $\Gamma, \mathsf{F}^\star \to C \vdash \blacksquare$ est aussi un séquent potentiellement irréductible ce qui n'est pas le cas de $\Gamma, \mathsf{F} \to C \vdash \mathsf{F}$ dans G4ip.

Il s'agit donc de revoir ce point de la preuve. Soit donc un séquent avec partage irréductible de la forme

$$\Gamma, \Delta, \alpha^\star \to C \vdash \blacksquare$$

où $\Gamma \equiv X_1, \ldots, X_n, Y_1 \to D_1, \ldots, Y_p \to D_p$, $\Delta \equiv (A_1 \to B_1) \to C_1, \ldots, (A_k \to B_k) \to C_k$ et $\alpha \in \mathsf{Var} \cup \{\mathsf{F}\}$ ou alors $\alpha \equiv A \vee B$. On rappelle aussi que $\alpha \notin \{X_1, \ldots, X_n\}$ car sinon $[X^\star]$ est applicable. Le cas $\alpha \equiv \mathsf{F}$ se traite comme le cas des variables (pas de règle $\star$ applicable) donc seul le cas $\alpha \equiv A \vee B$ distingue vraiment les deux preuves : on introduit donc $\mathcal{T}_A$ et $\mathcal{T}_B$ deux contre-modèles stricts de respectivement $\Gamma, A^\star \to C, B \to C \vdash \blacksquare$ et $\Gamma, A \to C, B^\star \to C \vdash \blacksquare$ en plus des contre-modèles $\mathcal{T}_i$ des $\Gamma, \Delta_i, A_i, B_i^\star \to C_i, (A \vee B) \to C \vdash \blacksquare$. Il s'agit de montrer que

$$\mathcal{T} \triangleq (\{X_1, \ldots, X_n\}, \{\mathcal{T}_1, \ldots, \mathcal{T}_k, \mathcal{T}_A, \mathcal{T}_B\})$$

est un contre-modèle de $\Gamma, \Delta, (A \vee B)^\star \to C \vdash \blacksquare$. Seul le cas de la formule $(A \vee B) \to C$ diffère. Chacuns des $\mathcal{T}_i$ force cette formule. Comme $\mathcal{T}_A$ et $\mathcal{T}_B$ forcent $A \to C$ et $B \to C$, ils forcent aussi $(A \vee B) \to C$. Enfin $\mathcal{T} \not\models A \vee B$ car $\mathcal{T}_A \not\models A$ et $\mathcal{T}_B \not\models B$. On peut donc en déduire que $\mathcal{T}$ est un contre-modèle strict du séquent avec partage $\Gamma, \Delta, (A \vee B)^\star \to C \vdash \blacksquare$.

Une fois la co-validité acquise, la complétude sémantique découle d'une simple application du théorème 1.5.4 page 16. □

3.4.3 Représentation de LI par SLJ

Dans cette section nous montrons comment il est possible de se contenter des séquents avec partage pour faire de la recherche de preuve en logique intuitionniste. Il existe un codage immédiat des séquents dans les séquents avec partage.

Proposition 3.4.4 *Le séquent avec partage* $\Gamma, A^\star \to \mathsf{T} \vdash \blacksquare$ *est valide si et seulement si le séquent* $\Gamma \vdash A$ *est valide.*

Démonstration. On obtient très facilement une preuve de $\Gamma \vdash A \to \mathsf{T}$ puis par la règle [cut], on peut éliminer la formule $A \to \mathsf{T}$. Pour la réciproque, il suffit d'utiliser la règle d'affaiblissement [weak]. □

3.4.4 Construction de contre-modèles

L'algorithme de construction de preuves ou de réfutations de la section 1.5 au chapitre 1 s'applique au système analytique SLJ. Comme ce système est aussi co-valide par rapport à la sémantique des arbres de Kripke finis, c.f. la preuve du théorème de complétude 3.4.3, nous pouvons en déduire un algorithme de construction de contre-modèles. Il s'agit essentiellement de construire une réfutation et d'en extraire un contre-modèle en tenant compte de l'inversibilité stricte de certaines prémisses ou règles.

Algorithme

Nous détaillons l'algorithme de construction d'une preuve ou d'un contre-modèle du séquent intuitionniste $\Gamma \vdash A$. Tout d'abord, on le transforme en séquent partagé et on obtient $\Gamma, A^\star \to \mathsf{T} \vdash \blacksquare$. On décrit ensuite la construction de la preuve ou du contre-modèle associé à ce séquent partagé. Cette procédure est déduite de la preuve de complétude du théorème 3.4.3. Les justifications de terminaison (analycité) et de correction (co-validité) se trouvent au sein de la preuve de ce théorème. Il s'agit d'un algorithme récursif qui s'arrête soit sur (la conclusion d') un axiome, soit sur un séquent atomique, c'est-à-dire quand plus aucune règle ne peut-être appliquée. Dans les autres cas, il se relance sur un ou plusieurs séquents plus simples :

- dans le cas d'un **séquent atomique**, on construit un contre-modèle classique, c'est-à-dire un arbre de Kripke qui n'est composé que d'une feuille ;
- dans le cas d'un **axiome**, on construit la preuve qui n'a qu'une seule règle, cet axiome ;
- si une **règle inversible** s'applique, autrement dit, si le séquent est réductible, alors on choisit une telle règle et on relance l'algorithme sur chacune de ses prémisses. Deux cas se présentent : soit l'algorithme renvoie une preuve pour toutes les prémisses et dans ce cas on obtient une preuve du séquent en cours, soit l'algorithme renvoie un contre-modèle pour au moins l'une d'entre elles et cet arbre de Kripke est aussi un contre-modèle du séquent en cours, par inversibilité stricte. Il n'est pas nécessaire ici de faire un retour arrière et d'essayer d'autres règles, c'est l'intérêt de l'inversibilité.
- si **aucune règle inversible** ne s'applique alors on applique l'algorithme à toutes les prémisses inversibles restantes, c'est-à-dire celles de règles non-inversibles. Dans le cas de SLJ, il ne reste que la prémisse droite de la règle $[\to^4_L]$. Mais attention, plusieurs instances de cette règle peuvent être applicables.
  - Soit l'une de ces prémisses fournit un contre-modèle auquel cas, c'est aussi un contre-modèle du séquent en cours ;
  - soit toutes ces prémisses inversibles admettent une preuve. Passons aux autres prémisses, c'est-à-dire les prémisses gauches des instances de $[\to^4_L]$ et les prémisses des instances de $[\vee^\star_1]$ et $[\vee^\star_2]$.[33] On leur applique l'algorithme récursivement ;
  - si pour l'une des instances de règles, la prémisse non inversible a une preuve (donnée par l'algorithme) alors on obtient une preuve du séquent en cours en la combinant avec la preuve de la prémisse inversible ;

[33] On notera que dans les cas de SLJ et LJT, on a exactement une prémisse non inversible par règle non-inversible.

– sinon pour chaque instance de règle non-inversible, la prémisse non inversible admet un contre-modèle, donné par l'algorithme, et en combinant tous ces contre-modèles au dessus d'une racine commune, on obtient un contre-modèle du séquent en cours.

Un exemple

Nous présentons en guise d'exemple la construction d'un contre-modèle de la formule $(\neg\neg X \rightarrow X) \vee \neg X$. On considère le séquent avec partage $((\neg\neg X \rightarrow X) \vee \neg X)^\star \rightarrow \mathsf{T} \vdash \blacksquare$. On lance l'algorithme de construction. Nous donnons la trace utile de cet algorithme qui est un arbre de réfutation partiel et commentons chaque étape en figure 3.6. On remarquera que l'on obtient le contre-modèle $(\{\}, \{\mathcal{T}_5, \mathcal{T}_{10}\})$ qui n'est pas minimal, car $\mathcal{T}_5$ est aussi un contre-modèle de $(\neg\neg X \rightarrow X) \vee \neg X$.

L'algorithme présenté ne construit pas des contre-modèles minimaux. En effet, il arrive que certaines parties des contre-modèles générés soient dupliquées. Il est toutefois possible d'appliquer des réductions à ces contre-modèles, c'est-à-dire des transformations qui diminuent la taille de l'arbre de Kripke. Cependant, la notion de contre-modèle minimal n'est pas triviale à définir et surtout ne paraît pas très naturelle : elle ne semble pas correspondre à un ordre sur les arbres de Kripke. Ainsi un contre-modèle de taille minimale n'a pas forcément de rapport structurel (sous-arbre, inclusion...) avec un autre contre-modèle. Il peut d'ailleurs exister plusieurs contre-modèles de taille minimale pour une formule donnée. La réduction des contre-modèles est un point sur lequel des recherches sont encore en cours.

De la réfutation au contre-modèle de Kripke

Il est aussi possible de voir cet algorithme de construction de contre-modèle comme la combinaison d'un algorithme de construction de réfutations (déjà décrit au chapitre 1) avec l'extraction d'un contre-modèle à partir d'une réfutation, extraction dont nous avons donné un exemple en section 3.3.5. On rappelle que cette extraction consiste tout d'abord à éliminer les branches d'une réfutation qui ne correspondent pas à une prémisse inversible : si un séquent possède un fils qui correspond à une prémisse inversible, alors on élimine tous les autres fils ainsi que les sous-arbres associés. On identifie ensuite tous ces pères avec leur (unique) fils. On obtient le squelette de l'arbre de Kripke, les valuations pour les variables étant données par les variables présentes en tant qu'hypothèses dans les séquents associés aux nœuds de l'arbre de réfutation.

L'intérêt d'un algorithme combiné plutôt que l'extraction d'un contre-modèle à partir d'une réfutation réside dans le fait que le contre-modèle est généralement bien plus simple (en terme de taille) que la réfutation qui devient vite énorme, même pour un séquent de petite taille.

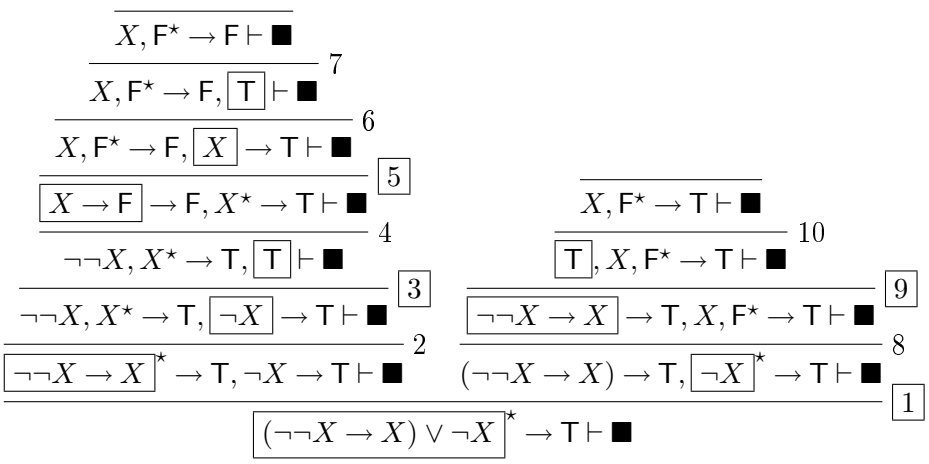
3.5 Le système STRIP

Dans cette section, nous décrivons une implantation du système SLJ. Cette implantation appelée STRIP est disponible en ligne à l'adresse

`http://www.loria.fr/~larchey/STRIP`

Ce logiciel est écrit en langage C et distribué sous une licence (GPL) qui en fait un logiciel libre. Nous n'entrons pas dans les détails mais discutons des différents points ayant orienté les choix d'implantations et les conséquences en termes de ressources et de performances.

Nous venons de voir dans la section précédente comment il est possible de se débarrasser de la duplication de la sous-formule B introduite par la règle $[\rightarrow^4_L]$ en considérant des séquents avec



Les $\boxed{\text{formules principales}}$ ont été encadrées ce qui permet d'identifier la/les règle(s) appliquée(s). Elle(s) est/sont aussi rappelée(s) dans le commentaire. Le numéro qui se trouve à coté de chaque étape correspond au commentaire. Les numéros encadrés correspondent à des séquents de conclusion $\boxed{\text{irréductibles.}}$

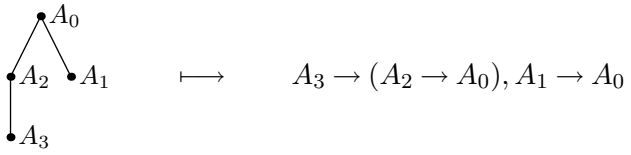
1. Le séquent de conclusion $(\neg\neg X \to X) \vee \neg X^\star \to \mathsf{T} \vdash \blacksquare$ est irréductible. Seules les deux règles non-inversibles $[\vee_1^\star]$ et $[\vee_2^\star]$ peuvent être appliquées et elles n'ont qu'une seule prémisse chacune, cette prémisse étant non-inversible. Nous verrons par la suite que pour la branche gauche (règle $[\vee_1^\star]$), on obtient le contre-modèle $\mathcal{T}_5$ et pour la branche droite (règle $[\vee_2^\star]$) on obtient le contre-modèle $\mathcal{T}_{10}$. Par combinaison, on obtient $(\{\}, \{\mathcal{T}_5, \mathcal{T}_{10}\})$ comme contre-modèle ;
2. le séquent $(\neg\neg X \to X)^\star \to \mathsf{T}, \neg X \to \mathsf{T} \vdash \blacksquare$ est réductible par la règle inversible $[\to^\star]$;
3. le séquent $\neg\neg X, X^\star \to \mathsf{T}, \neg X \to \mathsf{T} \vdash \blacksquare$ est irréductible mais la prémisse droite de la règle $[\to_L^4]$ est inversible. On essaye donc d'abord de construire un contre-modèle de cette prémisse droite : c'est possible et on obtient le contre-modèle fournit par $\boxed{5}$, c'est-à-dire $\mathcal{T}_5$;
4. on réduit le séquent par la règle inversible $[\mathsf{T}_L]$;
5. le séquent $(X \to \mathsf{F}) \to \mathsf{F}, X^\star \to \mathsf{T} \vdash \blacksquare$ est irréductible. Seule la règle $[\to_L^4]$ est applicable. La prémisse droite de cette règle vaut $\mathsf{F}, X^\star \to \mathsf{T} \vdash \blacksquare$ et est donc valide : c'est un axiome. Par conséquent, nous cherchons à construire un contre-modèle de la prémisse gauche : on obtient celui de 7 ce qui nous donne $\mathcal{T}_5 \triangleq (\{\}, \{\mathcal{T}_7\})$ comme contre-modèle ;
6. le séquent de conclusion est réductible par la règle $[\to_L^1]$;
7. le séquent de conclusion est réductible par la règle $[\mathsf{T}_L]$. On obtient un séquent atomique ayant la feuille $\mathcal{T}_7 \triangleq (\{X\}, \{\})$ pour contre-modèle.
8. Le séquent de conclusion est réductible par la règle $[\to^\star]$;
9. le séquent de conclusion $(\neg\neg X \to X) \to \mathsf{T}, X, \mathsf{F}^\star \to \mathsf{T} \vdash \blacksquare$ est irréductible mais on essaye tout d'abord la prémisse droite de la règle $[\to_L^4]$ qui est la seule applicable. Nous en déduisons le contre-modèle obtenu en 10, c'est-à-dire $\mathcal{T}_{10}$;
10. le séquent est réductible par la règle $[\mathsf{T}_L]$. On obtient un séquent atomique ayant la « racine-feuille » $\mathcal{T}_{10} \triangleq (\{X\}, \{\})$ pour contre-modèle.

FIG. 3.6 – Construction d'un contre-modèle à $(\neg\neg X \to X) \vee \neg X$.

partage. Nous nous intéressons maintenant à la duplication de la sous-formule C introduite par la règle

$$\frac{\Gamma, A{\rightarrow}\boxed{C}, B{\rightarrow}\boxed{C} \vdash G}{\Gamma, (A \vee B) \rightarrow C \vdash G}\ [{\rightarrow}^3_L]$$

La solution que nous proposons est de ne plus représenter les hypothèses sous la forme d'un multi-ensemble de formules mais d'une forêt d'arbres. Les arêtes représentent des implications logiques et chaque branche représente une formule, l'arbre représentant donc un multi-ensemble de formules. Par exemple l'arbre suivant représente le multi-ensemble des deux formules $A_3 \rightarrow (A_2 \rightarrow A_0)$ et $A_1 \rightarrow A_0$:



On remarque que la formule A_0 est partagée dans la représentation sous forme d'arbres et dupliquée dans la représentation des séquents à plat.

3.5.1 Les séquents sont des forêts

Le système STRIP est fondé sur la représentation des séquents sous forme de forêts et les règles sont dérivées du système SLJ. On peut voir SLJ comme un système intermédiaire entre G4ip (alias LJT) et une implantation de ce système qui évite les duplications.

Définition 3.5.1 ($\mathcal{T}$-séquent) *Un **$\mathcal{T}$-séquent** est un multi-ensemble d'arbres dont les nœuds sont indexés par des formules. La formule **partagée** avec la conclusion est l'une de ses feuilles (strictes) marquée d'une étoile $\star$.*

Nous expliquons maintenant comment associer un séquent partagé à tout $\mathcal{T}$-séquent. Nous donnons une description informelle car le principe de la conversion est très simple et une description formelle ne ferait que compliquer inutilement. Soit $F_0, \ldots, F_l$ la liste des formules rencontrées le long de la branche l d'un des arbres d'un $\mathcal{T}$-séquent. F_l est la formule indexant la feuille (éventuellement F_l est marquée d'une étoile) et F_0 indexe la racine de cet arbre. On associe à cette branche la formule $F_l{\rightarrow}(\cdots{\rightarrow}F_0)$ interprétant les arêtes comme des implications $\rightarrow$. Le $\mathcal{T}$-séquent $\mathcal{T}$ est interprété comme l'union de toutes les formules associées aux feuilles (ou branches) de cette forêt :

$$\mathcal{T} \longmapsto \{F_l \rightarrow \cdots \rightarrow F_0 \mid l \text{ est une feuille de } \mathcal{T}\}$$

On remarque que la formule partagée est toujours à la gauche d'une implication et est donc de la forme $F_l^\star \rightarrow (\cdots \rightarrow F_0)$ ce qui justifie la définition : la formule étoilée correspond à une feuille qui n'est pas une racine en même temps.

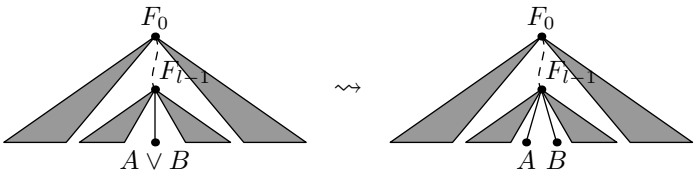
3.5.2 Ce que deviennent les règles de SLJ

Le point fondamental est que dans cette interprétation la duplication introduite par la règle $[{\rightarrow}^3_L]$ (et aussi $[\vee^\star_1]$, $[\vee^\star_2]$ dans SLJ) se traduit par un partage le long d'une branche. En effet si

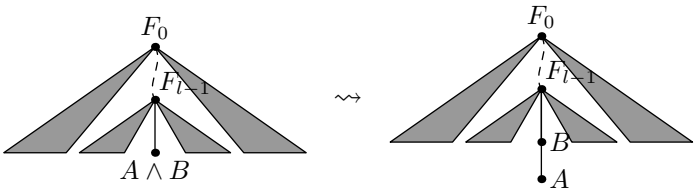
on suppose que la feuille F_l est de la forme $A \vee B$, on obtient une règle $[\rightarrow^4_L]$ de la forme

$$\frac{\ldots, A \to (F_{l-1} \to \cdots \to F_0), B \to (F_{l-1} \to \cdots \to F_0) \vdash \ldots}{\ldots, (A \vee B) \to (F_{l-1} \to \cdots \to F_0) \vdash \ldots} \; [\rightarrow^3_L]$$

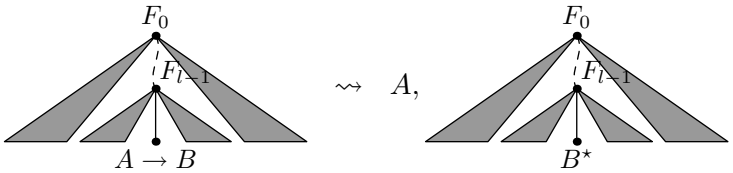
On s'aperçoit qu'il est possible de représenter les deux formules $A \to (F_{l-1} \to \cdots \to F_0)$ et $B \to (F_{l-1} \to \cdots \to F_0)$ sans la moindre duplication, en se servant du partage naturel le long de la branche $F_{l-1}, \ldots, F_0$. Ceci peut-être représenté sous forme d'arbre de la manière suivante :



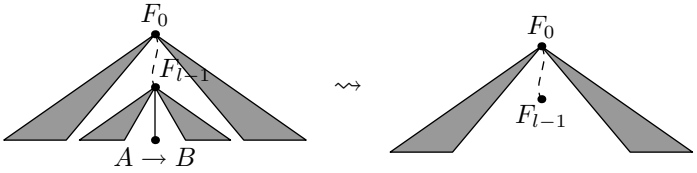
Toutes les règles logiques de SLJ peuvent se ré-interpréter sur cette nouvelle structure de $\mathcal{T}$-séquent. Il serait très fastidieux de décrire toutes ces règles à cause de la structure d'arbre. Les opérations mises en œuvre sur les arbres sont des opérations au niveau des feuilles. Dans l'exemple précédent on a vu que la feuille indexée par $A \vee B$ est dédoublée « horizontalement. » Dans le cas de la règle $[\rightarrow^2_L]$ la feuille indexée par $A \wedge B$ est dédoublée « verticalement » :



Enfin, le cas de l'implication au niveau d'une feuille est aussi intéressant. Si $A \to B$ se trouve au niveau d'une feuille alors d'après la règle $[\rightarrow^4_L]$, A sort de l'arbre et devient une nouvelle racine alors que $B^\star$ remplace tout simplement $A \to B$:



Ceci concerne la premisse gauche de la règle $[\rightarrow^4_L]$. Le traitement de la prémisse droite est un peu particulier. En effet, on remarque que tous les descendants du nœud F_{l-1} sont éliminés en même temps que la feuille $A \to B$:



A priori, la prémisse gauche de la règle $[\rightarrow^4_L]$ indique seulement qu'il faut introduire une feuille indexée par F_{l-1} :

$$\frac{\ldots \qquad \ldots, F^i_l \to F_{l-1} \to \cdots \to F_0, \ldots, F_{l-1} \to \cdots \to F_0, \ldots}{\ldots, F^i_l \to F_{l-1} \to \cdots \to F_0, \ldots, (A \to B) \to F_{l-1} \to \cdots \to F_0, \ldots} \; [\rightarrow^4_L]$$

Cette action aurait pour effet une duplication de la feuille, ce qui n'est pas souhaitable. Mais on peut alors utiliser plusieurs fois la règle inversible[34] suivante :

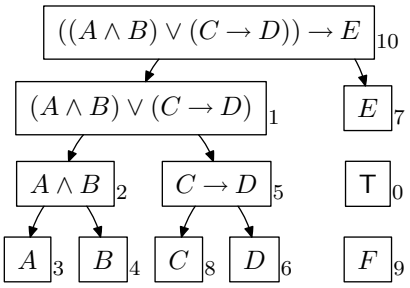
$$\frac{\Gamma, A_0 \vdash G}{\Gamma, A_l \to \cdots \to A_0, A_0 \vdash G}$$

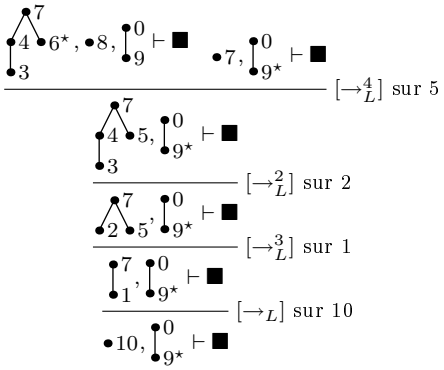
pour éliminer tous les descendants de F_{l-1}.[35] C'est en ce point particulier que le système STRIP se distingue d'une implantation triviale de SLJ. Cette élimination de tout un sous-arbre est non seulement nécessaire mais aussi très utile du point de vue de la complexité car elle peut réduire de manière considérable la taille du $\mathcal{T}$-séquent.

Pour finir, on remarquera que les règles préservent la polarité des sous-formules. Les racines correspondent à une polarité négative et les autres nœuds à une polarité positive de la sous-formule correspondante.

3.5.3 Un exemple de recherche de preuve

Nous n'entrerons pas plus dans les détails de la lecture des règles de SLJ sur les $\mathcal{T}$-séquents. Nous préférons présenter un exemple de recherche de preuve dans le système des $\mathcal{T}$-séquents. Cet exemple illustre la décomposition des arbres suivant les règles de SLJ. Soit à prouver le séquent $((A \wedge B) \vee (C \to D)) \to E \vdash F$. Nous transformons ce séquent en séquent avec partage ce qui donne $((A \wedge B) \vee (C \to D)) \to E, F^\star \to \mathsf{T} \vdash \blacksquare$.





Pour une question de place, nous avons codé les formules suivant le graphe de parenté ci-dessus à gauche.[36] Ainsi le code 7 représente la formule E alors que 5 représente $C \to D$. Ci-dessus à droite, on trouve l'arbre de preuve (partiel) pour le séquent $10, 9^\star \to 0 \vdash \blacksquare$. Nous commentons cette preuve.

- On commence par réécrire la formule $10 \equiv 1 \to 7$ sous forme d'un arbre à deux nœuds dont la racine est indexée par 7 et le fils par 1. Cette transformation ne change pas le sens du séquent puisque l'interprétation de la branche de feuille 1 est exactement $1 \to 7$.
  Cette règle que l'on note $[\to_L]$ est nouvelle dans STRIP par rapport à SLJ. Elle est bien sûr à la fois valide et inversible puisque l'interprétation du $\mathcal{T}$-séquent est **inchangée**. Elle ne s'applique qu'aux « racines-feuilles » c'est-à-dire les arbres à un seul nœud ;
- on applique ensuite la règle inversible $[\to^3_L]$ à la formule $1 \equiv 2 \vee 5$ ce qui nous donne un dédoublement horizontal de la feuille 1 en 2 et 5. L'instance de règle correspondante de

[34] On peut par exemple prouver son inversibilité forte dans la mesure où $A_0 \Vdash A_l \to \cdots \to A_0$.
[35] On notera que l'on utilisera une instance de la règle où $A_0 \equiv F_{l-1} \to \cdots \to F_0$.
[36] On remarquera que dans le programme STRIP, les formules sont codées de la même manière.

SLJ est :

$$\frac{2\rightarrow \boxed{7}, 5\rightarrow \boxed{7}, 9^\star \rightarrow 0 \vdash \blacksquare}{\boxed{2 \vee 5} \rightarrow 7, 9^\star \rightarrow 0 \vdash \blacksquare}\ [\rightarrow^3_L]$$

On rappelle que dans la représentation sous forme d'arbres, la formule 7 se trouve **partagée** sur les branches qui mènent aux feuilles 2 et 5. Ce fait est souligné par le lien supérieur entre les deux instances de 7, tandis que l'encadré marque la formule active ;

– vient ensuite la règle inversible $[\rightarrow^2_L]$ appliquée à la feuille $2 \equiv 3 \wedge 4$. On obtient un dédoublement vertical où 3 devient une nouvelle feuille[37] de père 4 qui vient remplacer 2. Dans SLJ :

$$\frac{3 \rightarrow (4\rightarrow \boxed{7}), 5\rightarrow \boxed{7}, 9^\star \rightarrow 0 \vdash \blacksquare}{\boxed{3 \wedge 4}\rightarrow \boxed{7}, 5\rightarrow \boxed{7}, 9^\star \rightarrow 0 \vdash \blacksquare}\ [\rightarrow^2_L]$$

– on se retrouve avec un $\mathcal{T}$-séquent où les feuilles sont 3, $5 \equiv 8 \rightarrow 6$ et $9^\star$. Donc nous avons un $\mathcal{T}$-séquent irréductible. Comme 3 et 9 sont atomiques, seule une règle peut-être appliquée : $[\rightarrow^4_L]$ sur la formule 5. Dans SLJ :

$$\frac{3 \rightarrow (4\rightarrow \boxed{7}), 8, 6^\star\rightarrow \boxed{7}, 9 \rightarrow 0 \vdash \blacksquare \qquad 3 \rightarrow (4 \rightarrow 7), 7, 9^\star \rightarrow 0 \vdash \blacksquare}{3 \rightarrow (4\rightarrow \boxed{7}), \boxed{8 \rightarrow 6}\rightarrow \boxed{7}, 9^\star \rightarrow 0 \vdash \blacksquare}\ [\rightarrow^4_L]$$

– on constate cependant que la prémisse droite diffère dans le cas STRIP par rapport au cas de SLJ. En effet, la formule $3 \rightarrow (4 \rightarrow 7)$ a été supprimée dans STRIP. Ceci correspond à l'élimination d'un sous-arbre dont nous avons parlé en section 3.5.2. **Cette simplification évite la duplication** de 7 qui apparaît dans SLJ et qu'il ne serait pas possible « d'absorber » dans la structure d'arbre car 7 devrait apparaître comme un arbre à un seul nœud dans la forêt.

3.5.4 Complétude et complexité

Dans cette section nous énonçons des résultats sans les démontrer en détail. La démonstration est techniquement simple à partir des résultats sur SLJ mais implique l'explicitation des règles sur les $\mathcal{T}$-séquents ce que nous avons choisi de ne pas faire. Ces règles sont bien sûr explicitées dans le programme STRIP.

Théorème 3.5.1 *Le système* STRIP *est complet pour la logique intuitionniste.*

On peut voir STRIP comme un implantation fidèle et efficace de SLJ,[38] lui-même complet pour la logique intuitionniste. La propriété fondamentale du système STRIP est que plus aucune duplication de sous-formule n'apparaît dans les règles du système, grâce aux deux formes de partage.

[37] On pourra noter que ce choix de mettre 3 en feuille plutôt que 4 est tout à fait arbitraire. On aurait très bien pu faire le choix inverse. Un ordonnancement variable peut faire partie d'une stratégie plus élaborée, voir section 3.5.5.

[38] Modulo l'élimination des sous-arbres qui correspond à l'application multiple d'une règle strictement inversible.

Théorème 3.5.2 (Analycité) *La profondeur de l'espace de recherche est majorée par la taille du séquent à prouver. Par conséquent, c'est aussi le cas de la profondeur des preuves et des réfutations.*

Démonstration. On mesure la complexité d'un $\mathcal{T}$-séquent par la somme de complexité des formules qui indexent les nœuds de la forêt. Comme il n'y a aucune duplication et qu'au moins une formule est décomposée pour chaque prémisse, la complexité décroît strictement lorsqu'on passe de la conclusion d'une règle à l'une des prémisses. La hauteur maximale d'une branche est donc majorée par la complexité de la racine, c'est-à-dire celle du séquent que l'on cherche à prouver ou réfuter. □

Par rapport à la preuve d'analycité de G4ip ou SLJ, celle de STRIP est beaucoup plus simple car une mesure de complexité linéaire suffit alors que nous avions utilisé des ordinaux dans les deux autres cas. Le partage sous forme d'arbre rend le système plus difficile à manipuler mais sa complexité algorithmique est moindre.

Théorème 3.5.3 (Propriété des modèles finis) *Il existe une procédure qui construit soit une preuve, soit un contre-modèle de n'importe quel $\mathcal{T}$-séquent.*

Démonstration. Il suffit de reprendre l'algorithme décrit en section 3.4.4 et de l'adapter à la représentation sous forme d'arbres des séquents. La partie qui correspond à la construction de contre-modèles est exactement la même car par rapport à SLJ, STRIP ne rajoute que des règles strictement inversibles. □

3.5.5 Des critères pour les choix d'implantation

Dans cette section, nous évoquons certains critères ayant déterminés nos choix d'implantation. Nous montrons comment l'absence de duplication dans STRIP permet un grand nombre d'optimisations autant au niveau de la gestion de ressources spatiales (espace mémoire) que temporelles (complexité en temps de la recherche de preuve.)

Les problèmes relatifs aux stratégies

Une stratégie consiste à choisir, pour un séquent donné, l'ordre dans lequels on va appliquer les règles de décomposition. C'est donc d'abord un choix de la formule active, et ensuite un choix d'ordre sur les prémisses, s'il y en a plusieurs possibles. Par exemple, dans le séquent avec partage suivant, où le lien supérieur représente le partage de la formule C au niveau d'une racine commune de la structure d'arbre,

$$K^\star \to \mathsf{T}, X, \boxed{X} \to A, \boxed{A \wedge B} \to \boxed{C}, \boxed{A \vee B} \to \boxed{C} \vdash \blacksquare$$

il y a trois formules principales possibles. $\boxed{X}$ correspond à l'application de la règle $[\to^1_L]$, $\boxed{A \wedge B}$ correspond à la règle $[\to^2_L]$ et $\boxed{A \vee B}$ à la règle $[\to^3_L]$. La détermination de ces formules implique un parcours du séquent. Ici, on examine les formules $K^\star$, X, X, $A \wedge B$ et $A \vee B$.

Or, non seulement ce parcours peut prendre du temps lorsque le séquent est de grande taille mais en plus, il doit être fait à chaque étape. Supposons que l'on décide d'appliquer la règle $[\to^3_L]$ par exemple. On obtient alors le séquent

$$K^\star \to \mathsf{T}, X, \boxed{X} \to A, \boxed{A \wedge B} \to \boxed{C}, A \to \boxed{C}, B \to \boxed{C} \vdash \blacksquare$$

qui est la prémisse correspondant à la règle $[\rightarrow^3_L]$. Or on constate que l'on est à nouveau obligé de parcourir les formules $K^\star$, X, X, $A \wedge B$ pour déterminer les formules actives.

Minimiser le plus possible le nombre de nœuds de la forêt visités pour raccourcir le temps passé à choisir la formule active est l'un des critères majeurs qui ont déterminé la structure de donnée la plus adaptée à la représentation de l'arbre.

D'autre part, le choix d'une stratégie est très important pour l'efficacité globale de la recherche de preuve. Dans [LW 00b], nous avons comparé la stratégie qui consiste à toujours choisir la première formule active rencontrée (la plus économe de point de vue du temps qui lui est consacré) à une stratégie plus sophistiquée qui tient compte de l'inversibilité des règles, du nombre de prémisses, retardant le plus possible l'application des règles qui semblent les plus coûteuses a priori, par exemple les règles à deux prémisses qui « dupliquent » l'espace de recherche de preuve. Nous avons constaté que la deuxième stratégie, qui est plus coûteuse en temps de parcours des formules, permet en revanche de grandement réduire l'espace de recherche de preuves. Ce dernier critère a une influence beaucoup plus grande sur les performances globales.

Si on désire construire des contre-modèles, on ne peut pas appliquer n'importe quelle stratégie. Les choix doivent respecter l'ordre partiel décrit dans l'algorithme de la section 3.4.4 : en particulier il impose d'appliquer d'abord toutes les règles inversibles avant de pouvoir appliquer une règle non-inversible. Par exemple, ce n'est pas ce que fait la stratégie élémentaire qui choisit la première formule active (feuille) rencontrée. Cependant, c'est le cas de la deuxième stratégie, plus sophistiquée.

Nous avons donc choisi une représentation qui ne limite pas trop les choix de stratégies. Il est tout à fait possible d'en implanter plusieurs au sein du prouveur et de les comparer. Par contre, le choix de représentation des données ne peut pas vraiment être découplé du choix de la stratégie, si on veut des performances optimales.

La gestion des ressources

Du point de vue de la gestion des ressources, le gros avantage de STRIP sur SLJ est que les (sous-)formules ne sont plus dupliquées. Ce simple fait permet d'effectuer des choix d'implantation qui ne seraient tout simplement pas envisageables autrement. Supposons que l'on travaille sur une formule logique (ou un séquent) de taille n. En numérotant les sous-formules, on peut les identifier par des nombres stockés sur un espace mémoire de taille $\log n$.

Par exemple, il est possible de prévoir à l'avance l'espace mémoire qui sera nécessaire à l'exécution de l'algorithme de recherche de preuve. Pour se repérer dans cet espace, l'information qui est nécessaire est la branche courante de cet arbre de recherche. Or chaque nœud de cet arbre de recherche est un choix d'une prémisse d'une instance de règle logique, correspondant à une occurence particulière d'une sous-formule de la formule (ou du séquent) de départ. Or l'absence de duplication garantit que la profondeur de cet espace de recherche, autrement dit la longueur de la branche la plus longue, est de taille linéaire en la taille de la formule de départ, c'est-à-dire inférieur ou égale à n. Chaque nœud nécessite un espace de taille proportionnelle à $\log n$ pour être stocké.

> On obtient une implantation $\mathcal{O}(n \log n)$ en espace de la recherche de validité en logique intuitionniste, c'est-à-dire le meilleur résultat théorique connu [Hud 93].[39]

Dans la pratique, c'est une taille linéaire que l'on obtient dans la mesure où il est difficile d'envisager des formules dont la taille dépasse 2^{32} (ou 2^{64} pour les processeurs à 64 bits.) Ceci

[39] Nous rappelons que le problème de décision dans LI est PSPACE-complet [Sta 79].

veut dire que l'on peut **éviter toute allocation dynamique** lors du parcours de l'espace de recherche de preuve, contrôlant parfaitement les ressources mémoires nécessaires à STRIP, ce qui permet aussi une accélération à l'exécution dans la mesure où la recherche de preuve n'est plus en fait qu'un (long) calcul.

L'absence de duplication permet aussi d'autres optimisations au niveau du temps de calcul. Ces optimisations qui sont naturelles en logique classique propositionnelle par exemple, car le calcul des séquents ne possède pas non plus de duplications, sont rendues possibles par le fait qu'une sous-formule donnée ne peut servir qu'au plus une fois le long d'une branche de l'espace de recherche de preuve. La numérotation préalable des sous-formules permet de les comparer à l'avance, et pas à chaque fois que cela est nécessaire. Cette comparaison a lieu par exemple lorsque l'on doit reconnaitre un axiome :

$$\frac{}{\Gamma, A_0, \ldots, A_n, A_i^\star \to C \vdash \blacksquare}$$

En effet, on doit comparer A_i à $A_0, \ldots, A_n$ pour déterminer si on a un axiome ou pas. Dans l'implantation de STRIP, l'identification d'un axiome se fait en temps constant. Cela ne serait pas possible si les formules devaient être comparées les unes aux autres à chaque fois. On pourait faire la même remarque pour la règle $[\to^1_L]$.

De manière générale, l'application de toutes les règles de STRIP se fait en temps constant.[40] Ceci veut dire que le temps de parcours d'une branche quelconque de l'espace de recherche de preuve est linéaire en n. Cependant, on notera que les choix d'orientation dans l'espace de recherche, qui sont effectués par la stratégie, ne prennent pas toujours un temps constant. Ceci constitue une voie d'améloriation de l'implantation sur laquelle nous continuons de travailler.

Les performances

Nous avons effectué des comparaisons de performance entre STRIP et d'autres systèmes de recherche de preuve en logique intuitionniste propositionnelle. Il s'agit des systèmes Porgi[41] et ft[42]. Même s'il ne s'agit pas encore de tests exhaustifs, nous avons pu constater de très gros écarts de performances à l'avantage de STRIP, au niveau de l'algorithme de recherche de validité.

Dans le cas de Porgi qui est aussi basé sur LJT, on constate qu'à stratégie égale[43] STRIP prend nettement l'avantage. D'autre part, la taille des formules que l'on peut tester avec Porgi est très vite limitée si l'on ne veut pas attendre trop longtemps, ce qui fait que nos tests se cantonnent à des formules de petite taille. À la décharge de Porgi, on notera qu'il est écrit en SML ce qui le rend bien sûr moins performant qu'un programme écrit en C tel que STRIP. Pour ne donner qu'un exemple, la formule qui exprime le principe des « trous de pigeons » pour 3 trous et 4 pigeons est prouvée en 2 milli-secondes par STRIP et en 15 secondes par Porgi.

Dans le cas de ft qui lui est écrit en C, nous avons fait des comparaisons sur des formules plus conséquentes (taille 1000 à 2000.) Par exemple, la formule

$$(\exists x(p(x) \wedge \forall y(q(y) \to r(x,y))) \wedge \neg\exists x(q(x) \wedge \forall y(p(y) \to r(x,y))) \to \exists x(p(x) \wedge \neg q(x))$$

instanciée sur un domaine de taille 6 est prouvée en 2 secondes par STRIP alors que ft met 6 minutes. Sur un domaine de taille 7, notre prouveur met 35 secondes. Nous n'avons pas attendu

[40] À l'exception notable d'une règle particulière qui doit couper un sous-arbre. Cependant, le temps nécessaire à son application est linéaire à la réduction de taille de la forêt.

[41] `http://www.cis.ksu.edu/~allen/porgi.html`

[42] `http://www.sics.se/isl/ft.html`

[43] Porgi n'en a qu'une et elle correspond à notre stratégie sophistiquée.

la fin du calcul pour ft.[44] Pour information, la taille de la preuve obtenue par STRIP dans ce dernier cas est de 4 500 000 nœuds. Il se trouve que dans ce cas précis, la stratégie de STRIP ne se « trompe » jamais et il n'y a donc pas de retour arrière. L'espace de recherche de preuve fait donc la même taille.
Pour plus d'informations sur les tests de performances que nous avons effectués, le lecteur est invité à se référer à [LW 00b].

Conclusion et perspectives

Dans ce chapitre consacré à la recherche de preuves et à la construction de contre-modèles en logique intuitionniste, nous avons mise en œuvre les techniques de construction d'arbres de réfutation et la notion de co-validité pour prouver la complétude et la propriété des modèles finis de plusieurs systèmes adaptés à la logique intuitionniste, à savoir LJT, SLJ et STRIP. De ces démonstrations, nous montrons qu'il est possible d'extraire un algorithme de construction de preuves ou de contre-modèles. Nous établissons un lien entre deux approches opposés du problème de la décision pour la logique intuitionniste : l'approche syntaxique qui consiste à fournir un système analytique et à en prouver la complétude et l'approche sémantique qui consiste à prouver la propriété des modèles finis. Dans notre démarche, la preuve de complétude de système en question contient l'algorithme de construction de contre-modèles. Cette preuve met en œuvre les notions de réfutation et de co-validité.

Nous cherchons ensuite à supprimer la duplication de sous-formules dans LJT dans le but d'obtenir une implantation plus efficace des algorithme de décision. Un première étape consiste à supprimer une forme de duplication introduite par une règle d'implication gauche, qui se matérialise par une formule commune entre la conclusion et la sous-formule gauche d'une implication dans les hypothèses. Nous montrons que ce lien peut-être conservé par la suite ce qui supprime cette première forme de duplication. L'autre forme de duplication est introduction par une disjonction à la gauche d'une implication au niveau d'une hypothèse. Dans le système STRIP nous montrons comment cette deuxième forme de duplication peut-être absorbée au niveau des branches d'un arbre. En représentant les séquents sous forme de forêts, nous les appelons alors de $\mathcal{T}$-séquents, nous obtenons un système où toutes les duplications ont été supprimées.

Enfin nous présentons les conséquences de la suppression des duplications aux niveaux des choix d'implantation et des performances du système STRIP au niveau de la gestion des ressources temporelles et spatiales.

L'une des perspectives de travail que nous envisageons serait de prouver formellement[45] la validité et les propriétés relatives à la complexité de l'implantation du système STRIP que nous avons réalisé. Les travaux de K. Weich sur l'extraction de prouveurs à partir de preuves de complétudes [Wei 98] peuvent être adaptés à la représentation particulière la logique intuitionniste que constitue le système STRIP. L'intérêt d'une telle démarche, outre la certification d'un programme, réside dans le fait que les propriétés de complexité dépendent évidemment de l'implantation précise de STRIP.

D'autres part, il paraît tout à fait possible, à la lumière des travaux [Ave 99], d'adapter les techniques de partage à d'autres logiques intermédiaires, c'est-à-dire situées entre la logique intuitionniste et la logique classique. Par exemple, l'adaptation au cas de la logique de Gödel-Dummet [Dyc 99] des techniques issues de LJT est suffisamment avancée pour pouvoir envisager un transfert de la notion de $\mathcal{T}$-séquent à cette logique.

[44] Nous l'avons interrompu après 2 heures de temps.
[45] C'est-à-dire à l'aide d'un outil pour faire des preuves de programmes comme Coq ou Hol.

Enfin, la réduction des contre-modèles générés par STRIP est un point sur lequel nous travaillons également. Une approche basée sur la correspondance entre arbre de Kripke et algèbre de Heyting est à l'étude et permet d'obtenir certaines réductions. Notons cependant que la notion de minimalité d'un contre-modèle reste encore à préciser.

4 | Logique intuitionniste linéaire

Introduction

Dans ce chapitre, nous étudions certains aspects essentiellement sémantiques de la logique intuitionniste linéaire, mais basés sur la notion de réfutation et donc de la recherche de preuve. De nombreux travaux présentent différentes approches de la recherche de preuve en logique linéaire classique mais aussi dans sa version intuitionniste.

En logique linéaire, les problèmes de décision de validité sont généralement complexes. Rappelons que la logique linéaire propositionnelle (exponentielles comprises) est indécidable [Lin 92b] et que le fragment de Horn de MLL est NP-complet [Kan 94]. Toutefois, de nombreux travaux ont été consacrés à la recherche de preuve dans divers fragments. Les méthodes et techniques usuelles de recherche de preuve des logiques classique ou intuitionniste ne peuvent pas s'adapter directement au cas linéaire. Au niveau sémantique, l'adaptation des modèles booléens ou de Kripke n'est pas immédiate non plus [All 93].

Il existe des méthodes de recherche de preuve qui se fondent sur le calcul des séquents linéaire avec par exemple la définition de formes canoniques [Hod 94, Gal 94, And 92], de stratégies particulières de recherche [Tam 94] et d'une gestion efficace des formules vues comme des ressources [Har 97, Cer 00].

D'autres méthodes fondées sur la construction de réseaux de preuves [Gir 95b] ont été proposées dans différents fragments de la logique linéaire [Gal 00, Gal 98b]. D'autres travaux ont conduits à adapter des méthodes classiques de la recherche de preuve (tableaux, connections, résolution,...) [Har 92, Man 99, Kre 97, Min 93].

La recherche de preuves dans LLI peut être abordées à partir de la logique linéaire classique. Cependant la relation entre LLI et LL n'est pas la même que dans le cas non linéaire (entre LI et LC.) Ainsi, il existe des versions multi-conclusions LI [Sch 91] alors qu'il est nécessaire d'étendre LLI avec le $\parr$ pour en obtenir une version multi-conclusions [Hyl 93, Br 97].

Au niveau sémantique, de nombreux travaux existent également. Pour l'interprétation des formules, nous avons la sémantique algébrique et la sémantique des phases [Gir 95a, Tro 92, Yet 90, Ros 90, Ono 93] ou encore la sémantique à base de réseaux de Petri [Bro 89, Mes 91, Eng 90, Eng 97].

Pour la sémantique des preuves que nous n'aborderons pas ici, on pourra citer les travaux sur la sémantique catégorique [Bar 91], la sémantique des jeux [Laf 91, Abr 94, Bla 92], les espaces de Chu [Pra 95, Pra 97, Dev 99, Abr 99] ou sur les espaces cohérents [Gir 87, Gir 89].

Dans le chapitre précédent, notre étude de la recherche de preuve dans des calculs de séquents pour LI nous a mené jusqu'à l'implantation concrète d'un logiciel. Dans cette partie, nous étudions essentiellement la sémantique des formules de la logique linéaire intuitionniste, sous différentes formes, dans le but d'aborder la construction de contre-modèles pour les formules invalides et les formes que peuvent prendre ces contre-modèles.

Partant du calcul des séquents sans coupures usuel pour LLI — voir figure 4.1 — nous établirons l'analycité de ce système ce qui nous permettra de construire des contre-modèles à partir de réfutations. Nous rappelons la sémantique algébrique (quantales) et la sémantique des phases pour LLI ainsi que leur propriété de complétude.

Nous introduisons une nouvelle sémantique fondée sur la notion de ressource. Cette interprétation s'avère très adaptée à la preuve des propriétés sémantiques fondamentales de complétude et des modèles finis. Les deux démonstrations ne diffèrent guère que d'un passage au quotient. Ce quotient est construit à partir d'un arbre de réfutation. Nous montrons également que la même démonstration permet d'obtenir une preuve sémantique de l'élimination des coupures. Comme il est possible de transformer un espace de ressources (fini) en espace de phases (fini) tout en préservant l'interprétation des formules, et qu'une transformation de ce type est aussi possible des espaces de phases vers les quantales, nous en déduisons la propriété des modèles finis pour la sémantique des phases et la sémantique algébrique. De manière étonnante, cette dernière propriété n'a été établie que très récemment pour LLI [Oka] alors que dans le cas classique, elle est connue depuis un certain temps déjà [Laf 97].

Enfin, nous introduisons une autre sémantique à base de monoïdes ordonnés qui est une version abstraite de la sémantique à base de réseaux de Petri. Notre cadre de travail nous permet d'expliquer pourquoi seuls des résultats de complétude partielle avaient pu être établis dans [Eng 97]. Nous démontrons enfin la propriété des modèles finis pour ces deux dernières interprétations.

4.1 Syntaxe

4.1.1 Formules de LLI

L'ensemble Form des formules de **la logique intuitionniste linéaire propositionnelle, commutative et sans exponentielles** notée LLI construites à partir d'un ensemble de variables Var est défini par induction par :

$$\mathsf{Form}: \quad A ::= V \mid 1 \mid \mathsf{F} \mid \mathsf{T} \mid A \,\square\, A \text{ pour } \square \in \{\otimes, \&, \oplus, \multimap\} \text{ et } V \in \mathsf{Var}$$

Ainsi, l'ensemble des variables Var que nous noterons usuellement par les lettres V ou W est contenu dans l'ensemble des formules, i.e. $\mathsf{Var} \subseteq \mathsf{Form}$, formules que nous dénoterons généralement par les lettres A, B ou C. On appelle **additifs** les opérateurs $\oplus$ et $\&$ et **multiplicatifs** les opérateurs $\otimes$ et $\multimap$. Les symboles 1, T et F représentent des constantes. Par exemple, l'écriture $A \multimap (B \otimes C)$ est une formule si A, B et C sont des formules.

Si on compare LI et LLI on s'apperçoit que la conjonction $\wedge$ est dédoublée en une version additive $\&$ et une version multiplicative $\otimes$ qui capturent les aspects linéaires de la logique. Dans sa version intuitionniste, la logique linéaire n'a pas de disjonction multiplicative $\parr$ mais seulement une additive $\oplus$, contrairement à la logique linéaire classique [Gir 87, Gir 95a]. Cependant, des versions multi-conclusions intuitionnistes de la logique linéaire ont aussi été étudiées. Dans ce cas, une disjonction multiplicative $\parr$ peut-être introduite comme par exemple dans le système FILL [Hyl 93, Bie 96].

On notera $\mathsf{Context}$ l'ensemble des contextes. Un contexte généralement dénoté par les lettres Γ ou Δ est un multi-ensemble fini de formules, c'est-à-dire une liste dans lequel l'ordre ne compte pas, mais la multiplicité des occurences compte. On écrira souvent $A_1, \ldots, A_n$ le multi-ensemble de n formules. Ainsi les deux multi-ensembles A, A, B et A, B, A sont égaux mais diffèrent tous les deux de A, B car le nombre d'occurences de A est différent.

$$\frac{}{A \vdash A}\ [\mathrm{id}] \qquad \frac{\Gamma \vdash A \qquad A, \Delta \vdash B}{\Gamma, \Delta \vdash B}\ [\mathrm{cut}]$$

$$\frac{\Gamma, A, B \vdash C}{\Gamma, A \otimes B \vdash C}\ [\otimes_L] \qquad \frac{\Gamma \vdash A \qquad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}\ [\otimes_R]$$

$$\frac{\Gamma \vdash A \qquad \Delta, B \vdash C}{\Gamma, \Delta, A \multimap B \vdash C}\ [\multimap_L] \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\ [\multimap_R]$$

$$\frac{\Gamma, A \vdash C}{\Gamma, A \,\&\, B \vdash C}\ [\&^1_L] \qquad \frac{\Gamma, B \vdash C}{\Gamma, A \,\&\, B \vdash C}\ [\&^2_L] \qquad \frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \,\&\, B}\ [\&_R]$$

$$\frac{\Gamma, A \vdash C \qquad \Gamma, B \vdash C}{\Gamma, A \oplus B \vdash C}\ [\oplus_L] \qquad \frac{\Gamma \vdash A}{\Gamma \vdash A \oplus B}\ [\oplus^1_R] \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \oplus B}\ [\oplus^2_R]$$

$$\frac{\Gamma \vdash A}{\Gamma, 1 \vdash A}\ [1_L] \qquad \frac{}{\vdash 1}\ [1_R] \qquad \frac{}{\Gamma, \mathsf{F} \vdash A}\ [\mathsf{F}_L] \qquad \frac{}{\Gamma \vdash \mathsf{T}}\ [\mathsf{T}_R]$$

FIG. 4.1 – Le calcul Gil des séquents pour LLI

4.1.2 Calcul des séquents

Un séquent intuitionniste est une paire $(\Gamma, A) \in \mathsf{Context} \times \mathsf{Form}$ où Γ est un contexte et A une formule. On écrira plutôt $\Gamma \vdash A$ pour dénoter un séquent. Si Γ est le contexte vide $\emptyset$ alors on écrira même $\vdash A$ plutôt que $\emptyset \vdash A$.

Le calcul des séquents pour LLI est un système de règles décrit en figure 4.1. Ce calcul est noté Gil. Il permet de définir une notion de validité pour les séquents en posant

$$\Vdash \triangleq \vdash_{\mathsf{Gil}} \tag{4.1}$$

Lorsqu'un séquent $\Gamma \vdash A$ est prouvable, i.e. admet une preuve, on écrira $\Gamma \vdash_{\mathsf{Gil}} A$ (ou $\Gamma \Vdash A$) plutôt que $\vdash_{\mathsf{Gil}} \Gamma \vdash A$ (ou $\Vdash \Gamma \vdash A$.) Le calcul des séquents possède une règle particulière [cut] appelée **règle de coupure**. Nous montrerons en section 4.4.7 que cette règle est en fait redondante.

Définition 4.1.1 ($\mathsf{Gil}_{\mathsf{sc}}$) *Le système* $\mathsf{Gil}_{\mathsf{sc}}$ *est le système* Gil *privé de la règle de coupure* [*cut*].

Pour l'instant, nous ne supposons pas l'élimination des coupures, autrement dit, nous distinguons la relation de déduction $\vdash_{\mathsf{Gil}}$ dans Gil de la relation de déduction notée $\vdash_{\mathsf{sc}}$ dans le système $\mathsf{Gil}_{\mathsf{sc}}$.

4.1.3 Analycité du système $\mathsf{Gil}_{\mathsf{sc}}$

Dans cette section nous expliquons pourquoi le système $\mathsf{Gil}_{\mathsf{sc}}$ est analytique. On constate que les prémisses de chaque règle sont déterminées par le choix de la formule principale — i.e. la formule qui est décomposée, — et éventuellement d'un découpage en deux du contexte gauche, dans le cas des règles $[\otimes_R]$ et $[\multimap_L]$. Or ce choix d'une formule principale s'effectue soit à gauche du symbole $\vdash$ où se trouve un nombre fini d'occurences de formules, soit à droite où il y a exactement une formule. Enfin, il y a un nombre fini de façons de découper un contexte Γ, Δ en deux parties Γ et Δ.

Proposition 4.1.1 *Étant donné un séquent* $\Gamma \vdash A$*, il y a un nombre fini d'instances de règles de* $\mathsf{Gil}_{\mathsf{sc}}$ *qui ont ce séquent pour conclusion.*

Donc les arbres de réfutation sont de largeur finie. Qu'en est-il de leur hauteur ? Pour la logique linéaire, il est très facile de montrer qu'elle est finie elle aussi. Il suffit de voir que le nombre de symboles diminue quand on passe d'une conclusion à une prémisse, dans la mesure où la virgule ne compte pas comme un symbole. Seule la règle [cut] ne suit pas cette loi mais nous l'avons justement supprimée.

Proposition 4.1.2 *Le système* $\mathsf{Gil}_{\mathsf{sc}}$ *est analytique.*

La conséquence immédiate, voir section 1.5, est que les arbres de réfutation de $\mathsf{Gil}_{\mathsf{sc}}$ sont finis et que l'algorithme de recherche de preuves se termine, retournant soit une preuve, soit une réfutation. Nous sommes donc capables de fabriquer une réfutation pour n'importe quelle formule ou séquent invalide.

4.2 Sémantique algébrique

4.2.1 Interprétation

L'interprétation des formules et des séquents de LLI consiste à associer des opérateurs algébriques aux opérateurs logiques et à vérifier que cette association transmet la validité (sémantique) des prémisses vers la conclusion des règles du calcul de séquents Gil. La structure algébrique correspondant à LLI peut être retrouvée en considérant l'algèbre des termes (ou de Lindenbaum) décrite en section 4.2.2 : il s'agit de la notion de quantale. Les résultats de complétude sont bien connus depuis longtemps, voir [Yet 90, Tro 92] par exemple. Nous en faisons une présentation rapide. La définition de la notion de quantale a déjà été donnée en section 2.5 mais il nous semble utile de la rappeler :

Définition 4.2.1 (Quantale) *Une quantale est un triplet* $(\mathcal{Q}, \bullet, \leqslant)$ *où* $(\mathcal{Q}, \bullet)$ *est un monoïde et* $(\mathcal{Q}, \leqslant)$ *est un treillis complet. De plus, pour tous éléments* a, $(b_i)_i$ *de* $\mathcal{Q}$, *on a* $a \bullet \bigvee_i b_i = \bigvee (a \bullet b_i)$, *relation que l'on appelle* ***distributivité infinie****.*

En particulier, une quantale est un monoïde résidué — voir définition 2.3.5 page 28 — et son $\vee$-complété lui est isomorphe d'après 2.5.8. Nous rappelons que le résidu est défini par $a \multimap b \triangleq \max\{z \mid z \bullet a \leqslant b\}$ (d'après 2.5.2). Pour interpréter les formules, nous nous donnons une valuation $\sigma : \mathsf{Var} \to \mathcal{Q}$, c'est à dire une interprétation des formules atomiques, puis nous interprétons les opérateurs multiplicatifs $1, \otimes, \multimap$ par les opérations monoïdales respectives $\mathsf{e}, \bullet, \multimap$ et les additifs $\oplus, \&, \mathsf{F}, \mathsf{T}$ par les opérations du treillis respectives $\vee, \wedge, \bot, \top$, ce qui nous donne la définition inductive suivante :

Sémantique algébrique

$$
\begin{array}{ll}
[\![V]\!] \triangleq \sigma(V) & [\![A \multimap B]\!] \triangleq [\![A]\!] \multimap [\![B]\!] \\
[\![1]\!] \triangleq \mathsf{e} & [\![A \otimes B]\!] \triangleq [\![A]\!] \bullet [\![B]\!] \\
[\![\mathsf{F}]\!] \triangleq \bot & [\![A \oplus B]\!] \triangleq [\![A]\!] \vee [\![B]\!] \\
[\![\mathsf{T}]\!] \triangleq \top & [\![A \,\&\, B]\!] \triangleq [\![A]\!] \wedge [\![B]\!]
\end{array}
$$

Le lecteur aura remarqué que cette définition est paramétrée par $\mathcal{Q}$ et σ mais nous n'écrirons $[\![\cdot]\!]^{\sigma}_{\mathcal{Q}}$ qu'en cas d'ambiguïté potentielle, ceci afin de ne pas alourdir les notations. Les contextes sont ensuite interprétés suivant la méthode présentée en section 2.6, c'est-à-dire $[\![A_1, \ldots, A_n]\!] \triangleq [\![A_1]\!] \bullet \cdots \bullet [\![A_n]\!]$. Montrer que cette interprétation est valide consiste à prouver que la relation

binaire $\succ \subseteq \mathsf{Context} \times \mathsf{Form}$ définie par $\Gamma \succ A \Leftrightarrow [\![\Gamma]\!] \leqslant [\![A]\!]$ est close pour les règles du calcul des séquents en figure 4.1.

Proposition 4.2.1 (Adéquation) *Si* $\Gamma \vdash_{\mathsf{Gil}} A$ *alors* $[\![\Gamma]\!] \leqslant [\![A]\!]$.

Démonstration. D'après le résultat 1.2.2, il suffit de montrer que les règles du calcul des séquents Gil sont valides par rapport à la relation $\succ$, car on obtient alors $\vdash_{\mathsf{Gil}} \subseteq \succ$. Ceci revient exactement à montrer que $\succ$ est close pour les règles de Gil. Considérons par exemple la règle

$$\frac{\Gamma \vdash A \qquad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B} \ [\otimes_R]$$

On suppose que $\Gamma \succ A$ et $\Delta \succ B$, c'est à dire $[\![\Gamma]\!] \leqslant [\![A]\!]$ et $[\![\Delta]\!] \leqslant [\![B]\!]$. Montrons que $\Gamma, \Delta \succ A \otimes B$. Or $[\![\Gamma, \Delta]\!] = [\![\Gamma]\!] \bullet [\![\Delta]\!]$ par définition de $[\![\cdot]\!]$ sur les contextes. D'autre part, $[\![A \otimes B]\!] = [\![A]\!] \bullet [\![B]\!]$ par définition aussi. On obtient donc $[\![\Gamma, \Delta]\!] = [\![\Gamma]\!] \bullet [\![\Delta]\!] \leqslant [\![A]\!] \bullet [\![B]\!] = [\![A \otimes B]\!]$ parce que l'opération monoïdale $\bullet$ est croissante. La relation $\succ$ est donc close pour la règle $[\otimes_R]$. Pour les autres règles, nous donnons les principaux arguments, sans les détails.

- les cas des axiomes [id], $[1_R]$ et $[\mathsf{T}_R]$ sont triviaux, tout comme pour les règles $[1_L]$ et $[\otimes_R]$;
- la règle de coupure s'obtient par transitivité de $\leqslant$;
- pour l'implication linéaire $[\multimap_L]$, on obtient $[\![\Gamma, \Delta, A \multimap B]\!] = [\![\Delta]\!] \bullet [\![\Gamma]\!] \bullet ([\![A]\!] \multimap\!\!\bullet\ [\![B]\!]) \leqslant [\![\Delta]\!] \bullet [\![A]\!] \bullet ([\![A]\!] \multimap\!\!\bullet\ [\![B]\!]) \leqslant [\![\Delta]\!] \bullet [\![B]\!] = [\![\Delta, B]\!] \leqslant [\![C]\!]$;
- pour $[\multimap_R]$, il s'agit simplement de la propriété caractéristique du résidu $\multimap\!\!\bullet$, voir définition 2.3.5 ;
- pour $[\&_L^1]$ par exemple, on obtient le calcul $[\![\Gamma, A \,\&\, B]\!] = [\![\Gamma]\!] \bullet ([\![A]\!] \wedge [\![B]\!]) \leqslant [\![\Gamma]\!] \bullet [\![A]\!] = [\![\Gamma, A]\!] \leqslant [\![C]\!]$;
- pour $[\&_R]$, on obtient $[\![\Gamma]\!] \leqslant A$ et $[\![\Gamma]\!] \leqslant B$ donc $[\![\Gamma]\!] \leqslant [\![A]\!] \wedge [\![B]\!] = [\![A \,\&\, B]\!]$;
- pour $[\oplus_L]$, on obtient $[\![\Gamma, A \oplus B]\!] = [\![\Gamma]\!] \bullet ([\![A]\!] \vee [\![B]\!]) = ([\![\Gamma]\!] \bullet [\![A]\!]) \vee ([\![\Gamma]\!] \bullet [\![B]\!]) \leqslant [\![C]\!] \vee [\![C]\!] = [\![C]\!]$;
- pour $[\oplus_R]$, il suffit de voir que $[\![A]\!] \leqslant [\![A]\!] \vee [\![B]\!] = [\![A \oplus B]\!]$;
- enfin pour $[\mathsf{F}_L]$, il suffit que voir que $[\![\Gamma, \mathsf{F}]\!] = [\![\Gamma]\!] \bullet [\![\mathsf{F}]\!] = [\![\Gamma]\!] \bullet \bot = \bot$, voir section 2.5 chapitre 2.

□

Cette preuve peut paraître longue et fastidieuse (calculatoire) mais elle est en fait très naturelle pour peu que l'on maîtrise les règles du calcul dans une quantale. Le lecteur pourra noter que les additifs $\oplus, \&, \mathsf{F}, \mathsf{T}$ se traitent de la même manière que dans LI (voir section 3.2.1 chapitre 3.)

Nous venons donc de définir une interprétation sémantique valide de LLI. Toute formule A (ou séquent $\Gamma \vdash A$) prouvable dans le système Gil est (sémantiquement) valide dans notre interprétation, i.e. $\mathsf{e} \leqslant [\![A]\!]$ (ou $[\![\Gamma]\!] \leqslant [\![A]\!]$). Nous montrons maintenant la complétude de cette sémantique.

4.2.2 Algèbre de Lindenbaum et complétude

Le concept fondamental de la sémantique algébrique est la notion d'**algèbre de termes**, ou **algèbre de Lindenbaum** [Ras 63]. L'idée consiste à se servir des formules (ou des contextes) elles-mêmes comme éléments sémantiques (ceux d'une quantale dans le cas de LLI) et leur donner une structure algébrique. Ensuite on diagonalise, autrement dit, on interprète les formules par elles-mêmes.

Toutes les constructions qui fondent cette démarche ont été présentées au chapitre 2. On va s'en servir largement ici. On utilise la notation $\Vdash$ pour $\vdash_{\mathsf{Gil}}$ puisque par définition, une formule est valide si elle est prouvable dans le système Gil. On se permet un petit abus de notation : $\Vdash$ est une relation sur $\mathsf{Context} \times \mathsf{Form}$, mais en identifiant une formule A et le contexte (multi-ensemble) $\{A\}$, on peut aussi la voir comme une relation sur $\mathsf{Form} \times \mathsf{Form}$. On considère le triplet $(\mathsf{Form}, \otimes, \Vdash)$ où $\otimes$ est identifié[46] à l'opération binaire $(A, B) \mapsto A \otimes B$ définie sur l'ensemble Form :

Proposition 4.2.2 *Le triplet* $(\mathsf{Form}, \otimes, \Vdash)$ *est un monoïde préordonné ayant* 1 *comme élément neutre, résidué par* $\multimap$*, dans lequel* $A \oplus B$ *est une borne supérieure (resp. inférieure) de* A *et* B *(resp.* $A \,\&\, B$*.)* F *est un plus petit élément et* T *un plus grand élément.*

Démonstration. On voit que $\Vdash$ est un préordre du fait des règles [id] et [cut], cette dernière étant **essentielle** ici.[47] Nous rappelons que $A \simeq B$ veut dire que l'on a à la fois $A \Vdash B$ et $B \Vdash A$. Il suffit ensuite de montrer les relations suivantes,

Associativité	$(A \otimes B) \otimes C \simeq A \otimes (B \otimes C)$
Commutativité	$A \otimes B \simeq B \otimes A$
Élément neutre	$1 \otimes A \simeq A$
Résidu	$A \otimes B \Vdash C \Leftrightarrow A \Vdash B \multimap C$
Borne sup.	$A \oplus B \simeq A \vee B$
Borne inf.	$A \,\&\, B \simeq A \wedge B$
Plus petit élément	$\mathsf{F} \Vdash A$
Plus grand élément	$A \Vdash \mathsf{T}$

La preuve de ces propriétés est simple mais un peu fastidieuse. Elle nécessite l'utilisation de la règle de coupure [cut]. Une remarque cependant : quand on écrit $A \oplus B \simeq A \vee B$, il s'agit d'une abréviation pour l'expression $\forall C\ A \oplus B \Vdash C \Leftrightarrow (A \Vdash C$ et $B \Vdash C)$ et de même pour la borne inférieure (voir section 2.2.1.) □

On peut donc choisir la plus grande prétopologie $(\cdot)^\circ$ sur le monoïde préordonné Form, c.f. résultat 2.5.4, et obtenir, d'après le résultat 2.5.6, une quantale $(\mathsf{Form}^\circ, (\cdot \otimes \cdot)^\circ, \subseteq)$. D'autre part $A \mapsto \{A\}^\circ = {\downarrow}\{A\} = \{F \mid F \Vdash A\}$ est une $\vee$-complétion. Dans cette quantale, l'élément neutre est $\mathsf{e} = \{1\}^\circ = \{F \mid F \Vdash 1\}$. De plus, d'après 2.5.7, toutes les opérations précitées sont préservées par ce plongement. On définit l'interprétation suivante dans cette quantale, $\sigma(V) \triangleq \{V\}^\circ = \{F \mid F \Vdash V\}$. Alors on obtient le résultat suivant :

Lemme 4.2.3 *Pour toute formule* A*, on a* $[\![A]\!] = \{A\}^\circ = \{F \in \mathsf{Form} \mid F \Vdash A\}$*.*

Démonstration. Par induction triviale puisque $A \mapsto \{A\}^\circ$ préserve toutes les opérations utilisées pour définir $[\![\cdot]\!]$. □

Proposition 4.2.4 (Complétude) *La sémantique algébrique est complète.*

Démonstration. On considère la quantale précédente $(\mathsf{Form}, \otimes, \Vdash)^\circ$ munie de l'interprétation $\sigma(V) \triangleq \{V\}^\circ$. Soit A une formule invalide $\nVdash A$. Alors on a aussi $1 \nVdash A$ (d'après la règle $[1_L]$) et donc $1 \notin \{F \mid F \Vdash A\} = [\![A]\!]$. Or $1 \in \{F \mid F \Vdash 1\} = \mathsf{e}$ d'après la règle [id]. Ainsi nous avons un contre-modèle de A car $\mathsf{e} \not\subseteq [\![A]\!]$. □

[46] A priori, $\otimes$ est juste un symbole. On lui associe naturellement une opération binaire.

[47] Nous verrons en section 4.4 que l'on peut s'affranchir de [cut] dans une sémantique plus générale, ce qui nous donne aussi une preuve sémantique de l'élimination des coupures.

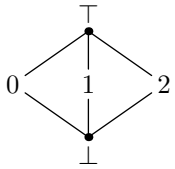


FIG. 4.2 – La structure de treillis d'une quantale.

4.2.3 Contre-exemples

On utilise maintenant le treillis décrit en figure 4.2. Sur ce treillis noté $\mathcal{K}$, nous pouvons considérer la structure de quantale obtenue par complétion du monoïde plat $\mathbb{Z}/3\mathbb{Z}$, voir section 2.5.3 chapitre 2. Si nous interprétons les formules de la manière suivante, $[\![A]\!] \triangleq 0$, $[\![B]\!] \triangleq 1$ et $[\![C]\!] \triangleq 2$, alors nous obtenons $[\![A \oplus (B \,\&\, C)]\!] = [\![A]\!] \vee ([\![B]\!] \wedge [\![C]\!]) = 0 \vee (1 \wedge 2) = 0 \vee \bot = 0$. Par contre, on obtient $[\![(A \oplus B) \,\&\, (A \oplus C)]\!] = \top \wedge \top = \top$. D'où

$$[\![(A \oplus B) \,\&\, (A \oplus C)]\!] \not\leqslant [\![A \oplus (B \,\&\, C)]\!]$$

et nous avons donc un contre-exemple à la distributivité de $\oplus$ par rapport à &.

Sur le même treillis $\mathcal{K}$, nous pouvons considérer la structure de quantale $(\mathcal{K}, \vee, \leqslant)$ (voir résultat 2.5.1 en page 31) autrement dit, l'opération monoïdale est la borne supérieure qui distribue bien évidement sur elle-même. Avec la même interprétations des variables A, B et C, on obtient $[\![(A \otimes B) \,\&\, (A \otimes C)]\!] = \mathsf{T}$ alors que $[\![A \otimes (B \,\&\, C)]\!] = 0$ donc

$$[\![(A \otimes B) \,\&\, (A \otimes C)]\!] \not\leqslant [\![A \otimes (B \,\&\, C)]\!]$$

Nous avons là un contre-exemple à la distributivité de $\otimes$ par rapport à &.

4.3 Sémantique des phases

On pourrait considérer que la sémantique des phases est la sémantique initiale de la logique linéaire [Gir 87], et qu'elle précède la sémantique algébrique. Cependant la sémantique algébrique est une notion générique qui est valable pour presque toutes les logiques, à partir du moment où on a défini un système de règles, par exemple le calcul des séquents linéaire pour LLI. À la lumière des résultats de complétions et des rapports entre quantales et espaces de phases (voir section 2.5,) on peut considérer que ces deux sémantiques ne représentent que deux manières différentes d'aborder la même notion.

4.3.1 Interprétation, complétude

Si $(\mathcal{M}, \bullet, (\cdot)^\star)$ est un espace de phases, i.e. $(\cdot)^\star$ est une clôture stable, alors $(\mathcal{M}^\star, (\cdot \bullet \cdot)^\star, \subseteq)$ est une quantale et on peut donc utiliser l'interprétation définie dans la section précédente en y instanciant les opérations algébriques qui ont été calculées en proposition 2.5.3 page 31. Soit $\sigma : \mathsf{Var} \to \mathcal{M}$ une interprétation des variables, on obtient l'interprétation suivante basée sur la valuation $V \mapsto \sigma(V)^\star \in \mathcal{M}^\star$:

Sémantique des phases

$$\begin{array}{ll} [\![V]\!] \triangleq \sigma(V)^\star & [\![A \multimap B]\!] \triangleq [\![A]\!] \multimap\!\bullet\, [\![B]\!] \\ [\![1]\!] \triangleq \mathsf{e}^\star & [\![A \otimes B]\!] \triangleq ([\![A]\!] \bullet [\![B]\!])^\star \\ [\![\mathsf{F}]\!] \triangleq \emptyset^\star & [\![A \oplus B]\!] \triangleq ([\![A]\!] \cup [\![B]\!])^\star \\ [\![\mathsf{T}]\!] \triangleq \mathcal{M} & [\![A \,\&\, B]\!] \triangleq [\![A]\!] \cap [\![B]\!] \end{array}$$

Cette sémantique est bien sûr adéquate, étant un cas particulier de la sémantique algébrique, c.f. proposition 4.2.1. Ainsi si $\Gamma \Vdash A$ alors il vient $[\![\Gamma]\!] \subseteq [\![A]\!]$.

Proposition 4.3.1 (Complétude) *Si A est invalide $\nVdash A$ alors il existe un espace de phase $(\mathcal{M}, \bullet, (\cdot)^\star)$ et une valuation $\sigma : \mathsf{Var} \to \mathcal{M}$ tels que $\mathsf{e} \notin [\![A]\!]$, et a fortiori $\mathsf{e}^\star \not\subseteq [\![A]\!]$.*

Démonstration. On peut faire une preuve directe suivant le même schéma que la preuve de complétude de la sémantique algébrique, en considérant le monoïde quotient $\mathcal{M}/\simeq$ où $\simeq$ est l'équivalence du préordre $\Vdash$ et en utilisant la clôture stable $(\cdot)^\circ$. Une autre preuve consiste à utiliser la complétude algébrique. Si on considère une quantale $(\mathcal{Q}, \bullet, \leqslant)$ qui est un contre-modèle de A alors $(\mathcal{Q}, \bullet, (\cdot)^\circ)$ est un espace de phases et sa quantale $(\mathcal{Q}^\circ, (\cdot \bullet \cdot)^\circ, \subseteq)$ est isomorphe à $\mathcal{Q}$ par $x \mapsto {\downarrow}x$ d'après le résultat 2.5.8. Il suffit de « transférer » l'interprétation par cet isomorphisme. □

4.3.2 Espace de phases quotient

On donne ici une condition suffisante pour pouvoir « passer au quotient » tout en préservant les propriétés sémantiques. Nous rappelons la définition des espaces de phases quotient de la section 2.4.2.

Définition 4.3.1 (Espace de phases quotient) *Soit $(\mathcal{M}, \bullet, (\cdot)^\star)$ un espace de phase. Le quadruplet $(\mathcal{Q}, \bullet, (\cdot)^\diamond, \lfloor\cdot\rfloor)$ est un **quotient** de $\mathcal{M}$ si $(\mathcal{Q}, \bullet, (\cdot)^\diamond)$ est un espace de phase et $\lfloor\cdot\rfloor : \mathcal{M} \twoheadrightarrow \mathcal{Q}$ est un morphisme de monoïdes surjectif — encore appelé projection. On suppose de plus que l'on a l'équivalence $\lfloor x \rfloor \in \lfloor X \rfloor^\diamond \Leftrightarrow x \in X^\star$ pour tous $x \in \mathcal{M}$ et $X \subseteq \mathcal{M}$.*

On rappelle que l'on a alors les équations $\lfloor x \bullet y \rfloor = \lfloor x \rfloor \bullet \lfloor y \rfloor$ et $\lfloor \mathsf{e}_\mathcal{M} \rfloor = \mathsf{e}_\mathcal{Q}$. Par ailleurs, tout élément q de $\mathcal{Q}$ est de la forme $q = \lfloor m \rfloor$ pour au moins un élément $m \in \mathcal{M}$. On rappelle aussi l'identité $\lfloor X \rfloor^\diamond = \lfloor X^\star \rfloor$. La projection $\lfloor\cdot\rfloor$ s'étend extensionnellement en une fonction $\tau : \mathcal{M}^\star \to \mathcal{Q}^\diamond$ définie par $\tau(X) \triangleq \lfloor X \rfloor$ dont on rappelle qu'elle est un morphisme de la quantale $(\mathcal{M}^\star, (\cdot \bullet \cdot)^\star, \subseteq)$ vers la quantale $(\mathcal{Q}^\diamond, (\cdot \bullet \cdot)^\diamond, \subseteq)$ qui préserve la structure monoïdale (et le résidu $\multimap\!\bullet$) ainsi que la structure de treillis, d'après le lemme 2.4.4. On peut projeter une interprétation $\sigma_\mathcal{M} : \mathsf{Var} \to \mathcal{M}$ sur $\mathcal{Q}$ en posant $\sigma_\mathcal{Q}(V) = \tau(\sigma_\mathcal{M}(V))$. On note par $[\![\cdot]\!]_\mathcal{M}$ l'interprétation définie dans $\mathcal{M}$ par $\sigma_\mathcal{M}$ et par $[\![\cdot]\!]_\mathcal{M}$ l'interprétation définie dans $\mathcal{Q}$ par $\sigma_\mathcal{Q}$.

Proposition 4.3.2 *Soit $\lfloor\cdot\rfloor : \mathcal{M} \twoheadrightarrow \mathcal{Q}$ un quotient et $\sigma_\mathcal{M}$ une valuation dans $\mathcal{M}$. Alors pour toute formule A, on a l'égalité $[\![A]\!]_\mathcal{Q} = \tau\big([\![A]\!]_\mathcal{M}\big)$.*

Démonstration. Pour une variable V, on a

$$[\![V]\!]_\mathcal{Q} = \{\sigma_\mathcal{Q}(V)\}^\diamond = \{\tau(\sigma_\mathcal{M}(V))\}^\diamond = \{\lfloor \sigma_\mathcal{M}(V) \rfloor\}^\diamond = \lfloor \{\sigma_\mathcal{M}(V)\}^\star \rfloor = \tau([\![V]\!]_\mathcal{M})$$

Le reste de la preuve se fait par une induction immédiate dans la mesure où τ est un morphisme qui préserve toutes les structures (monoïdale, résidu, treillis complet) qui interviennent dans la définition de $[\![\cdot]\!]$. □

Ainsi, si $\mathcal{M}$ est un contre-modèle de A alors on a $\mathsf{e}_{\mathcal{M}} \notin [\![A]\!]_{\mathcal{M}}$ et par application du résultat précédent, $\mathsf{e}_{\mathcal{Q}} \notin [\![A]\!]_{\mathcal{Q}}$. Il est important de noter que l'on ne perd pas les contre-modèles en passant au quotient. Ceci va nous permettre de transformer un contre-modèle en contre-modèle fini en section 4.4.9.

4.4 Sémantique des ressources

En logique intuitionniste — voir chapitre 3 — nous nous sommes intéressés au problème de la gestion des ressources dans la recherche de preuves. Les règles structurelles de contraction et d'affaiblissement

$$\frac{\Gamma \vdash B}{\Gamma, A \vdash B}\ [\text{affaiblissement}] \qquad \frac{\Gamma, A, A \vdash B}{\Gamma, A \vdash B}\ [\text{contraction}]$$

par exemple, autorisent la duplication ou l'oubli des hypothèses lors de la recherche de preuves. Nous avons donc du rechercher une représentation particulière de LI sous la forme du système STRIP pour nous débarrasser du problème de la duplication : la gestion des ressources reste extérieure à la logique.

Les logiques sous-structurelles et LLI en particulier, essayent d'intégrer la gestion des ressources au sein même de la logique, en identifiant les formules à des ressources. Dans LLI par exemple, la duplication ou l'oubli d'hypothèses est invalide. Il paraît donc naturel d'essayer de fonder une interprétation de LLI sur la notion de ressource.

4.4.1 La notion de ressource

Nous retenons deux aspects importants de la notion de ressource. Tout d'abord l'axiome [id] exprime l'idée que l'hypothèse X vue comme une ressource doit absolument être consommée pour produire la conclusion X. Les ressources peuvent donc être **comptées** pour **produire** quelque chose. Dans cette interprétation, une ressource peut être identifiée avec l'ensemble des objets qu'elle peut produire. Cette approche qui est reprise dans la sémantique à base de réseaux de Petri de Winskel et Engberg — voir section 4.6 — ne conduit malheureusement pas à une sémantique complète de LLI.

$$\frac{}{X \vdash X}\ [\text{id}]$$

Un autre aspect important est reflété par la règle $[\otimes_R]$ qui exprime le fait que si l'on peut produire A à partir des ressources Γ et B à partir des ressources Δ alors on peut produire la combinaison des deux $(A \otimes B)$ à partir de la « somme » Γ, Δ des ressources. Cette règle met en avant la notion de **partage**. L'opération de composition des ressources ne sert pas seulement à les compter mais aussi à partager les ressources entre différents contextes.

$$\frac{\Gamma \vdash A \qquad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}\ [\otimes_R]$$

Dans le cadre de la logique BI qui mélange la logique intuitionniste et le fragment multiplicatif de la logique intuitionniste linéaire, une sémantique de Kripke avec partage des ressources a été proposée [O'H 99]. Dans notre étude, nous proposons également une sémantique qui prend en compte à la fois la notion production (à partir de ressources) et celle de partage et de composition des ressources. Cependant, notre approche est fondée sur la construction d'un modèle algébrique de LLI. En choisissant la plus grande clôture stable — voir section 4.4.3 — nous proposons une interprétation qui conduit à des démonstrations très « naturelles » des résultats fondamentaux de complétude, d'élimination des coupures et de propriété des modèles finis.

4.4.2 Définitions

Nous introduisons un ensemble de ressources $\mathcal{M}$ et un ensemble de produits $\mathcal{P}$. Les ressources peuvent être composées (ou superposées) par une opération binaire notée $\bullet$ et il existe une ressource vide notée $\mathtt{e}$. Les ressources et les produits peuvent être comparés par une relation $\succ$. Par exemple, on peut voir les éléments de $\mathcal{M}$ comme des sommes d'argent que l'on compose en les additionnant. Les produits eux ne peuvent pas être composés. Par exemple, on peut considérer que ce sont des fruits ou des friandises. Et la relation $m \succ p$ exprime le fait qu'à partir d'une ressource $m \in \mathcal{M}$ il est possible de produire $p \in \mathcal{P}$. Par exemple, avec 10 frs il est possible d'acheter un melon.[48] Mais avec les mêmes 10 frs il est aussi possible d'acheter 100 g de bonbons à la fraise. La composition d'un melon et de 100 g de bonbons n'est pas définie. Les produits permettent donc juste de comparer les ressources par l'intermédiaire de la relation $\succ$.

Définition 4.4.1 (Espace de ressources) *Le quadruplet $(\mathcal{M}, \mathcal{P}, \bullet, \succ)$ est un **espace de ressources** si $(\mathcal{M}, \bullet)$ est un monoïde dont les éléments sont appelés des **ressources**, $\mathcal{P}$ est un ensemble de **produits** et $\succ \subseteq \mathcal{M} \times \mathcal{P}$ est une relation binaire. On précise que l'espace est dit **fini** si l'ensemble $\mathcal{M}$ est fini.*

Pour tout produit $p \in \mathcal{P}$, nous définissons la **section** $\mathcal{M}_p \subseteq \mathcal{M}$ par $\mathcal{M}_p \triangleq \{m \mid m \succ p\}$. On voit que cette définition est très proche de celle d'une section initiale — on aurait $\mathcal{M}_p = {\downarrow}p$, mis à part que $\leqslant$ a été remplacé par $\succ$, voir chapitre 2. La famille des sections $(\mathcal{M}_p)_{p \in \mathcal{P}}$ est appelée **famille des faits de base**. Par exemple, si p est une pomme alors $\mathcal{M}_p$ est l'ensemble des sommes qui permettent d'acheter une pomme.

4.4.3 Plus grande clôture stable

Nous définissons[49] l'opérateur $(\cdot)^{\succ}$ sur $\mathbb{P}(\mathcal{M})$ de la manière suivante :

$$X^{\succ} \triangleq \left\{x \in \mathcal{M} \mid \forall m, p\ (m \bullet X \succ p \Rightarrow m \bullet x \succ p)\right\} \tag{4.2}$$

Que signifie cette définition ? Nous verrons dans quelques instants que $(\cdot)^{\succ}$ est un opérateur de clôture. Ainsi la ressource x se trouve la clôture de X si dans n'importe quel contexte de ressources $m \bullet (\cdot)$ tout produit engendré par toutes les ressources de X l'est aussi par la ressource x.

Théorème 4.4.1 *$(\cdot)^{\succ}$ est la plus grande clôture stable sur $(\mathcal{M}, \bullet)$ pour laquelle les faits de base $\mathcal{M}_p$ sont clos.*

Démonstration. Vérifions d'abord que $(\cdot)^{\succ}$ est une clôture. Il est clair que $X \subseteq X^{\succ}$. Il est aussi clair que si $X \subseteq Y$ alors $X^{\succ} \subseteq Y^{\succ}$. Il reste donc à vérifier $X^{\succ\succ} \subseteq X^{\succ}$. Soit $x \in X^{\succ\succ}$, il s'agit de montrer que $x \in X^{\succ}$. Soit $m \in \mathcal{M}$ et $p \in \mathcal{P}$ tels que $m \bullet X \succ p$. Or par définition de $(\cdot)^{\succ}$, il vient $m \bullet X^{\succ} \succ p$. Donc comme $x \in X^{\succ\succ}$, il vient $m \bullet x \succ p$. $(\cdot)^{\succ}$ est donc une clôture.

Montrons maintenant que cette clôture est stable. Il s'agit de montrer $X \bullet Y^{\succ} \subseteq (X \bullet Y)^{\succ}$ pour X et Y sous-ensembles de $\mathcal{M}$. Soit $x \in X$ et $y \in Y^{\succ}$. Montrons que $x \bullet y \in (X \bullet Y)^{\succ}$. Soit donc m et p tels que $m \bullet (X \bullet Y) \succ p$. On a donc a fortiori $m \bullet (x \bullet Y) \succ p$ puisque $x \bullet Y \subseteq X \bullet Y$. D'où par associativité, $(m \bullet x) \bullet Y \succ p$. En considérant $m \bullet x$ et p, comme $y \in Y^{\succ}$, on en déduit

[48] Uniquement en saison.

[49] Dans cette définition, il est implicite que la variable m prend ses valeurs dans $\mathcal{M}$ et la variable p dans $\mathcal{P}$. Nous ne l'avons pas écrit et nous le ferons pas non plus par la suite pour ne pas alourdir les notations, lorsque les ensembles $\mathcal{M}$ et $\mathcal{P}$ seront des structures aux noms longs et complexes.

$(m \bullet x) \bullet y \succ p$ puis finalement $m \bullet (x \bullet y) \succ p$. Donc $x \bullet y \in (X \bullet Y)^{\succ}$. La clôture $(\cdot)^{\succ}$ est donc stable.

Enfin, on pourra constater que $m \bullet X \succ p \Leftrightarrow X \subseteq \{m\} \multimap \mathcal{M}_p$ et par conséquent

$$X^{\succ} = \bigcap \{ \{m\} \multimap \mathcal{M}_p \mid m \in \mathcal{M}, p \in \mathcal{P} \text{ et } X \subseteq \{m\} \multimap \mathcal{M}_p \} \tag{4.3}$$

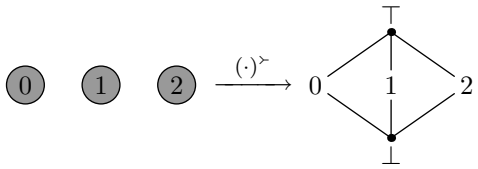
Ainsi si $(\cdot)^{\star}$ est une autre clôture stable pour laquelle les $\mathcal{M}_p$ sont clos, alors d'après la propriété 2.4.2 en page 29, les $\{m\} \multimap \mathcal{M}_p$ sont $(\cdot)^{\star}$-clos également. Et donc $X^{\star} = \bigcap\{Y^{\star} \mid X \subseteq Y^{\star}\} \subseteq \bigcap\{\{m\} \multimap \mathcal{M}_p \mid X \subseteq \{m\} \multimap \mathcal{M}_p\} = X^{\succ}$. Donc $(\cdot)^{\succ}$ est bien la plus grande clôture stable telle que les $\mathcal{M}_p$ soient $(\cdot)^{\succ}$-clos. □

Une remarque importante que l'on peut faire à propos de l'équation (4.3) est que l'opérateur $(\cdot)^{\succ}$ ne dépend que de l'ensemble des sections $\{\mathcal{M}_p \mid p \in \mathcal{P}\}$ de $\succ$. Nous avons choisi une relation plutôt qu'un ensemble de faits de base par souci de cohérence avec la section sur la complétion des monoïdes ordonnés 2.5. En fait, le résultat précédent généralise le théorème 2.5.4.

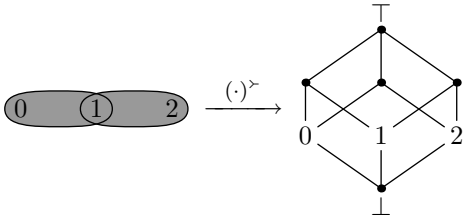
Comme nous l'avons mentionné en section 4.3, à partir d'un espace de phases, il est possible de fabriquer une quantale, et donc de donner un interprétation aux formules de LLI. Mais donnons tout d'abord quelques exemples.

4.4.4 Exemples

Nous présentons deux exemples d'espaces de ressources et leur quantale associée. Sur les figures suivantes, la structure de treillis a été représentée. On considère $\mathcal{M} \triangleq \mathbb{Z}/3\mathbb{Z} = \{0, 1, 2\}$ le groupe cyclique d'ordre trois. Nous représentons la relation $\succ$ par ses sections qui sont grisées. Dans le premier cas, nous avons trois sections, $\{0\}$, $\{1\}$ et $\{2\}$ et nous obtenons la quantale suivante, qui est la même que celle obtenue par $(\cdot)^{\circ}$-complétion du monoïde plat :



Dans le second exemple, nous avons deux sections $\{0, 1\}$ et $\{1, 2\}$. Nous obtenons la même quantale que celle obtenue par $\downarrow(\cdot)$-complétion du monoïde plat :



Si on considère un espace de phases $(\mathcal{M}, \bullet, (\cdot)^{\star})$ avec la relation $\in$ d'appartenance entre éléments de $\mathcal{M}$ et parties closes dans $\mathcal{M}^{\star}$, on obtient un espace de ressources $(\mathcal{M}, \mathcal{M}^{\star}, \bullet, \in)$. On peut alors vérifier que l'on a l'identité $(\cdot)^{\in} = (\cdot)^{\star}$ et donc la quantale obtenue est la même dans les deux cas. Ceci décrit une correspondance entre espaces de phases et espaces de ressources.

Si on considère enfin un monoïde préordonné $(\mathcal{M}, \bullet, \leqslant)$, nous pouvons définir l'espace de ressources $(\mathcal{M}, \mathcal{M}, \bullet, \leqslant)$. Nous obtenons alors l'identité $(\cdot)^{\leqslant} = (\cdot)^{\circ}$. Voici donc une correspondance entre la sémantique des ressources et la sémantique des monoïdes préordonnés que nous décrirons en section 4.5.

4.4.5 Interprétation

Étant donné un espace de ressources $(\mathcal{M}, \mathcal{P}, \bullet, \succ)$, on considère l'espace de phases $(\mathcal{M}, \bullet, (\cdot)^\succ)$ et la quantale $(\mathcal{M}^\succ, (\cdot \bullet \cdot)^\succ, \subseteq)$ associés. On peut interpréter les formules de LLI dans cette quantale. Soit $\sigma : \mathsf{Var} \to \mathcal{P}$ une valuation alors en considérant la valuation $V \mapsto \mathcal{M}_{\sigma(V)}$ dans $\mathcal{M}^\succ$, on obtient l'interprétation suivante, qui ne fait qu'expliciter la sémantique des phases sur notre cas particulier $(\mathcal{M}, \bullet, (\cdot)^\succ)$:

Sémantique des ressources

$$\begin{array}{rlrl}
[\![V]\!] &\triangleq \{m \mid m \succ \sigma(V)\} & [\![A \multimap B]\!] &\triangleq [\![A]\!] \multimap [\![B]\!] \\
[\![\mathsf{T}]\!] &\triangleq \mathcal{M} & [\![A \,\&\, B]\!] &\triangleq [\![A]\!] \cap [\![B]\!] \\
[\![\mathsf{F}]\!] &\triangleq \{x \mid \forall m, p\ m \bullet x \succ p\} & & \\
[\![1]\!] &\triangleq \{x \mid \forall m, p\ (m \succ p \Rightarrow m \bullet x \succ p)\} & & \\
[\![A \otimes B]\!] &\triangleq \{x \mid \forall m, p\ (m \bullet [\![A]\!] \bullet [\![B]\!] \succ p \Rightarrow m \bullet x \succ p)\} & & \\
[\![A \oplus B]\!] &\triangleq \{x \mid \forall m, p\ (m \bullet [\![A]\!] \succ p \text{ and } m \bullet [\![B]\!] \succ p \Rightarrow m \bullet x \succ p)\} & &
\end{array}$$

De manière informelle, une ressource x peut produire $A \otimes B$ (i.e. $x \in [\![A \otimes B]\!]$ si dans tout contexte $m \bullet (\cdot)$ de ressources, x peut engendrer chaque produit p qui peut l'être par une composition d'une ressource de $[\![A]\!]$ et d'une ressource de $[\![B]\!]$. Dans le cas de $A \oplus B$, on n'obtient pas une composition mais un choix, soit une ressource de $[\![A]\!]$, soit une de $[\![B]\!]$. Les ressources de $[\![\mathsf{F}]\!]$ sont celles qui peuvent produire n'importe quoi dans n'importe quel contexte, comme une somme infinie d'argent par exemple.

Théorème 4.4.2 (Adéquation) *Si* $\Gamma \Vdash A$ *alors* $[\![\Gamma]\!] \subseteq [\![A]\!]$.

Ce résultat découle immédiatement de l'adéquation de la sémantique des phases bien sûr, puisque la sémantique des ressources en est un cas particulier. La réciproque est vraie également : la sémantique des phases est un cas particulier de la sémantique des ressources et en voici l'explication.

Soit $(\mathcal{M}, \bullet, (\cdot)^\star)$ un espace de phases, $(\cdot)^\star$ est donc une clôture stable. Alors nous pouvons considérer l'espace de ressources $(\mathcal{M}, \mathcal{M}^\star, \bullet, \in)$ où $\in$ représente l'appartenance ensembliste entre un élément de $\mathcal{M}$ et une partie $(\cdot)^\star$-close de $\mathcal{M}$. Dans cet espace de ressources, les sections sont exactement les parties $(\cdot)^\star$-closes. Il est alors facile de vérifier que l'on a $X^\star = X^{\in}$. Et donc les deux structures définissent la même interprétation. Ceci peut nous permettre d'obtenir un résultat de complétude pour la sémantique des ressources mais en fait nous en donnons une preuve directe.

4.4.6 Quand les ressources sont des contextes

Nous étudions maintenant un espace de ressources particulier, c'est-à-dire l'espace syntaxique où les ressources sont des contextes Γ, les produits, des formules A et la relation de production $\succ$ est une relation close pour les règles de LLI exceptée la règle de coupure [cut], système que l'on note $\mathsf{Gil}_{\mathsf{sc}}$. Ainsi on pourra par exemple considérer $\succ \triangleq \vdash_{\mathsf{sc}}$, c'est-à-dire la prouvabilité sans la règle de coupure, comme cas particulier de $\succ$. Le choix $\succ \triangleq \Vdash (= \vdash_{\mathsf{Gil}})$ conviendra également.

Fixons tout d'abord quelques notations. On pose donc $\mathcal{M} \triangleq \mathsf{Context}$. Vus comme des multi-ensembles de formules, nous aurions pu écrire $\Gamma + \Delta$ pour dénoter la combinaison des contextes Γ et Δ. Mais cette dernière notation peut porter à confusion avec la conjonction additive $\oplus$. Vus comme des parties gauches de séquents, nous aurions plutôt écrit Γ, Δ. Mais cette dernière notation est très ambiguë hors d'un séquent car la virgule sert aussi dans le discours. Nous écrirons

donc $\Gamma \bullet \Delta$ pour la composition des contextes. Et l'élément neutre sera noté e. Il existe aussi une ambiguïté lorsque l'on écrit $\{A\}$ pour une formule logique A. S'agit-il d'un singleton contenant une formule, ou alors contenant un contexte, ou alors s'agit-il d'un contexte qui contient juste la formule A. Pour lever cette ambiguïté, nous définissons la fonction identité[50] $\lfloor\cdot\rfloor : \mathsf{Context} \to \mathsf{Context}$. Ainsi, quand nous écrivons $\lfloor A \rfloor$, il s'agit d'un multi-ensemble constitué d'une unique formule A. L'ambiguïté est en fait levée par le type. $\lfloor A_1, \ldots, A_n \rfloor$ est donc un contexte (multi-ensemble) de formules qui en contient n.

Soit $\succ \subseteq \mathsf{Context} \times \mathsf{Form}$ une relation binaire entre contextes et formules, comme par exemple $\Vdash$.

Définition 4.4.2 (Espace de ressources syntaxique) *Un **espace de ressources syntaxique** est un espace de ressources de la forme* $(\mathsf{Context}, \mathsf{Form}, \bullet, \succ)$ *où* $\succ$ *est une relation close pour les règles de* $\mathsf{Gil_{sc}}$.

Soit $(\mathsf{Context}, \mathsf{Form}, \bullet, \succ)$ un espace de ressource syntaxique. Nous pouvons naturellement considérer les sections $\mathsf{Context}_A = \{\Gamma \mid \Gamma \succ A\}$ dans cet espace de ressources. Afin de ne pas alourdir les notations nous écrirons $\underline{A} \triangleq \mathsf{Context}_A$ de manière provisoire. Les règles de $\mathsf{Gil_{sc}}$ peuvent maintenant se réécrire sous une forme sémantique :

Proposition 4.4.3 *La relation* $\succ$ *est close pour les règles de* $\mathsf{Gil_{sc}}$ *si et seulement si elle vérifie les propriétés suivantes, pour toutes formules* A *et* B *:*

[id]	$\lfloor A \rfloor \in \underline{A}$

$[1_L]$	$\lfloor 1 \rfloor \in \mathsf{e}^{\succ}$	$[1_R]$	$\mathsf{e} \in \underline{1}$
$[\mathsf{F}_L]$	$\lfloor \mathsf{F} \rfloor \in \emptyset^{\succ}$	$[\mathsf{T}_R]$	$\mathsf{Context} \subseteq \top$
$[\otimes_L]$	$\lfloor A \otimes B \rfloor \in \lfloor A, B \rfloor^{\succ}$	$[\otimes_R]$	$\underline{A} \bullet \underline{B} \subseteq \underline{A \otimes B}$
$[\oplus_L]$	$\lfloor A \oplus B \rfloor \in \{\lfloor A \rfloor, \lfloor B \rfloor\}^{\succ}$	$[\oplus_R^{1,2}]$	$\underline{A} \cup \underline{B} \subseteq \underline{A \oplus B}$
$[\&_L^{1,2}]$	$\lfloor A \,\&\, B \rfloor \in \lfloor A \rfloor^{\succ} \cap \lfloor B \rfloor^{\succ}$	$[\&_R]$	$\underline{A} \cap \underline{B} \subseteq \underline{A \,\&\, B}$
$[\multimap_L]$	$\lfloor A \multimap B \rfloor \in \underline{A} \multimap\!\!\bullet \lfloor B \rfloor^{\succ}$	$[\multimap_R]$	$\{\lfloor A \rfloor\} \multimap\!\!\bullet \underline{B} \subseteq \underline{A \multimap B}$

Démonstration. Chaque relation du tableau correspond exactement à une règle d'inférence, celle qui est notée à sa gauche. En fait, il n'y a pas grand chose à prouver. Il suffit de prendre les règles une à une et de vérifier qu'il ne s'agit que d'une réécriture de la règle sous une forme sémantique.

Considérons par exemple la règle [id]. $\lfloor A \rfloor \in \underline{A}$ signifie $\lfloor A \rfloor \succ A$ ce qui veut encore dire que $\succ$ est close pour la règle [id]. Soit maintenant la règle $[\otimes_L]$. On réécrit $\lfloor A \otimes B \rfloor \in \lfloor A, B \rfloor^{\succ}$ en $\forall \Gamma, C\ (\lfloor \Gamma, A, B \rfloor \succ C \Rightarrow \lfloor \Gamma, A \otimes B \rfloor \succ C)$ ce qui signifie exactement que $\succ$ est close pour $[\otimes_L]$. Enfin prenons le cas de $[\otimes_R]$. Dans ce cas, $\underline{A} \bullet \underline{B} \subseteq \underline{A \otimes B}$ se réécrit

$$\forall \Gamma, \Delta\ (\lfloor \Gamma \rfloor \succ A \text{ et } \lfloor \Delta \rfloor \succ B) \Rightarrow \lfloor \Gamma, \Delta \rfloor \succ A \otimes B$$

ce qui veut précisément dire que $\succ$ est close pour $[\otimes_L]$. □

Chacune des équations précédentes est l'exacte traduction sémantique du fait que la relation $\succ$ est close pour la règle d'inférence correspondant à cette équation. Cependant, elles apparaissent en premier lieu comme nécessaires au bon déroulement de la démonstration du lemme qui va

[50] Le lecteur pourra trouver surprenant d'introduire cette fonction mais nous signalons que nous serons amenés à changer le sens de $\lfloor\cdot\rfloor$ par la suite, en le transformant en une projection — voir section 4.4.9. Cette notation a déjà été utilisée en section 4.3.2.

suivre. Ce n'est que par la suite que nous nous sommes rendu compte de la correspondance avec les règles d'inférences. Ceci nous suggère que cette sémantique des ressources est la duale naturelle de la formulation de $\mathsf{Gil}_{\mathsf{sc}}$ sous forme de calcul des séquents.

Nous pouvons considérer la sémantique des ressources dans l'espace syntaxique de ressources $(\mathsf{Context}, \mathsf{Form}, \bullet, \succ)$, et la valuation $\sigma(V) \triangleq V \in \mathsf{Form}$ qui diagonalise, c'est-a-dire qui interprète une formule par elle-même. Okada dans [Oka 99, Oka] a introduit cette technique de preuve adaptant de manière naturelle une méthode issue de la sémantique algébrique.

Lemme 4.4.4 *Pour toute formule C, on a la double relation $\lfloor C \rfloor \in [\![C]\!] \subseteq \underline{C} (= \mathsf{Context}_C)$.*

Démonstration. La preuve se fait par induction structurelle sur C, en utilisant les relations que nous venons de présenter dans la proposition 4.4.3. Le cas de base se présente lorsque C est une formule atomique, c'est-a-dire une variable ou bien une constante :

Si $\boldsymbol{C \equiv X}$ alors $[\![X]\!] = \{\Gamma \mid \lfloor \Gamma \rfloor \succ \sigma(X)\} = \underline{X}$. D'autre part, d'après [id], on a $\lfloor X \rfloor \in \underline{X} = [\![X]\!]$;

Si $\boldsymbol{C \equiv 1}$ alors $[\![C]\!] = \mathsf{e}^{\succ}$ et donc d'après $[1_L]$, on a $\lfloor 1 \rfloor \in [\![1]\!]$. Réciproquement, d'après $[1_R]$, $\mathsf{e} \in \underline{1}$, et donc, $\underline{1}$ étant une section, elle est $(\cdot)^{\succ}$-close d'après le résultat 4.4.1, et ainsi $[\![1]\!] = \mathsf{e}^{\succ} \subseteq \underline{1}$;

Si $\boldsymbol{C \equiv \mathsf{T}}$ alors $[\![\mathsf{T}]\!] = \underline{\top} = \mathsf{Context}$ d'après $[\mathsf{T}_R]$;

Si $\boldsymbol{C \equiv \mathsf{F}}$ alors $[\![\mathsf{F}]\!] = \emptyset^{\succ} \subseteq \underline{\mathsf{F}}$ parce que $\emptyset^{\succ}$ est la plus petite partie close, et $\lfloor \mathsf{F} \rfloor \in [\![\mathsf{F}]\!]$ d'après $[\mathsf{F}_L]$;

Si $\boldsymbol{C \equiv A \otimes B}$ alors $[\![A \otimes B]\!] = ([\![A]\!] \bullet [\![B]\!])^{\succ}$. Or d'après $[\otimes_R]$ et par hypothèse d'induction, on a $[\![A]\!] \bullet [\![B]\!] \subseteq \underline{A} \bullet \underline{B} \subseteq \underline{A \otimes B}$. Cette dernière section étant $(\cdot)^{\succ}$-close, on a donc $[\![A \otimes B]\!] = ([\![A]\!] \bullet [\![B]\!])^{\succ} \subseteq \underline{A \otimes B}$. D'autre part, $\lfloor A, B \rfloor \in [\![A]\!] \bullet [\![B]\!]$ par hypothèse d'induction et donc $\lfloor A \otimes B \rfloor \in \lfloor A, B \rfloor^{\succ} \subseteq ([\![A]\!] \bullet [\![B]\!])^{\succ} = [\![A \otimes B]\!]$;

Si $\boldsymbol{C \equiv A \oplus B}$ alors $[\![A \oplus B]\!] = ([\![A]\!] \cup [\![B]\!])^{\succ} \subseteq (\underline{A} \cup \underline{B})^{\succ} \subseteq (\underline{A \oplus B})^{\succ} = \underline{A \oplus B}$ d'après $[\oplus_R^{1,2}]$. Par ailleurs $\lfloor A \oplus B \rfloor \in \{\lfloor A \rfloor, \lfloor B \rfloor\}^{\succ} \subseteq ([\![A]\!] \cup [\![B]\!])^{\succ} = [\![A \oplus B]\!]$ par $[\oplus_L]$.

Si $\boldsymbol{C \equiv A \,\&\, B}$ alors $[\![A \,\&\, B]\!] = [\![A]\!] \cap [\![B]\!] \subseteq \underline{A} \cap \underline{B} \subseteq \underline{A \,\&\, B}$ par $[\&_R]$. D'autre part, $\lfloor A \,\&\, B \rfloor \in \lfloor A \rfloor^{\succ} \cap \lfloor B \rfloor^{\succ} \subseteq [\![A]\!] \cap [\![B]\!]$ par $[\&_L^{1,2}]$ et le fait que $[\![A]\!]$ et $[\![B]\!]$ sont deux parties $(\cdot)^{\succ}$-closes et donc, à l'aide de l'hypothèse d'induction, on a aussi $\lfloor A \rfloor^{\succ} \subseteq [\![A]\!]$ et $\lfloor B \rfloor^{\succ} \subseteq [\![B]\!]$.

Si $\boldsymbol{C \equiv A \multimap B}$ alors $[\![A \multimap B]\!] = [\![A]\!] \multimap [\![B]\!] \subseteq \{\lfloor A \rfloor\} \multimap \underline{B} \subseteq \underline{A \multimap B}$ par $[\multimap_R]$. On se sert ici des hypothèses $\lfloor A \rfloor \in [\![A]\!]$ et $[\![B]\!] \subseteq \underline{B}$. D'autre part, $\lfloor A \multimap B \rfloor \in \underline{A} \multimap \lfloor B \rfloor^{\succ} \subseteq [\![A]\!] \multimap [\![B]\!] = [\![A \multimap B]\!]$ par $[\multimap_R]$. On utilise ici des hypothèses $[\![A]\!] \subseteq \underline{A}$ et $\lfloor B \rfloor \in [\![B]\!]$.

□

Le lecteur averti aura pu remarquer que dans le cas de l'implication linéaire, on « croise » l'utilisation des hypothèses d'induction par contra-variance de cet opérateur en son premier argument. Une autre remarque importante est que l'on ne prouve pas $[\![C]\!] = \underline{C}$ comme c'est le cas dans le résultat 4.2.3. Nous pourrions le dériver ici mais nous aurions besoin de la règle de coupure [cut] qui force les ressources et les produits à partager la même structure monoïdale. Le résultat précédent $\lfloor C \rfloor \in [\![C]\!] \subseteq \underline{C}$ est plus faible en apparence, mais est obtenu avec une hypothèse en moins — la clôture de la relation $\succ$ par rapport à la règle de coupure [cut] — et nous allons montrer qu'il suffit pour obtenir la complétude. On obtient même encore beaucoup mieux que cela, une preuve sémantique de l'élimination des coupures et avec quelques manipulations supplémentaires, la propriété des modèles finis.

4.4.7 Complétude et élimination des coupures

De plus, le lemme 4.4.4 nous permet de prouver très simplement la complétude de la sémantique des ressources, l'espace de ressources syntaxique étant un contre-modèle de toutes les formules invalides.

Théorème 4.4.5 (Complétude) *Si A est une formule invalide $\not\Vdash A$ alors pour l'interprétation définie dans l'espace syntaxique des ressources* $(\mathsf{Context}, \mathsf{Form}, \bullet, \Vdash)$ *par la valuation $V \mapsto V$, on a $\lfloor \emptyset \rfloor \notin [\![A]\!]$. Par conséquent, nous avons un contre-modèle de A.*

Démonstration. On a donc choisi d'instancier $\succ = \Vdash$, ce qui est possible puisque $\Vdash$ est close pour toutes les règles de Gil, y compris [cut], ce qui n'est pas utile dans notre cas. Donc si A est invalide, cela signifie $\lfloor \emptyset \rfloor \not\Vdash A$ et par conséquent, $\lfloor \emptyset \rfloor \notin \underline{A}$. Or comme $[\![A]\!] \subseteq \underline{A}$, on a aussi $\lfloor \emptyset \rfloor \notin [\![A]\!]$. □

Ce qui est très « séduisant, » dans cette preuve c'est que l'on obtient l'élimination des coupures simplement en remplaçant $\Vdash$ par $\vdash_{\mathsf{sc}}$ dans la preuve précédente, $\vdash_{\mathsf{sc}}$ étant la plus petite relation close pour les règles de $\mathsf{Gil}_{\mathsf{sc}}$.

Théorème 4.4.6 (Élimination des coupures) *Si une formule A est prouvable dans* LLI *alors elle l'est aussi dans* $\mathsf{Gil}_{\mathsf{sc}}$.

Démonstration. On instancie ici par $\succ$ par $\vdash_{\mathsf{sc}}$. Soit A une formule valide, supposons que $\nvdash_{\mathsf{sc}} A$, autrement dit A n'est pas prouvable dans $\mathsf{Gil}_{\mathsf{sc}}$. Si on considère l'espace de ressources $(\mathsf{Context}, \mathsf{Form}, \bullet, \vdash_{\mathsf{sc}})$, on a donc $\lfloor \emptyset \rfloor \notin [\![A]\!]$ pour la même raison que précédemment, ce qui nous donne un contre-modèle de A. Ceci est en contradiction avec la validité de A. Par l'absurde, A est donc forcément prouvable dans $\mathsf{Gil}_{\mathsf{sc}}$. □

À notre connaissance, les premiers travaux faisant mention du remplacement de la condition $[\![C]\!] = \underline{C}$ par la condition plus faible $\lfloor C \rfloor \in [\![C]\!] \subseteq \underline{C}$ sont dus à Okada [Oka 99] ; le but étant d'obtenir une preuve sémantique de l'élimination des coupures pour la logique classique linéaire, preuve qu'il généralise d'ailleurs à l'ordre supérieur. La dissymétrie du cas intuitionniste que l'on retrouve dans la distinction ressources/produits n'est abordée que plus tard, dans le cadre de la sémantique des phases [Oka]. De plus, une forme particulière d'arbres de réfutation est utilisée pour construire des espaces de phases finis par un passage au quotient implicite.

Nous allons montrer que nous pouvons adapter le passage au quotient aux espaces de ressources syntaxique de manière à en extraire des contre-modèles finis.

4.4.8 Espace de ressources quotient

Étant donné un espace de ressources $(\mathcal{M}, \mathcal{P}, \bullet, \succ)$ nous avons vu que nous pouvons construire un espace de phases $(\mathcal{M}, \bullet, \subseteq)$ et définir une interprétation à partir de celui-ci. Mais nous avons aussi vu comment passer au quotient dans un espace de phases, c.f. section 2.4.2. Nous montrons maintenant comment combiner ces deux résultats.

Définition 4.4.3 (Espace de ressources quotient) *Soit $(\mathcal{M}, \mathcal{P}, \bullet, \succ)$ un espace de ressources et $(\mathcal{Q}, \bullet)$ un monoïde quotient ayant $\lfloor \cdot \rfloor : \mathcal{M} \twoheadrightarrow \mathcal{Q}$ pour projection. Le triplet $(\mathcal{Q}, \bullet, \lfloor \cdot \rfloor)$ est appelé **quotient** de $\mathcal{M}$ si on a $\forall p \in \mathcal{P}$, $x \succ p \Leftrightarrow y \succ p$ si $\lfloor x \rfloor = \lfloor y \rfloor$.*

Étant donné un quotient $(\mathcal{Q}, \bullet, \lfloor\cdot\rfloor)$ de $\mathcal{M}$, on définit une relation $\rhd \subseteq \mathcal{Q}\times\mathcal{P}$ par l'équivalence $\lfloor x\rfloor \rhd p \Leftrightarrow x \succ p$. Cette définition est valide à cause de la propriété $\forall p \in \mathcal{P},\ x \succ p \Leftrightarrow y \succ p$. On en déduit immédiatement les sections de $\mathcal{Q}$. Pour tout $p \in \mathcal{P}$, on a $\mathcal{Q}_p = \lfloor \mathcal{M}_p \rfloor$.

Théorème 4.4.7 *Soit $(\mathcal{M}, \mathcal{P}, \bullet, \succ)$ un espace de ressources et $(\mathcal{Q}, \bullet, \lfloor\cdot\rfloor)$ un quotient $\mathcal{M}$. Alors $(\mathcal{Q}, \mathcal{P}, \bullet, \rhd)$ est un espace de ressources et son espace de phases $(\mathcal{Q}, \bullet, (\cdot)^{\rhd})$ est un quotient de $(\mathcal{M}, \bullet, (\cdot)^{\succ})$ par la projection $\lfloor\cdot\rfloor$, autrement dit, on a l'équivalence $\lfloor x\rfloor \in \lfloor X\rfloor^{\rhd} \Leftrightarrow x \in X^{\succ}$.*

Démonstration. Il est clair que $\mathcal{Q}$ muni de la relation $\rhd$ est un espace de ressources. D'autre part $\lfloor\cdot\rfloor$ est un morphisme de monoïde surjectif par hypothèse. Il s'agit donc de vérifier l'équivalence $\lfloor x\rfloor \in \lfloor X\rfloor^{\rhd} \Leftrightarrow x \in X^{\succ}$ pour $x \in \mathcal{M}$ et $X \subseteq \mathcal{M}$:

$$\begin{aligned}
\lfloor x\rfloor \in \lfloor X\rfloor^{\rhd} &\Leftrightarrow \forall q \in \mathcal{Q}, \forall p,\ q \bullet \lfloor X\rfloor \rhd p \Rightarrow q \bullet \lfloor x\rfloor \rhd p \\
&\Leftrightarrow \forall m \in \mathcal{M}, \forall p,\ \lfloor m\rfloor \bullet \lfloor X\rfloor \rhd p \Rightarrow \lfloor m\rfloor \bullet \lfloor x\rfloor \rhd p \\
&\Leftrightarrow \forall m, p,\ \lfloor m \bullet X\rfloor \rhd p \Rightarrow \lfloor m \bullet x\rfloor \rhd p \\
&\Leftrightarrow \forall m, p,\ m \bullet X \succ p \Rightarrow m \bullet x \succ p \\
&\Leftrightarrow x \in X^{\succ}
\end{aligned}$$

□

Ainsi, nous pouvons aussi passer au quotient dans les espaces de ressources à condition que ce quotient respecte la relation $\succ$, c'est-a-dire vérifie la condition $\forall p \in \mathcal{P},\ x \succ p \Leftrightarrow y \succ p$ pour des éléments x et y identifiés par le quotient, i.e. $\lfloor x\rfloor = \lfloor y\rfloor$. Si $\sigma : \mathsf{Var} \to \mathcal{M}$ est une valuation dans $\mathcal{M}$ alors $\sigma' : \mathsf{Var} \to \mathcal{Q}$ définie par $\sigma'(V) \triangleq \lfloor\sigma(V)\rfloor$ est une valuation dans $\mathcal{Q}$, et pour les interprétations $[\![\cdot]\!]$ et $[\![\cdot]\!]'$ correspondantes, on a la relation $[\![A]\!]' = \lfloor [\![A]\!] \rfloor$ par application du résultat 4.3.2.

4.4.9 Quotient par un arbre de réfutation

Dans cette section, nous considérons un arbre de réfutation $\mathcal{R}$ fixé pour le système $\mathsf{Gil_{sc}}$. Comme $\mathsf{Gil_{sc}}$ est un système analytique — voir section 4.1, — cet arbre de réfutation est fini. Ceci n'est qu'un rappel des résultats 1.5. Nous commençons par définir un monoïde quotient de $(\mathsf{Context}, \bullet)$ où tous les contextes qui n'apparaissent pas comme sous-contextes des index[51] de $\mathcal{R}$ sont identifiés, c'est-à-dire appartiennent à la même classe d'équivalence, ce qui fait que ce monoïde quotient est fini. Il est alors possible de construire un espace de ressources fini sur la base de ce monoïde quotient et d'un déduire un (contre-)modèle fini.

Soit donc un arbre de réfutation (fini car on est dans $\mathsf{Gil_{sc}}$) noté $\mathcal{R}$. Si $\Gamma \vdash A$ est un séquent, on notera $\Gamma \vdash A \in \mathcal{R}$ pour exprimer le fait que ce séquent est l'index de l'un des nœuds de $\mathcal{R}$. On choisit une variable logique arbitraire Y qui n'apparaît pas dans $\mathcal{R}$ et on note Π le contexte qui ne contient que la formule logique Y, i.e. $\Pi \triangleq \{Y\}$. On définit alors

$$\begin{aligned}
\mathsf{Context}_{\mathcal{R}} &\triangleq \{\Gamma \mid \exists \Sigma, A \text{ tel que } \Sigma, \Gamma \vdash A \in \mathcal{R}\} \\
\mathcal{M} &\triangleq \mathsf{Context}_{\mathcal{R}} \cup \{\Pi\}
\end{aligned}$$

Alors, comme $\mathcal{R}$ est un arbre fini, il contient un nombre fini de sous-contextes — ceux de $\mathsf{Context}_{\mathcal{R}}$ — et par conséquent $\mathcal{M}$ est un ensemble fini. Nous faisons remarquer au lecteur que $\mathcal{M}$

[51] Nous rappelons que les nœuds de $\mathcal{R}$ sont indexés par des séquents

n'est qu'une représentation particulière du monoïde quotient précité, où tous les contextes extérieurs à $\mathcal{R}$ sont identifiés avec Π, puisqu'il est clair que $\Pi \notin \mathsf{Context}_{\mathcal{R}}$. La projection canonique du quotient $\lfloor\cdot\rfloor : \mathsf{Context} \to \mathcal{M}$ est définie par :

$$\lfloor\Gamma\rfloor \triangleq \begin{cases} \Gamma \text{ if } \Gamma \in \mathsf{Context}_{\mathcal{R}} \\ \Pi \text{ otherwise} \end{cases}$$

Cette projection permet de transporter la structure monoïdale de $\mathsf{Context}$ sur $\mathcal{M}$ par la relation $\lfloor\Gamma\rfloor \bullet \lfloor\Delta\rfloor \triangleq \lfloor\Gamma,\Delta\rfloor$ qui définit une opération monoïdale $\bullet$ sur $\mathcal{M}$. Cette définition est valide car si $\lfloor\Gamma\rfloor = \Pi$ ou $\lfloor\Delta\rfloor = \Pi$ alors $\lfloor\Gamma,\Delta\rfloor = \Pi$.

Proposition 4.4.8 $(\mathcal{M}, \bullet, \lfloor\cdot\rfloor)$ *est un quotient fini du monoïde des contextes* $(\mathsf{Context}, \bullet)$.

On considère la relation $\succ \subseteq \mathsf{Context} \times \mathsf{Form}$ définie par $\Gamma \succ A \Leftrightarrow \Gamma \vdash A \notin \mathcal{R}$, on aurait pu noter $\succ \triangleq (\cdot) \notin \mathcal{R}$. D'après le résultat 1.4.1 de la section 1.4, la relation $\succ$ est close pour les règles de $\mathsf{Gil}_{\mathsf{sc}}$. Donc $(\mathsf{Context}, \mathsf{Form}, \bullet, \succ)$ est un espace de ressources syntaxique.

Proposition 4.4.9 *Le triplet* $(\mathcal{M}, \bullet, \lfloor\cdot\rfloor)$ *est un quotient fini de l'espace de ressources syntaxique* $(\mathsf{Context}, \mathsf{Form}, \bullet, \succ)$.

Démonstration. D'après la définition 4.4.3, il s'agit de montrer que si $\lfloor\Gamma\rfloor = \lfloor\Delta\rfloor$ alors pour toute formule A, $\Gamma \succ A \Leftrightarrow \Delta \succ A$. Cette condition se réécrit $\Gamma \vdash A \notin \mathcal{R} \Leftrightarrow \Delta \vdash A \notin \mathcal{R}$.[52] Deux cas se présentent, soit $\lfloor\Gamma\rfloor = \lfloor\Delta\rfloor = \Pi$, soit $\Gamma = \Delta$, ceci par définition de $\lfloor\cdot\rfloor$. Dans le premier cas, ni Γ ni Δ ne sont dans $\mathsf{Context}_{\mathcal{R}}$. Dans ce cas nous avons donc $\Gamma \vdash A \notin \mathcal{R}$ et $\Delta \vdash A \notin \mathcal{R}$. L'équivalence est acquise. Dans l'autre cas, on a $\Gamma = \Delta$ et l'équivalence est triviale. □

4.4.10 Propriété des modèles finis

L'espace des ressources syntaxiques $(\mathsf{Context}, \mathsf{Form}, \bullet, \succ)$ constitue un contre-modèle de toutes les formules invalides mais ce contre-modèle est infini. En passant au quotient par un arbre de réfutation, on obtient un contre-modèle fini de tous les séquents qui sont des index dans cet arbre de réfutation. Tous les ingrédients sont réunis pour prouver la propriété des modèles finis.

Théorème 4.4.10 (Propriété des modèles finis) *Si A est une formule invalide $\nVdash A$ alors il existe un espace de ressources fini qui est un contre-modèle de A.*

Démonstration. Soit $\mathcal{R}$ un arbre de réfutation dans $\mathsf{Gil}_{\mathsf{sc}}$ du séquent $\vdash A$ qui est a fortiori invalide dans $\mathsf{Gil}_{\mathsf{sc}}$.[53] Comme $\mathsf{Gil}_{\mathsf{sc}}$ est un système analytique, $\mathcal{R}$ est fini — voir la section 1.5.1 théorème 1.5.2 pour une justification de l'existence du caractère fini de $\mathcal{R}$. On considère alors l'espace de ressources syntaxique $(\mathsf{Context}, \mathsf{Form}, \bullet, \succ)$ défini dans la section précédente où $\succ = (\cdot) \notin \mathcal{R}$. On interprète alors les variables logiques suivant $\sigma(V) = V$. D'après le résultat 4.4.4, on obtient $[\![A]\!] \subseteq \mathsf{Context}_A$. On considère alors l'espace de ressources quotient $\mathcal{M} = \mathsf{Context}_{\mathcal{R}} \cup \{\Pi\}$ de la section précédente. Son espace des phases $(\mathcal{M}, \bullet, (\cdot)^{\triangleright})$ est alors un espace quotient de l'espace $(\mathsf{Context}, \bullet, (\cdot)^{\succ})$ d'après le résultat 4.4.7. Donc on peut calculer l'interprétation $[\![A]\!]'$ de A dans cet espace $\mathcal{M}$. D'après le résultat 4.3.2, on obtient $[\![A]\!]' = \lfloor[\![A]\!]\rfloor$. Ainsi $[\![A]\!]' \subseteq \lfloor\mathsf{Context}_A\rfloor = \mathcal{M}_A$. Comme $\emptyset \vdash A \in \mathcal{R}$ et $\lfloor\emptyset\rfloor = \emptyset$, on a $\emptyset \not\rhd A$ d'où $\emptyset \notin \mathcal{M}_A$. D'où enfin, $\emptyset \notin [\![A]\!]'$. L'espace des ressources quotient est un contre-modèle fini de A. □

[52] On se rend compte ici qu'on ne peut pas quotienter n'importe comment et que le choix de la relation $(\cdot) \notin \mathcal{R}$ est judicieux.

[53] On n'a pas besoin d'utiliser l'élimination des coupures ici, autrement dit, si une formule n'est pas prouvable dans Gil, elle ne l'est pas non plus dans $\mathsf{Gil}_{\mathsf{sc}}$.

4.4.11 Construction de contre-modèles : un exemple

Dans cette section nous développons la construction d'un contre-modèle au séquent $X, Y \vdash X$ sous la forme d'un espace de ressources. La simplicité du séquent permet de limiter la taille du contre-modèle au raisonnable.

Le séquent $X, Y \vdash X$ est un séquent atomique. Aucune règle ne peut lui être appliquée. Il constitue à lui tout seul un arbre de réfutation appelé $\mathcal{R}$. Nous décrivons l'espace de ressources $(\mathcal{M}, \mathcal{P}, \bullet, \succ)$ où $\mathcal{M}$ est le monoïde quotient des sous-contextes de $\mathcal{R}$ et $\mathcal{P} \triangleq \{X\}$ l'ensemble des sections.[54] D'autre part, on rappelle que $\succ$ est défini par $\succ \triangleq (\cdot) \notin \mathcal{R}$. Nous avons quatre sous-contextes, $\emptyset$ noté 0, X noté 1, Y noté 2 et X, Y noté 3.

$$\begin{array}{rcl} \mathcal{M} & \triangleq & \mathsf{Context}_{\mathcal{R}} \cup \{\Pi\} \\ & = & \{0, 1, 2, 3, \Pi\} \end{array}$$

	$\bullet$	0	1	2	3	Π
$\emptyset$	0	0	1	2	3	Π
X	1	1	Π	3	Π	Π
Y	2	2	3	Π	Π	Π
X, Y	3	3	Π	Π	Π	Π
	Π	Π	Π	Π	Π	Π

$\succ$	X
0	vrai
1	vrai
2	vrai
3	faux
Π	vrai

Calculons maintenant la quantale des parties $(\cdot)^{\succ}$-closes. D'après l'équation (4.3), cette quantale est constituée des intersections quelconques des parties de la forme $\{m\} \multimap \mathcal{M}_p$ où $m \in \mathcal{M}$ et $p \in \mathcal{P}$. Or on constate que $\mathcal{M}_X = \{0, 1, 2, \Pi\}$ et :

$$\begin{array}{rl} \{0\} \multimap \mathcal{M}_X = \{0, 1, 2, \Pi\} & = \mathcal{M} - \{3\} \\ \{1\} \multimap \mathcal{M}_X = \{0, 1, 3, \Pi\} & = \mathcal{M} - \{2\} \\ \{2\} \multimap \mathcal{M}_X = \{0, 2, 3, \Pi\} & = \mathcal{M} - \{1\} \\ \{3\} \multimap \mathcal{M}_X = \{1, 2, 3, \Pi\} & = \mathcal{M} - \{0\} \\ \{\Pi\} \multimap \mathcal{M}_X = \{0, 1, 2, 3, \Pi\} & = \mathcal{M} \end{array}$$

En effet, on a par exemple $\{1\} \multimap \mathcal{M}_X = \{\Gamma \in \mathcal{M} \mid \Gamma, X \vdash X \notin \mathcal{R}\} = \{\Gamma \in \mathcal{M} \mid \Gamma \bullet 1 \neq 3\} = \{0, 1, 3, \Pi\}$. On en déduit la quantale $\mathcal{M}^{\succ} = \{K \mid \Pi \in K \subseteq \mathcal{M}\}$. Enfin, nous pouvons calculer l'interprétation sémantique du séquent $X, Y \vdash X$ dans cette quantale : $[\![X]\!] = \mathcal{M}_X = \{0, 1, 2, \Pi\}$, $[\![Y]\!] = \mathcal{M}$ et $[\![X, Y]\!] = ([\![X]\!] \bullet [\![Y]\!])^{\succ} = \mathcal{M}$. Par conséquent on constate que $[\![X, Y]\!] \not\subseteq [\![X]\!]$ car $3 \notin [\![X]\!]$. Nous avons donc bien un contre-modèle du séquent $X, Y \vdash X$.

Une technique de construction de contre-modèle similaire est décrite dans [Oka]. La notion d'OR-tree qui y est utilisée correspond à la notion de réfutation particularisée à LLI. Les contre-modèles obtenus sont sous la forme d'espaces de phases.

Nous disposons donc d'un algorithme de construction de contre-modèles lorsque l'on possède un arbre de réfutation, arbre qui peut être obtenu pour toute formule (ou séquent) invalide par application de l'algorithme de construction de réfutation de la section 1.5.1 chapitre 1. On pourrait donc penser que l'on a résolu la question de la génération automatique de contre-modèles. Il n'en est rien. Comme nous l'avons vu pour la logique intuitionniste, une réfutation est un objet certes fini, mais très lourd à manipuler. Dans le cas de LI, les modèles de Kripke permettent une réduction substancielle de la taille des objets à manipuler. Dans le cas de LLI, les espaces de ressources obtenus à partir de réfutations sont encore plus grands que les réfutations elles-mêmes. Dans l'exemple précédent, la réfutation est un arbre de taille 1 et la quantale engendrée par l'espace de ressources de taille $16 = 2^{5-1}$. Bien sûr il existe des contre-modèles bien plus petits pour le séquent $X, Y \vdash X$.

[54] Les autres sections $\mathcal{M}_A$ où A est une formule différente de la variable X sont toutes égales à $\mathcal{M}$ car X est la seule formule qui apparaît comme conclusion dans la réfutation.

La mécanisation de la procédure passe par la recherche de contre-modèles ayant des représentations plus compactes. Ceci constitue une motivation importante pour l'étude d'autres sémantiques comme par exemple la sémantique des monoïdes ordonnés —c.f. section 4.5— ou la sémantique à base de réseaux de Petri —c.f. section 4.6.

4.4.12 Application aux sémantiques précédentes

Nous avons vu que la sémantique des ressources est fondée sur la sémantique des phases et la construction de l'espace des phases $(\mathcal{M}, \bullet, (\cdot)^{\succ})$. De même, la sémantique des phases est fondée sur la sémantique algébrique et la construction de la quantale $(\mathcal{M}^{\succ}, (\cdot \bullet \cdot)^{\succ}, \subseteq)$.

La sémantique des phases a donc hérité son adéquation de celle de la sémantique algébrique, que nous avons prouvée en détails en section 4.2.1. De même, la sémantique des ressources hérite son adéquation de celle de la sémantique des phases. Réciproquement, la propriété des modèles finis est transmise à la sémantique des phases, puis à la sémantique algébrique.

Théorème 4.4.11 (Propriété des modèles finis) *La sémantique algébrique et la sémantique des phases ont toutes les deux la propriété des modèles finis.*

Démonstration. Dans la preuve de la propriété des modèles finis pour la sémantique des ressources 4.4.10, on construit un espace des phases quotient $(\mathcal{M}, \bullet, (\cdot)^{\triangleright})$ qui est un contre-modèle fini de A. La quantale correspondante $(\mathcal{M}^{\triangleright}, (\cdot \bullet \cdot)^{\triangleright}, \subseteq)$ est aussi finie et est également un contre-modèle de A. □

4.4.13 Conclusion

Dans cette section, nous avons présenté une nouvelle sémantique de LLI basée sur la notion de ressource. Nous montrons comment il est possible de transformer un espace de ressources en modèle algébrique (quantale) par le biais de la plus grande clôture stable qui préserve les sections, ce qui prouve l'adéquation de cette nouvelle interprétation.

La sémantique des ressources est naturellement adaptée aux preuves de complétude, d'élimination des coupures et de la propriété des modèles finis. Cette dernière démonstration se fonde sur un algorithme de construction de réfutations, issu de la recherche de preuves. Nous montrons comment une réfutation permet de quotienter l'espace de ressources syntaxique des contextes — qui est infini,— pour en faire un contre-modèle fini. Nous en déduisons la propriété des modèles finis et une procédure qui permet de construire un contre-modèle (fini) de n'importe quel formule (ou séquent) invalide. Enfin, nous montrons comment la sémantique algébrique et la sémantique des phases « héritent » de la propriété des modèles finis.

Cependant, la procédure de construction de contre-modèles à partir d'une réfutation engendre des objets de taille importante. Il ne paraît pas envisageable de l'utiliser telle qu'elle en dehors d'un cadre théorique, c'est-à-dire d'en faire un programme qui construit effectivement des contre-modèles. Il paraît nécessaire de trouver des représentations plus compactes et les algorithmes de construction directs correspondant, comme cela a été fait pour LI avec les arbres de Kripke, voir section 3.4.4 chapitre 3. Dans les sections suivantes nous présentons d'autres sémantiques pour LLI à partir desquelles la construction effective de contre-modèles pourrait être plus praticable.

4.5 Sémantique des monoïdes préordonnés

Nous introduisons maintenant une autre interprétation des formules et séquents de LLI basée sur la notion de monoïde ordonné. Au départ, cette sémantique nous est apparue comme une

abstraction naturelle de la sémantique à base de réseaux de Petri, voir les travaux de Winskel [Eng 97] et la section 4.6. En effet, lorsque l'on oublie l'aspect opérationnel — c'est-à-dire l'ordre dans lequel s'effectue les transitions — un réseau de Petri n'est rien d'autre qu'un monoïde ordonné.

On peut aussi voir un monoïde ordonné comme une étape intermédiaire vers la construction d'une quantale, c'est-à-dire une quantale partiellement spécifiée. Ceci peut laisser plus de liberté sur la construction d'un contre-modèle tout en conservant la complétude, comme nous allons le voir.

4.5.1 Choix d'une prétopologie

On considère un monoïde préordonné $(\mathcal{M}, \bullet, \leqslant)$ et une prétopologie $(\cdot)^\star$ sur $\mathcal{M}$, c'est-à-dire une clôture à la fois compatible et stable, voir section 2.5.2 chapitre 2.. Alors d'après la proposition 2.5.6, le triplet $(\mathcal{M}^\star, (\cdot \bullet \cdot)^\star, \subseteq)$ forme une quantale et par conséquent, il est possible d'interpréter les formules de LLI dans $\mathcal{M}^\star$. Si on se donne une valuation $\sigma : \mathsf{Var} \to \mathcal{M}$ alors on obtient les mêmes équations que pour la sémantique des phases, voir en section 4.3.1, page 79.

Cette sémantique est bien évidemment adéquate comme cas particulier de — au choix — la sémantique algébrique ou la sémantique des phases. L'interprétation $[\![F]\!]$ est donc une partie de $\mathcal{M}$ calculée dans la quantale $\mathcal{M}^\star$.

Proposition 4.5.1 *Pour toute formule F, $[\![F]\!]$ est une partie $(\cdot)^\star$-close de $\mathcal{M}$, et donc aussi une partie $\downarrow(\cdot)$-close.*

Il existe deux prétopologies particulières sur $\mathcal{M}$, la plus petite $\downarrow(\cdot)$ et la plus grande $(\cdot)^\circ$. Le choix de l'une d'elles nous donne une sémantique à base de monoïdes préordonnés.

Proposition 4.5.2 *La sémantique correspondant au choix de $\downarrow(\cdot)$ est incomplète pour* LLI. *En particulier, nous avons l'identité $[\![A \,\&\, (B \oplus C)]\!] = [\![(A \,\&\, B) \oplus (A \,\&\, C)]\!]$ dans cette interprétation.*

Démonstration. Les deux formules $A \,\&\, (B \oplus C)$ et $(A \,\&\, B) \oplus (A \,\&\, C)$ ne sont pas équivalentes dans LLI ; on peut par exemple construire un arbre de réfutation ou alors utiliser un contre-exemple sous la forme d'un monoïde plat complété, voir proposition 2.5.9 en page 34. Par contre, on a $[\![E \oplus F]\!] = \downarrow([\![E]\!] \cup [\![F]\!]) = \downarrow[\![E]\!] \cup \downarrow[\![F]\!] = [\![E]\!] \cup [\![F]\!]$. Donc $\oplus$ est interprété par une union ensembliste $\cup$ et & par une intersection $\cap$. Donc ils distribuent l'un sur l'autre. □

Avant de nous intéresser au cas de $(\cdot)^\circ$, nous montrons tout d'abord que les interprétations dans deux monoïdes préordonnés sont « les mêmes » si ces monoïdes sont équivalents.

Définition 4.5.1 (Équivalence entre interprétations) *Soit $(\mathcal{M}_1, \sigma_1)$ et $(\mathcal{M}_2, \sigma_2)$ deux interprétations dans des monoïdes préordonnés. Une **équivalence d'interprétations** est une équivalence $i : \mathcal{M}_1 \rightleftharpoons \mathcal{M}_2$ entre les deux monoïdes préordonnés qui vérifie $\sigma_2(V) \simeq i\big(\sigma_1(V)\big)$ pour toute variable V.*

Proposition 4.5.3 *Soit une équivalence $i : \mathcal{M}_1 \rightleftharpoons \mathcal{M}_2$ et soit $(\cdot)^\star$ l'une des prétopologies $\downarrow(\cdot)$ ou $(\cdot)^\circ$. On note $[\![\cdot]\!]_1$ et $[\![\cdot]\!]_2$ les interprétations respectives correspondant au choix de la même[55] prétopologie $(\cdot)^\star$ pour $\mathcal{M}_1$ et $\mathcal{M}_2$. Alors pour tout formule F, on a l'identité $[\![F]\!]_2 = \downarrow i([\![F]\!]_1)$.*

[55] Au sens de la théorie des types [Nor 90], les prétopologies extrémales sont paramétriques en leur monoïde préordonné.

Démonstration. Pour une variable V, on a

$$[\![V]\!]_2 = \sigma_2(V)^\star = {\downarrow}\sigma_2(V) = {\downarrow}i\big(\sigma_1(V)\big) = {\downarrow}i\big({\downarrow}\sigma_1(V)\big) = {\downarrow}i([\![V]\!]_1)$$

Pour les autres cas, il suffit de voir que $X \in \mathcal{M}_1^\star \mapsto {\downarrow}i(X) \in \mathcal{M}_2^\star$ est un isomorphisme entre les quantales $(\mathcal{M}_1^\star, (\cdot \bullet \cdot)^\star, \subseteq)$ et $(\mathcal{M}_2^\star, (\cdot \bullet \cdot)^\star, \subseteq)$. □

4.5.2 Complétude et contre-modèles finis

On appellera **sémantique des monoïdes préordonnés** l'interprétation correspondant au choix de la plus grande prétopologie $(\cdot)^\circ$ et nous étudions dans cette section les propriétés de complétude de cette sémantique.

La quantale $(\mathcal{M}^\circ, (\cdot \bullet \cdot)^\circ, \subseteq)$ peut-être vue soit comme provenant de l'espace de phases $(\mathcal{M}, \bullet, (\cdot)^\circ)$, voir section 2.5 au chapitre 2, soit comme provenant de l'espace de ressources $(\mathcal{M}, \mathcal{M}, \bullet, \leqslant)$ avec $(\cdot)^\leqslant = (\cdot)^\circ$, voir section 4.4.4.

> On peut voir la sémantique des monoïdes ordonnés comme un cas particulier de la sémantique des ressources, lorsque les produits et les ressources sont de même nature.

Si on choisit une valuation $\sigma : \mathsf{Var} \to \mathcal{M}$ on obtient l'interprétation suivante :

Sémantique des monoïdes préordonnés

$$\begin{array}{rcl@{\qquad}rcl}
[\![V]\!] & \triangleq & {\downarrow}\sigma(V) & [\![A \multimap B]\!] & \triangleq & [\![A]\!] \multimap [\![B]\!] \\
[\![\mathsf{T}]\!] & \triangleq & \mathcal{M} & [\![A \,\&\, B]\!] & \triangleq & [\![A]\!] \cap [\![B]\!] \\
[\![1]\!] & \triangleq & {\downarrow}\mathsf{e} & [\![\mathsf{F}]\!] & \triangleq & \{x \mid \forall m, p\ m \bullet x \leqslant p\} \\
\multicolumn{3}{r}{[\![A \otimes B]\!] \triangleq \{x \mid \forall m, p\ (m \bullet [\![A]\!] \bullet [\![B]\!] \leqslant p \Rightarrow m \bullet x \leqslant p)\}} & & & \\
\multicolumn{3}{r}{[\![A \oplus B]\!] \triangleq \{x \mid \forall m, p\ (m \bullet [\![A]\!] \leqslant p \text{ et } m \bullet [\![B]\!] \leqslant p \Rightarrow m \bullet x \leqslant p)\}} & & &
\end{array}$$

Théorème 4.5.4 *La sémantique des monoïdes préordonnés est complète pour* LLI*, et possède la propriété des modèles finis.*

Démonstration. Il suffit de montrer la propriété des modèles finis, car elle implique la complétude. Soit A une formule invalide de LLI. Alors, d'après le théorème 4.4.11, il existe une quantale finie $(\mathcal{Q}, \bullet \leqslant)$ et une interprétation $\sigma : \mathsf{Var} \to \mathcal{Q}$ telle que $\mathsf{e} \not\leqslant [\![A]\!]$. On peut tout aussi bien voir $\mathcal{Q}$ comme un monoïde ordonné. Soit $\sigma' \triangleq \sigma$ une valuation dans le monoïde ordonné $\mathcal{Q}$ et $[\![\cdot]\!]'$ l'interprétation correspondante. D'après le corollaire 2.5.8, $i : \mathcal{Q} \rightleftharpoons \mathcal{Q}^\circ$ défini par $i(x) \triangleq {\downarrow}x$ est un isomorphisme de quantales et si V est une variable, on a $[\![V]\!]' = i([\![V]\!])$. Alors par isomorphie, il est clair que $[\![F]\!]' = i([\![F]\!])$ pour toute formule F. En particulier, comme $\mathsf{e} \not\leqslant [\![A]\!]$, on obtient ${\downarrow}\mathsf{e} \not\subseteq [\![A]\!]'$ ou encore, $\mathsf{e} \notin [\![A]\!]'$. Nous avons bien un contre-modèle fini de A. □

4.6 Sémantique à base de réseaux de Petri

Les travaux de Winskel et Engberg [Eng 90, Eng 97] ont montré qu'il existe des modèles de LLI basés sur des réseaux de Petri. Malheureusement, les modèles qu'ils décrivent sont incomplets. Ainsi, dans leur interprétation, les opérateurs additifs $\oplus$ et & distribuent l'un sur l'autre.

Ces travaux furent le point de départ de toute notre analyse des relations entre clôtures, quantales et monoïdes ordonnés (ou réseaux de Petri.) Nous avons cherché à comprendre pourquoi ces modèles étaient incomplets. Nous établirons que cette incomplétude n'est pas inhérente au choix des réseaux de Petri mais à celui, trop restrictif, de la plus petite clôture (en fait prétopologie.)

Pour cela, nous montrerons que l'on peut voir un réseau de Petri comme une représentation concrète mais générique d'un monoïde ordonné. En choisissant la plus grande clôture, comme dans la section précédente, on retrouve évidement la complétude ainsi la propriété des modèles finis. Nous indiquons que ces résultats s'étendent au cas non-commutatif de LLI à la condition d'utiliser une version non-commutative des réseaux de Petri.[56]

Ici, nous n'aborderons pas l'utilisation de LLI comme logique de spécification pour les réseaux de Petri. À l'avenir, nous envisageons d'étudier les conséquences des nouveaux résultats de complétude par rapport aux problèmes de spécification des réseaux.

4.6.1 Les réseaux de Petri sont des monoïdes préordonnés

Nous rappelons les bases de la théorie des réseaux de Petri. Ils permettent de modéliser des processus concurrents [Rei 85]. Un réseau de Petri est constitué d'un ensemble de **places** sur lesquelles se trouvent des **jetons**. La description de la répartition des jetons sur les places est faite par une **distribution** qui peut-être vue comme un multi-ensemble de places. Les jetons sont des ressources et la façon dont ces jetons sont produits ou consommés est décrite par un système de **transitions**.

Définition 4.6.1 (Réseau de Petri) *Soit $\mathcal{P}$ un ensemble formé d'éléments appelés **places** et $\mathcal{T}$ un ensemble dont les éléments sont appelés **transitions**. On note $\mathbb{M}_{\mathrm{f}}(\mathcal{P})$ l'ensemble des **distributions** qui sont des multi-ensembles finis d'éléments de $\mathcal{P}$. Un **réseau de Petri** $\mathcal{R}$ est la donnée de $\mathcal{R} = (\mathcal{P}, \mathcal{T}, {}^\bullet(\cdot), (\cdot)^\bullet)$ où ${}^\bullet(\cdot)$ et $(\cdot)^\bullet$ sont deux fonctions $\mathcal{T} \to \mathbb{M}_{\mathrm{f}}(\mathcal{P})$.*

On rappelle que l'on note par $+$ la superposition (monoïdale) des distributions et par 0 la distribution vide. De façon à ne pas alourdir les notations, nous noterons par $\mathcal{M}$ le monoïde (libre) des distributions $\mathbb{M}_{\mathrm{f}}(\mathcal{P})$. Une **transition** t correspond à la clôture contextuelle d'une transformation élémentaire entre distributions représentée par la paire $({}^\bullet t, t^\bullet)$. Ainsi nous définissons une relation de transition $[t\rangle \subseteq \mathcal{M} \times \mathcal{M}$ par :

$$m\ [t\rangle\ m' \quad \text{ssi} \quad \exists c \in \mathcal{M},\ m = c + {}^\bullet t \text{ and } m' = c + t^\bullet \tag{4.4}$$

De manière intuitive, cela veut dire que l'on peut obtenir la distribution m' à partir de la distribution m en effectuant une transition t. Cette opération consiste à enlever ${}^\bullet t$ jetons de m obtenant ainsi c puis à rajouter $t^\bullet$ jetons à c arrivant ainsi à m'. Afin de distinguer les places (éléments de $\mathcal{P}$) et les distributions (multi-ensembles sur $\mathcal{P}$), on dénotera par $\underline{\mathsf{a}}$ la distribution qui correspond à un unique jeton posé sur la place a.

Prenons par exemple le cas du réseau de Petri en figure 4.3. Dans ce réseau, nous avons par exemple la transition t_3 pour laquelle nous avons la relation $m + \underline{\mathsf{b}}\ [t_3\rangle\ m + 2\underline{\mathsf{a}}$ pour n'importe quel contexte m. La transition $3\underline{\mathsf{a}}\ [t_1\rangle\ \underline{\mathsf{a}} + \underline{\mathsf{b}}$ est possible également par contre aucune transition $\underline{\mathsf{a}}\ [t_i\rangle\ 3\underline{\mathsf{a}} (i = 1, \ldots, 4)$ n'est possible.

Nous introduisons maintenant la clôture réflexive transitive de l'ensemble des transitions possibles, c'est-à-dire

$$\rightsquigarrow \triangleq \left(\bigcup \{[t\rangle |\ t \in \mathcal{T}\}\right)^\star \qquad m \rightsquigarrow m' \text{ ssi } m\ [t_1\rangle \cdots [t_k\rangle\ m' \text{ pour une suite } t_1, \ldots, t_k \tag{4.5}$$

En revenant à l'exemple figure 4.3, on peut vérifier que l'on a $\underline{\mathsf{a}} \rightsquigarrow 3\underline{\mathsf{a}}$ car $\underline{\mathsf{a}}\ [t_1\rangle\ \underline{\mathsf{a}} + \underline{\mathsf{b}}\ [t_3\rangle\ 3\underline{\mathsf{a}}$. On constate par contre que les transitions préservent la parité des jetons en place a. Ainsi la

[56] Ceci consiste à définir les contextes non plus comme des multi-ensembles mais comme des mots qui sont les éléments du monoïde libre non-commutatif.

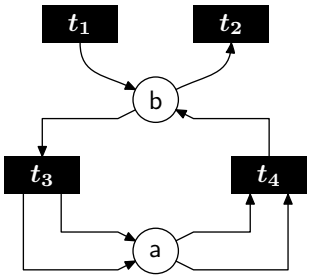


FIG. 4.3 – Exemple de réseau de Petri.

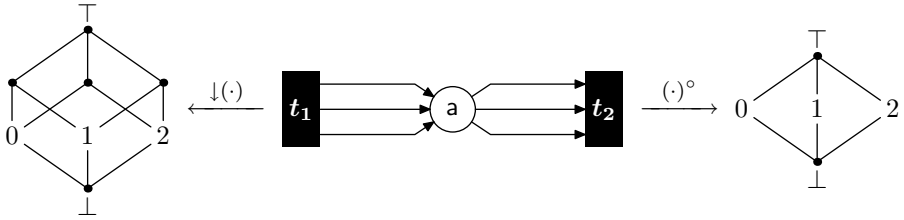


FIG. 4.4 – Comparaison des deux sémantiques.

relation $\underline{a} \rightsquigarrow 2\underline{a}$ n'est pas vraie. La relation binaire $\rightsquigarrow$ sur $\mathcal{M}$ est un préordre (par définition) et est contextuellement close.

Proposition 4.6.1 *Pour toutes distributions m_0, m, m' de $\mathcal{M}$, si $m \rightsquigarrow m'$ alors $m_0 + m \rightsquigarrow m_0 + m'$.*

Démonstration. La clôture réflexive et transitive d'une relation contextuellement close est contextuellement close. □

Corollaire 4.6.2 *Si $\mathcal{R} = (\mathcal{P}, \mathcal{T}, {}^\bullet(\cdot), (\cdot)^\bullet)$ est un réseau de Petri alors $(\mathbb{M}_{\mathrm{f}}(\mathcal{P}), +, \rightsquigarrow)$ est un monoïde préordonné.*

Nous dénoterons désormais par $\mathbb{M}_{\mathcal{R}}$ le monoïde préordonné $(\mathbb{M}_{\mathrm{f}}(\mathcal{P}), +, \rightsquigarrow)$. On pourra observer que $\mathbb{M}_{\mathcal{R}}$ est un vrai monoïde, pas seulement un quasi-monoïde, voir section 2.3.2 au chapitre 2. La transformation $\mathcal{R} \mapsto \mathbb{M}_{\mathcal{R}}$ nous permet de définir une sémantique pour LLI en nous basant sur la sémantique de monoïdes préordonnés.

4.6.2 Interprétation de LLI

Nous considérons un réseau de Petri fixé $\mathcal{R}$ et son monoïde préordonné associé $\mathbb{M}_{\mathcal{R}} = (\mathbb{M}_{\mathrm{f}}(\mathcal{P}), +, \rightsquigarrow)$. D'après les résultats de la section 4.5, en choisissant une prétopologie $(\cdot)^\star$ comme par exemple $\downarrow(\cdot)$ ou $(\cdot)^\circ$, on obtient une interprétation fidèle des séquents et formules de LLI, dans la mesure où $(\mathbb{M}_{\mathcal{R}}{}^\star, (\cdot + \cdot)^\star, \subseteq)$ est une quantale. Par exemple on peut considérer les deux complétions possibles du réseau de Petri en figure 4.4. En notant 0, 1 et 2 les ensembles de distributions respectifs $3\mathbb{N}.\underline{a}$, $(1 + 3\mathbb{N}).\underline{a}$ et $(2 + 3\mathbb{N}).\underline{a}$, et par $\bot$ et $\top$ respectivement $\emptyset.\underline{a}$ et $\mathbb{N}.\underline{a}$, on obtient les complétions dont les structures de treillis sont indiquées sur la figure.

Dans la suite, nous comparons les deux prétopologies « extrémales » du point de vue de la sémantique induite pour les formules de la logique LLI.

La sémantique initiale

Dans l'article [Eng 97] sur la sémantique de LLI à base de réseaux de Petri, les auteurs ont en fait choisi $\downarrow(\cdot)$ pour prétopologie et développe la complétion du monoïde préordonné $\mathbb{M}_{\mathcal{R}}$ pour obtenir l'interprétation suivante. Soit $\sigma : \mathsf{Var} \to \mathbb{M}_{\mathrm{f}}(\mathcal{P})$ une interprétation des variables :

La sémantique de Winskel-Engberg

$$\begin{array}{rcl@{\qquad}rcl} [\![1]\!] & \triangleq & \{x \mid x \rightsquigarrow 0\} & [\![\mathsf{T}]\!] & \triangleq & \mathcal{M}_f(\mathcal{P}) \\ [\![X]\!] & \triangleq & \{x \mid x \rightsquigarrow \sigma(X)\} & [\![A \multimap B]\!] & \triangleq & [\![A]\!] \multimap\!\bullet\, [\![B]\!] \\ [\![A \oplus B]\!] & \triangleq & [\![A]\!] \cup [\![B]\!] & [\![A \,\&\, B]\!] & \triangleq & [\![A]\!] \cap [\![B]\!] \end{array}$$

$$[\![A \otimes B]\!] \triangleq \{x \mid \exists a \in [\![A]\!], \exists b \in [\![B]\!],\ x \rightsquigarrow a + b\}$$

Malheureusement cette sémantique est incomplète, parce que distributive. Ceci est un corollaire de la proposition 4.5.2 :

Proposition 4.6.3 *Dans l'interprétation de* LLI *précédemment définie, nous avons l'identité* $[\![A \,\&\, (B \oplus C)]\!] = [\![(A \,\&\, B) \oplus (A \,\&\, C)]\!]$.

D'après [Eng 97], la complétude ne peut-être retrouvée qu'à condition d'affaiblir la logique en supprimant un des opérateurs $\&$, $\oplus$. D'une certaine manière, ces résultats de complétude partiels induisent en erreur. On pourrait penser que l'incomplétude provient de la nature des réseaux de Petri, alors qu'en fait, nous allons montrer qu'elle provient du choix de la plus petite prétopologie $\downarrow(\cdot)$ plutôt que de la plus grande $(\cdot)^\circ$.

La nouvelle sémantique

Nous considérons donc la plus grande prétopologie $(\cdot)^\circ$ sur le monoïde préordonné $\mathbb{M}_{\mathcal{R}}$. On se donne une valuation $\sigma : \mathsf{Var} \to \mathbb{M}_{\mathrm{f}}(\mathcal{P})$ pour interpréter les variables et on obtient la sémantique suivante :

Nouvelle sémantique à base de réseaux de Petri

$$\begin{array}{rcl@{\qquad}rcl} [\![X]\!] & \triangleq & \{x \mid x \rightsquigarrow \sigma(X)\} & [\![A \multimap B]\!] & \triangleq & [\![A]\!] \multimap\!\bullet\, [\![B]\!] \\ [\![\mathsf{T}]\!] & \triangleq & \mathcal{M}_f(\mathcal{P}) & [\![A \,\&\, B]\!] & \triangleq & [\![A]\!] \cap [\![B]\!] \\ [\![1]\!] & \triangleq & \{x \mid x \rightsquigarrow 0\} & [\![\mathsf{F}]\!] & \triangleq & \{x \mid \forall m, p\ m + x \rightsquigarrow p\} \end{array}$$

$$\begin{array}{rcl} [\![A \otimes B]\!] & \triangleq & \{x \mid \forall m, p\ (m + [\![A]\!] + [\![B]\!] \rightsquigarrow p \Rightarrow m + x \rightsquigarrow p)\} \\ [\![A \oplus B]\!] & \triangleq & \{x \mid \forall m, p\ (m + [\![A]\!] \rightsquigarrow p \text{ et } m + [\![B]\!] \rightsquigarrow p \Rightarrow m + x \rightsquigarrow p)\} \end{array}$$

Cette nouvelle interprétation est bien sûr valide comme cas particulier de la sémantique des phases par exemple. Nous montrons sa complétude et la propriété des modèles finis dans la section 4.6.4.

Proposition 4.6.4 (Validité) *Soit A une formule valide de* LLI. *Pour les deux interprétations $[\![\cdot]\!]$ définies respectivement par $\downarrow(\cdot)$ et $(\cdot)^\circ$, on a la relation $0 \in [\![A]\!]$.*

Autrement dit, lorsque A est une formule valide, dans la mesure où $[\![A]\!] = \downarrow[\![A]\!]$ est une section initiale, n'importe quelle distribution réductible à 0 doit faire partie de $[\![A]\!]$.

Comparaison des deux sémantiques

On constate tout d'abord que l'interprétation de la constante F et des opérateurs $\otimes$ et $\oplus$ est différente par rapport à la sémantique de Winskel. L'interprétation des variables et des opérateurs

1, $\top$, $\multimap$ et $\&$ est par contre identique. Ceci ne veut pas dire qu'une formule de type $A \& B$ possède la même interprétation dans les deux sémantiques. Il se pourrait très bien que les interprétations $[\![A]\!]$ et $[\![B]\!]$ soient déjà différentes dans les deux sémantiques.

Revenons au réseau de Petri de la figure 4.4. On base les deux interprétations sur la valuation $\sigma(A) \triangleq 0$, $\sigma(B) \triangleq \underline{\mathtt{a}}$ et $\sigma(B) \triangleq 2\underline{\mathtt{a}}$. On obtient la même interprétation des variables dans les deux sémantiques, à savoir $[\![A]\!] = 3\mathbb{N}.\underline{\mathtt{a}}$, $[\![B]\!] = (1 + 3\mathbb{N}).\underline{\mathtt{a}}$ et $[\![C]\!] = (2 + 3\mathbb{N}).\underline{\mathtt{a}}$. Par contre si on considère les interprétations de la formule $B \oplus C$, on constate qu'elles diffèrent : dans la sémantique de Winskel, on obtient $[\![A \oplus B]\!] = [\![B]\!] \cup [\![C]\!] = (\{1, 2\}) + 3\mathbb{N}).\underline{\mathtt{a}}$ alors que dans la nouvelle sémantique, on obtient $[\![B \oplus C]\!] = ([\![B]\!] \cup [\![C]\!])^\circ = \mathbb{N}.\underline{\mathtt{a}}$. En effet, en remarquant que le nombre de jetons en place $\mathtt{a}$ modulo 3 est préservé par les transitions t_1 et t_2, on constate que pour toute distribution m il n'existe pas de distribution p telle que les transitions $m + 1.\underline{\mathtt{a}} \rightsquigarrow p$ et $m + 2.\underline{\mathtt{a}} \rightsquigarrow p$ soient simultanément possibles. Ainsi, la condition $m + (\{1, 2\} + 3\mathbb{N}).\underline{\mathtt{a}} \rightsquigarrow p$ est toujours fausse et par conséquent on obtient $[\![B \oplus C]\!] = \mathbb{N}.\underline{\mathtt{a}}$.

Ainsi l'opérateur $\oplus$, qui peut se voir comme une disjonction dans la sémantique initiale de Winskel, ne peut plus être considéré comme tel. Ce point sera développé dans des travaux futurs.

4.6.3 L'universalité des réseaux de Petri

Dans cette section, nous montrons que la transformation $\mathcal{R} \mapsto \mathbb{M}_\mathcal{R}$ qui associe son monoïde préordonné à un réseau de Petri est surjective, à équivalence près ce qui a comme conséquence la complétude de la nouvelle sémantique à base de réseaux de Petri, par application de la proposition 4.5.3.

On fixe un monoïde préordonné $(\mathcal{M}, \bullet, \leqslant)$. Nous allons construire un réseau de Petri $\mathcal{R}$, fini si $\mathcal{M}$ l'est, tel que $\mathcal{M} \simeq \mathbb{M}_\mathcal{R}$. Les places de $\mathcal{R}$ sont les éléments de $\mathcal{M}$. $\mathcal{R}$ est donc de la forme $(\mathcal{M}, \mathcal{T}, {}^\bullet(\cdot), (\cdot)^\bullet)$. L'ensemble des transitions est constitué de 5 types de transitions, $\mathcal{T} \triangleq T_1 \cup \cdots \cup T_5$ où les T_i sont définis par le tableau :

$t_1 \in T_1$	$\underline{a}\ [t_1\rangle\ \underline{b}$	pour $a \leqslant b$
$t_2 \in T_2$	$\underline{a}\ [t_2\rangle\ \underline{b} + \underline{c}$	pour $a = b \bullet c$
$t_3 \in T_3$	$\underline{b} + \underline{c}\ [t_3\rangle\ \underline{a}$	pour $a = b \bullet c$

$\{t_4\} = T_4$	$0\ [t_4\rangle\ \underline{\mathtt{e}}$
$\{t_5\} = T_5$	$\underline{\mathtt{e}}\ [t_5\rangle\ 0$

Ainsi par exemple, il y a exactement une transition de type T_1 pour chaque paire (a, b) d'éléments de $\mathcal{M}$ tels que $a \leqslant b$. Nous avons donc un réseau de Petri avec 5 types de transitions. Il est clair que $\mathcal{R}$ est fini avec un nombre fini de transitions si et seulement si $\mathcal{M}$ est fini.

Proposition 4.6.5 *Pour toute suite $a_1, \ldots, a_n$ d'éléments de $\mathcal{M}$, l'équivalence $\underline{a_1} + \cdots + \underline{a_n} \simeq \underline{a_1 \bullet \cdots \bullet a_n}$ est vérifiée.*

Démonstration. La preuve se fait par induction sur n. Pour le cas $n = 0$, une transition de type T_4 (resp. T_5) donne $0 \rightsquigarrow \underline{\mathtt{e}}$ (resp. $\underline{\mathtt{e}} \rightsquigarrow 0$.) Le cas $n+1$ est traité par des transitions de types T_2 et T_3. □

Proposition 4.6.6 *La relation $\underline{a_1} + \cdots + \underline{a_n} \rightsquigarrow \underline{b_1} + \cdots + \underline{b_m}$ dans $\mathcal{R}$ est vérifiée si et seulement si la relation $a_1 \bullet \cdots \bullet a_n \leqslant b_1 \bullet \cdots \bullet b_m$ dans $\mathcal{M}$ est vérifée.*

Démonstration. Pour le cas **seulement si**, on pourra observer que les transitions de $T_1 \cup \cdots \cup T_5$ sont croissantes par rapport à la relation $\leqslant$ et donc la clôture contextuelle, réflexive, et transitive $\rightsquigarrow$ de l'union l'est aussi. Pour le cas **si**, il suffit d'utiliser une transition de type T_1 et d'appliquer la proposition 4.6.5. □

Lemme 4.6.7 *La fonction $i : \mathcal{M} \rightleftharpoons \mathbb{M}_{\mathrm{f}}(\mathcal{M})$ définie par $i(a) \triangleq \underline{a}$ est une équivalence entre les monoïdes préordonnés $(\mathcal{M}, \bullet, \leqslant)$ et $\mathbb{M}_{\mathcal{R}} = (\mathbb{M}_{\mathrm{f}}(\mathcal{M}), +, \rightsquigarrow)$.*

Démonstration. D'après la proposition 4.6.6, dans le cas $n = m = 1$, i est un plongement d'ensembles préordonnés. D'après 4.6.5, nous avons l'équivalence $i(a_1 \bullet \cdots \bullet a_n) \simeq \underline{a_1} + \cdots + \underline{a_n}$. Pour $n = 0$, on obtient $i(\mathsf{e}) = 0$. Pour $n = 2$, on obtient $i(x \bullet y) = i(x) + i(y)$. Enfin, tout élément de $\mathbb{M}_{\mathrm{f}}(\mathcal{M})$ est de la forme $\underline{a_1} + \cdots + \underline{a_n}$ donc i est surjective à équivalence près et i est bien une équivalence de monoïdes préordonnés. □

Théorème 4.6.8 *Soit $(\mathcal{M}, \bullet, \leqslant)$ un monoïde préordonné et $\mathcal{R}$ le réseau de Petri précédemment défini par les transitions $\mathcal{T} \triangleq T_1 \cup \cdots \cup T_5$. Alors $\mathbb{M}_{\mathcal{R}}$ et $\mathcal{M}$ sont deux monoïdes préordonnés équivalents. D'autre part, $\mathcal{R}$ est un réseau fini si l'ensemble $\mathcal{M}$ l'est.*

Nous faisons remarquer que [Eng 97] présente une autre construction qui ne préserve pas le caractère fini. Par cette construction on obtient toujours des réseaux de Petri infinis. Il est vrai aussi que n'ayant pas la complétude, obtenir des modèles finis devient secondaire.

4.6.4 Complétude et propriété des modèles finis

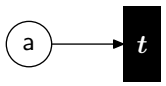
Théorème 4.6.9 *La nouvelle sémantique à base de réseau de Petri est complète et possède la propriété des modèles finis.*

Démonstration. L'adéquation a déjà été montrée ; il suffit de montrer que toute formule invalide admet un contre-modèle fini. Soit donc F une formule invalide, d'après le théorème 4.5.4, il existe un monoïde préordonné fini $(\mathcal{M}, \bullet, \leqslant)$ et une interprétation $\sigma : \mathsf{Var} \to \mathcal{M}$ pour laquelle $\mathsf{e} \notin [\![F]\!]$. D'après le lemme 4.6.8, nous considérons le réseau de Petri fini pour lequel $\mathbb{M}_{\mathcal{R}}$ et $\mathcal{M}$ sont équivalents, par $i(x) \triangleq \underline{x}$. Soit $\sigma' : \mathsf{Var} \to \mathbb{M}_{\mathrm{f}}(\mathcal{M})$ l'interprétation des variables dans $\mathcal{R}$ définie par $\sigma'(V) \triangleq i\big(\sigma(V)\big)$. On dénote par $[\![\cdot]\!]'$ la nouvelle interprétation correspondante. D'après la proposition 4.5.3, on a alors $[\![F]\!]' = {\downarrow} i([\![F]\!])$. Supposons que $0 \in [\![F]\!]'$ par l'absurde. Alors il existe $k \in [\![F]\!]$ tel que $0 \rightsquigarrow i(k) = \underline{k}$. Or $\underline{\mathsf{e}} \simeq 0$ donc $\underline{\mathsf{e}} \rightsquigarrow \underline{k}$. Par conséquent, $\mathsf{e} \leqslant k$ et comme $[\![F]\!]$ est ${\downarrow}(\cdot)$-close, il vient $\mathsf{e} \in [\![F]\!]$, ce qui est en contradiction avec l'hypothèse de départ. Donc $0 \notin [\![F]\!]'$ et nous avons un contre-modèle de F sous la forme d'un réseau de Petri fini. □

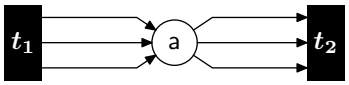
Nous avons donc montré que la sémantique des réseaux de Petri peut être modifiée en changeant de prétopologie. De cette manière elle devient complète et hérite également de la propriété des modèles finis. Ces deux résultats complètent et renforcent l'étude initiale de [Eng 97] qui porte sur les relations entre LLI et les réseaux de Petri. En fait, à la lumière de notre nouvelle interprétation, les réseaux de Petri deviennent un modèle convaincant et utile de toute la logique intuitionniste linéaire propositionnelle, alors qu'ils ne l'étaient que pour certains fragments dans l'interprétation initiale.

Pour compléter notre étude, il nous faut aussi aborder les conséquences de ces résultats lorsque l'on considère LLI comme une logique pour spécifier et raisonner sur les réseaux de Petri. C'est aussi l'un des thèmes abordés par les premières études portant sur les relations entre LLI et les réseaux de Petri [Bro 89, Mes 91, Lil 92, Eng 97]. Certains fragments de la logique LLI permettent de prouver des assertions positives élémentaires et capturent également des propriétés négatives intéressantes. Des travaux futures seront entrepris pour tirer les conséquences de la nouvelle interprétation dans ce contexte.

4.6.5 Exemple de contre-modèles



Nous développons maintenant un contre-modèle simple pour le séquent $X \vdash X \bullet X$ à base du réseau de Petri ci-contre. Il possède une seule place notée a et une seule transition t. On définit $\sigma(X) \triangleq \underline{\mathsf{a}}$ et ainsi, pour les deux sémantiques, on obtient $[\![X]\!] = \{m \mid m \rightsquigarrow \underline{\mathsf{a}}\} = (\mathbb{N}+1).\underline{\mathsf{a}}$. On a donc $[\![X]\!]+[\![X]\!] = (\mathbb{N}+2).\underline{\mathsf{a}} = \downarrow 2\underline{\mathsf{a}}$. Ainsi, pour n'importe quelle interprétation $(\cdot)^\star$ parmi $\downarrow(\cdot)$ et $(\cdot)^\circ$ on obtient $[\![X \otimes X]\!] = ([\![X]\!] + [\![X]\!])^\star = (\downarrow 2\underline{\mathsf{a}})^\star = \downarrow 2\underline{\mathsf{a}}$ car les sections initiales sont $(\cdot)^\star$-closes. On en déduit que $[\![X]\!] \not\subseteq [\![X \otimes X]\!]$ ce qui nous donne un contre-modèle du séquent $X \vdash X \otimes X$.



Nous reprenons le réseau de Petri de la figure 4.4 redonné ci-contre. Nous rappelons que nous avions les identités $[\![A]\!] = 3\mathbb{N}.\underline{\mathsf{a}}$, $[\![B]\!] = (1+3\mathbb{N}).\underline{\mathsf{a}}$ et $[\![C]\!] = (2+3\mathbb{N}).\underline{\mathsf{a}}$. Nous en déduisions que dans le cas de la nouvelle sémantique, on obtient $[\![B \oplus C]\!] = \mathbb{N}.\underline{\mathsf{a}}$. Et ainsi il s'ensuit $[\![A \,\&\, (B \oplus C)]\!] = [\![A]\!] \cap [\![B \oplus C]\!] = 3\mathbb{N}.\underline{\mathsf{a}}$. Par contre, on obtient $[\![A \,\&\, B]\!] = [\![A]\!] \cap [\![B]\!] = \emptyset$ et de manière analogue, $[\![A \,\&\, C]\!] = \emptyset$. Ainsi, $[\![(A \,\&\, B) \oplus (A \,\&\, C)]\!] = ([\![A \,\&\, B]\!] \cup [\![A \,\&\, C]\!])^\circ = \emptyset^\circ = \emptyset$. Par conséquent, nous avons là un contre-modèle à la distributivité de & sur $\oplus$ sous la forme d'un réseau de Petri.

Le problème de la génération automatique de contre-modèles sous la forme de réseaux de Petri par exemple est une voie de recherche à venir. On notera cependant que contrairement à LI, peu de règles de LLI sont inversibles et de plus, il y a en général beaucoup plus d'instances de règles applicables que pour LI. Ceci implique que les arbres de réfutation sont bien plus larges et par voie de conséquence, les contre-modèles d'autant plus gros. Le calcul des séquents n'est sans doute pas le plus adapté à la construction de contre-modèle et il faudra sans doutes se tourner vers les réseaux de preuves [Gir 95a] ou bien vers d'autres systèmes qui gèrent les ressources de manière plus efficace à la manière de Lolli [Cer 00].

Conclusion et perspectives

Après avoir rappelé la syntaxe et les sémantiques algébrique et des phases de LLI, nous introduisons une nouvelle sémantique de LLI basée sur la notion de ressource. Cette dernière se prête très bien à la construction de modèles syntaxiques. Ces modèles peuvent être rendus finis par passage au quotient. Ce quotient est construit très naturellement à partir d'un arbre de réfutation. Nous obtenons une preuve très modulaire de la propriété des modèles finis, propriété qui se transmet aux autres sémantiques de manière tout-à-fait standard.

Cette dernière preuve met à nouveau l'accent sur la notion de réfutation comme un lien entre une approche syntaxique de la décidabilité à travers l'analycité d'un calcul (complet) et une approche sémantique à travers la propriété des modèles finis. Elle permet de définir des bases pour la construction automatique de contre-modèles même si les problèmes de complexité et de taille subsistent dans la pratique.

Nous nous intéressons ensuite à l'analyse de modèles à base de réseaux de Petri. Nous proposons une sémantique fondée sur la notion de monoïde ordonné, qui est une version abstraite de la sémantique des réseaux de Petri. Nous expliquons pourquoi la sémantique initiale de Winskel [Eng 97] ne pouvait être qu'incomplète. Il apparaît que le choix de la plus grande clôture au lieu de la plus petite permet de corriger ce défaut. Nous démontrons la complétude de cette nouvelle sémantique ainsi que la propriété des modèles finis.

La construction automatique de contre-modèles est l'un des principaux objectifs à venir. Cependant, le choix de la forme des contre-modèles (quantales, espaces de ressources, réseaux de Petri...) ou de la méthode de recherche de preuves (construction de réseaux de preuve, calcul des

séquents...) n'est pas encore clair. L'une des principales difficultés est le problème de la taille. Ce problème provient à la fois de la taille de l'espace de recherche de preuve (où le calcul des séquents n'est sans doutes pas le plus économe) et de la taille des modèles qui ne stockent peut-être pas l'information de la manière la plus compacte possible. Des modèles de Kripke étendus pourraient peut-être constituer une solution. Il est fort possible que le choix d'une méthode de recherche de preuve ne soit pas indépendant du choix de la forme des contre-modèles.

L'adaptation des techniques développées pour LLI à la logique BI [O'H 99] qui est une logique mixte entre logique LI et LLI est un autre point à développer. Cependant, il n'y a, à notre connaissance, pas encore de système analytique connu pour la recherche de preuve dans BI.[57] Ce travail présente donc sans aucun doute des difficultés importantes. D'autre part, les sémantiques algébriques de cette logique semblent aussi relativement complexes.

[57] Pym et O'Hearn sont parvenus à une preuve d'élimination des coupures mais dans la mesure où les séquents ont une structure de contextes emboités non triviale, on ne peut en déduire la décidabilité car il faut encore prouver l'analycité du système.

Conclusion

Dans cette thèse nous avons montré comment la transformation de réfutations en contre-modèles peut être utilisée à la fois dans le cadre d'une étude théorique, comme la preuve de la propriété des modèles finis, et dans le cadre d'une application plus pratique comme la construction effective de contre-modèles. Nous avons aussi utilisé avantageusement les notions de ressources et de partage de ressources pour à la fois modéliser des systèmes logiques et fournir une implantation efficace de systèmes logiques fondés sur le calcul des séquents.

Le système STRIP est l'aboutissement d'une étude sur le partage des ressources dans la recherche de preuve combinée aux techniques de construction des contre-modèles à partir d'une réfutation, et ce en logique intuitionniste.

Nous partons du système LJT dont nous rappelons l'analycité. Nous montrons ensuite la co-validité du système par rapport à la sémantique des arbres de Kripke, ce qui nous donne à la fois la complétude de LJT et celle de la sémantique de Kripke par rapport à LI ainsi qu'un algorithme de construction automatique de preuves ou de contre-modèles.

Puis nous raffinons LJT en nous concentrant sur le partage des ressources, les formules logiques dans notre cas. Deux formes distinctes de partage sont mises en œuvre. Tout d'abord, au niveau logique, on montre qu'il est possible de se restreindre au cas où les séquents partagent une sous-formule entre une hypothèse et la conclusion ce qui élimine une forme de duplication par rapport au système LJT. On obtient un système que l'on appelle SLJ.

Puis, en représentant les séquents par des forêts où les chemins des racines aux feuilles codent des formules — les arêtes représentant des implications logiques, — il est possible de mettre en œuvre une deuxième forme de partage, celui des formules qui se trouvent sur les nœuds internes de la forêt. On élimine ainsi la deuxième forme de duplication présente dans LJT. On obtient alors un système appelé STRIP sans aucune duplication et cette propriété de linéarité autorise une implantation efficace, tant du point de vue de la gestion de la mémoire (pas d'allocation dynamique) que de la gestion du temps (l'application d'une règle se fait en temps constant.)

Au niveau des perspectives, plusieurs voies sont possibles. Tout d'abord, il s'avère à l'usage que STRIP ne fournit pas toujours les contre-modèles les plus petits possibles : il est clair que certaines branches de la recherche de preuve sont dupliquées. Par contre, il ne paraît pas évident a priori de supprimer ces duplications de manière automatique mais certaines voies sont actuellement explorées. Il peut aussi être intéressant d'étudier la généralisation des techniques de partage à la logique intuitionniste du premier ordre ou à certaines formes de logiques intuitionnistes modales, comme la logique Lax [Ave 96, Fai 97]. Il n'est pas non plus évident que ces généralisations soient possibles. Par contre, il semble possible d'appliquer des techniques de partages multiples à certaines logiques intermédiaires comme la logique de Gödel-Dummet. Enfin, nous envisageons l'implantation du système STRIP au sein d'un outil démonstration automatique de manière à pouvoir extraire automatiquement un programme certifié de la preuve de complétude sémantique.

La notion de plus grande prétopologie, un cas particulier de clôture, est à l'origine d'une

nouvelle construction de quantale comme complétion d'un monoïde ordonné qui généralise la complétion de MacNeille. Cette construction préserve au maximum la structure de quantale (partielle) qui préexiste et permet d'obtenir des sémantiques complètes pour LLI à base de monoïdes ordonnés. Les réseaux de Petri peuvent être vus comme des représentations particulières de monoïdes ordonnés, cette approche n'étant pas limitative : tout monoïde ordonné (fini) peut-être représenté par un réseau de Petri (fini). Nous en déduisons un résultat de complétude pour une nouvelle sémantique de LLI à base de réseaux de Petri qui donne un autre éclairage sur leurs rapports avec LLI.

Nous définissons également une nouvelle sémantique fondée sur la notion de ressources. Cette sémantique généralise les sémantiques précitées. De plus, elle se trouve être très adaptée à la preuve de la propriété des modèles finis. Nous présentons une transformation simple d'un arbre de réfutation dans $\mathsf{Gil}_{\mathsf{sc}}$ en contre-modèle fini sous la forme d'un espace de ressources : il s'agit du quotient fini d'un espace de ressources syntaxique, l'arbre de réfutation nous indiquant à quelles identifications il faut procéder. De ce résultat, combiné à l'analycité du système $\mathsf{Gil}_{\mathsf{sc}}$ découle la propriété des modèles finis pour la sémantique des ressources et par voie de conséquence pour toutes les autres sémantiques.

Au niveau des perspectives, il reste à explorer le problème de la construction automatique de contre-modèles. Le choix de représentation des contre-modèles (monoïdes ordonnés, réseaux de Petri, espace de ressources...) paraît crucial car une réfutation dans $\mathsf{Gil}_{\mathsf{sc}}$ est un objet très encombrant. Encore une fois le problème du partage des ressources risque de se poser. Il est aussi essentiel d'étudier les propriétés de notre nouvelle sémantique par rapport aux problèmes de spécifications et de preuves de propriété des réseaux de Petri par exemple. Enfin, l'extension de la sémantique des ressources à des logiques mélangeant des aspects intuitionnistes et intuitionnistes linéaires, comme par exemple la logique BI et l'adaptation des résultats obtenus pour LLI est une autre voie pour des recherches à venir.

Bibliographie

[Abr 93] Samson Abramsky. Computational Interpretations of Linear Logic. *Theoretical Computer Science*, 111(1–2) :3–58, 1993.

[Abr 94] Samson Abramsky et Radha Jagadeesan. Games and Full Completeness for Multiplicative Linear Logic. *Journal of Symbolic Logic*, 59(2) :543–574, 1994.

[Abr 99] Samson Abramsky et Paul A. Mellies. Concurrent Games and Full Completeness. *14^{th} Symposium on Logic in Computer Science*, pages 431–442, Trento, Italy, 1999.

[All 93] Gerard Allwein et J. Michael Dunn. Kripke Models for Linear Logic. *Journal of Symbolic Logic*, 58(2) :514–545, 1993.

[And 92] Jean-Marc Andreoli. Logic Programming with Focusing Proofs in Linear Logic. *Journal of Logic and Computation*, 2(3) :297–347, 1992.

[Ave 96] Alessandro Avellone et Mauro Ferrari. Almost Duplication-free Tableau Calculi for Propositional Lax Logics. *5^{th} International Workshop TABLEAUX '96*, volume 1071, série *Lecture Notes in Artificial Intelligence*, pages 48–64, Terrasini, Palermo, Italy, 1996. Springer Verlag.

[Ave 99] Alessandro Avellone, Mauro Ferrari et Pierangelo Miglioli. Duplication-Free Tableau Calculi and related cut-free Sequent Calculi for the Interpolable Propositional Intermediate Logics. *Logic Journal of the IGPL*, 7(4) :447–480, 1999.

[Bal 00] Vincent Balat et Didier Galmiche. Labelled Proof Systems for Intuitionistic Provability. D. Basin et al., éditeur, *Labelled Deduction*, volume 17, série *Applied Logic Series*. Kluwer Academic Publishers, 2000.

[Bar 91] Michael Barr. $\star$-Autonomous Categories and Linear Logic. *Mathematical Structures in Computer Science*, 1 :159–178, 1991.

[Bar 94] Franco Barbenera et Stefano Berardi. A symmetric lambda-calculus for classical program extraction. *International Symposium on Theoretical Aspects of Computer Software*, volume 789, série *Lecture Notes in Computer Science*, pages 494–515, 1994.

[Bee 84] Michael J. Beeson. *Foundations of Constructive Mathematics*. Modern Surveys in Mathematics. Springer-Verlag, 1984.

[Bie 96] Gavin M. Bierman. A note on Full Intuitionistic Linear Logic. *Annals of Pure and Applied Logic*, 79 :281–287, 1996.

[Bir 67] Garrett Birkhoff. *Lattice Theory*. American Mathematical Society, 1967.

[Bla 92] Andreas Blass. A Game Semantics for Linear Logic. *Annals of Pure and Applied Logic*, 56 :183–220, 1992.

[Br 97] Torben Bräuner et Valeria de Paiva. A Formulation of Linear Logic based on Dependency-Relations. *Computer Science Logic, CSL '97*, volume 1414, série *Lecture Notes in Computer Science*, pages 129–148. Springer Verlag, 1997.

[Bro 89] Carolyn Brown. Petri nets as Quantales. ECS LFCS no. 96, University of Edinburgh, 1989.

[Cer 00] Iliano Cervesato, Joshua Hodas et Frank Pfenning. Efficient resource management for linear logic proof search. *Theoretical Computer Science*, 232(1–2) :133–163, 2000.

[Cha 73] Chen-chung Chang et Howard J. Keisler. *Model Theory*, volume 73, série *Studies in logic and the foundations of mathematics*. North Holland, 1973.

[Coq 90] Thierry Coquand. On the Analogy between Propositions and Types. G. Huet, éditeur, *Logical Foundations of Functional Programming*, The UT Year of Programming series : Logical Foundations of Functional Programming, pages 399–418. Addison-Wesley, Reading, MA, 1990.

[Dav 90] Brian A. Davey et Hilary A. Priestley. *Introduction to Lattices and Order*. Cambridge Mathematical Textbooks. Cambridge University Press, 1990.

[Der 79] Nachum Dershowitz et Zohar Manna. Proving Termination with Multiset Orderings. *Communications of the ACM*, 22(8) :465–476, 1979.

[Dev 99] Harish Devarajan, Dominic Hughes, Gordon Plotkin et Vaughan Pratt. Full Completeness of the multiplicative linear logic of Chu spaces. *14^{th} Symposium on Logic in Computer Science*, pages 234–243, Trento, Italy, 1999.

[Dow 93] Gilles Dowek. A Complete Proof Synthesis Method for the Cube of Type Systems. *Journal of Logic and Computation*, 3(3) :287–315, 1993.

[Dyc 92] Roy Dyckhoff. Contraction-free Sequent Calculi for Intuitionistic Logic. *Journal of Symbolic Logic*, 57(3) :795–807, 1992.

[Dyc 99] Roy Dyckhoff. A Deterministic Terminating Sequent Calculus for Gödel-Dummett Logic. *Logic Journal of the IGPL*, 7 :319–326, 1999.

[Eng 90] Uffe H. Engberg et Glynn Winskel. Petri Nets as Models of Linear Logic. *CAAP '90, Coll. on Trees in Algebra and Programming*, volume 431, série *Lecture Notes in Computer Science*, pages 147–161, Copenhagen, Denmark, May 1990. Springer Verlag.

[Eng 97] Uffe H. Engberg et Glynn Winskel. Completeness Results for Linear Logic on Petri nets. *Annals of Pure and Applied Logic*, 86 :101–135, 1997.

[Fai 97] Matt Fairtlough et Michael Mendler. Propositional Lax Logics. *Information and Computation*, 137(1), 1997.

[Gal 92] Jean H. Gallier. Constructive Logics. Part I, II. Rapport, Digital Equipment Corporation, 1992.

[Gal 94] Didier Galmiche et Guy Perrier. Foundations of Proof Search Strategies Design in Linear Logic. *Logic at St Petersburg '94, Symposium on Logical Foundations of Computer Science*, volume 813, série *Lecture Notes in Computer Science*, pages 101–113, St Petersburg, Russia, July 1994.

[Gal 98a] Didier Galmiche et Dominique Larchey-Wendling. Formulae-as-Resources Management for an Intuitionistic Theorem Prover. *5^{th} Workshop on Logic, Language, Information and Computation, WoLLIC '98*, São Paulo, Brazil, Juillet 1998.

[Gal 98b] Didier Galmiche et Bruno Martin. Proof Nets Construction and Automated Deduction in Non-Commutative Linear Logic – extended abstract. *Electronic Notes in Theoretical Computer Science*, 17, 1998.

[Gal 99] Didier Galmiche et Dominique Larchey-Wendling. Structural Sharing and Efficient Proof-Search in Propositional Intuitionistic Logic. *Asian Computing Science Conference, ASIAN'99*, volume 1742, série *Lecture Notes in Computer Science*, pages 101–102, Phuket, Thaïlande, December 1999.

[Gal 00] Didier Galmiche. Connection Methods in Linear Logic and Proof nets Construction. *Theoretical Computer Science*, 232(1–2) :231–272, 2000.

[Gir 87] Jean-Yves Girard. Linear Logic. *Theoretical Computer Science*, 50(1) :1–102, 1987.

[Gir 89] Jean-Yves Girard, Yves Lafont et Paul Taylor. *Proofs and Types*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1989.

[Gir 95a] Jean-Yves Girard. Linear Logic : Its Syntax and Semantics. J.-Y. Girard, Y. Lafont et L. Regnier, éditeurs, *Advances in Linear Logic*, pages 1–42. Cambridge University Press, 1995.

[Gir 95b] Jean-Yves Girard. Proof Nets : the Parallel Syntax for Proof Theory. A. Ursini et P. Agliano, éditeurs, *Logic and Algebra*, New York, 1995. M. Dekker.

[GL 96] Jean Goubault-Larrecq. Implementing Tableaux by Decision Diagrams. Rapport, Institut für Logik, Komplexität und Deduktionssysteme, Universität Karlsruhe, Karlsruhe, Germany, 1996.

[Gou 94] Jean Goubault. Proving with BDDs and control of information. A. Bundy, éditeur, *12^{th} International Conference on Automated Deduction, CADE-12*, volume 814, série *Lecture Notes in Artificial Intelligence*, Nancy, France, June 1994. Springer-Verlag.

[Har 92] James Harland et David J. Pym. On Resolution in Fragments of Classical Linear Logic. *LPAR '92, International Conference on Logic Programming and Automated Reasoning*, volume 624, série *Lecture Notes in Artificial Intelligence*, pages 30–41, St Petersburg, Russia, July 1992.

[Har 97] James Harland et David J. Pym. Resource-Distribution via Boolean Constraints – extended abstract. *14^{th} International Conference on Automated Deduction, CADE-12*, volume 814, série *Lecture Notes in Artificial Intelligence*, pages 222–236, Townsville, North Queensland, Australia, July 1997.

[Hod 94] Josh Hodas et Dale Miller. Logic Programming in a Fragment of Intuitionistic Linear Logic. *Journal of Information and Computation*, 110 :327–365, 1994.

[How 80] William A. Howard. The Formulae-as-Types notion of Construction. *To H.B. Curry : Essays in Combinatory Logic, λ-calculus and Formalism*, pages 479–490. Academic Press, 1980.

[Hud 93] Jörg Hudelmaier. An $\mathcal{O}(n \log n)$-space decision procedure for Intuitionistic Propositional Logic. *Journal of Logic and Computation*, 3(1) :63–75, 1993.

[Hyl 93] Martin Hyland et Valeria de Paiva. Full Intuitionistic Linear Logic – extended abstract. *Annals of Pure and Applied Logic*, 64 :273–291, 1993.

[Jec 97] Thomas J. Jech. *Set Theory*. Perspectives in Mathematical Logic. Springer Verlag, $2^{\text{ème}}$ édition, 1997.

[Kan 94] Max Kanovich. The complexity of Horn fragments of Linear Logic. *Annals of Pure and Applied Logic*, 69 :195–241, 1994.

[Kre 97] Christoph Kreitz, Heiko Mantel, Jens Otten et Stephan Schmitt. Connection-based proof construction in Linear Logic. *14^{th} International Conference on Automated Deduction*, pages 207–221, Townsville, North Queensland, Australia, 1997.

[Kri 65] Saul A. Kripke. Semantical Analysis of Intuitionistic Logic. J. Crossley et M. A. E. Dummett, éditeurs, *Formal Systems and Recursive Functions*, pages 92–130. North-Holland, 1965.

[Laf 91] Yves Lafont et Thomas Streicher. Games Semantics for Linear Logic. *6^{th} IEEE Symposium on Logic in Computer Science*, pages 43–50, Amsterdam, The Netherlands, July 1991.

[Laf 97] Yves Lafont. The Finite Model Property for various fragments of Linear Logic. *Journal of Symbolic Logic*, 62(4) :1202–1208, 1997.

[Lil 92] Johan Lilius. High-level Nets and Linear Logic. *13^{th} International Conference on Applications and Theory of Petri Nets*, number 616 in Lecture Notes in Computer Science, pages 310–327, Sheffield, UK, 1992.

[Lin 92a] Patrick D. Lincoln et John Mitchell. Operational Aspects of Linear Lambda Calculus. *7^{th} IEEE Symposium on Logic in Computer Science*, pages 235–246, Santa-Cruz, California, 1992.

[Lin 92b] Patrick D. Lincoln, John Mitchell, Andre Scedrov et Natarajan Shankar. Decision problems for Propositional Linear Logic. *Annals of Pure and Applied Logic*, 56, 1992.

[LW 00a] Dominique Larchey-Wendling et Didier Galmiche. Quantales as completions of ordered monoids : Revised semantics for Intuitionistic Linear Logic. Dieter Spreen, éditeur, *Electronic Notes in Theoretical Computer Science*, volume 35. Elsevier Science Publishers, 2000.

[LW 00b] Dominique Larchey-Wendling, Daniel Méry et Didier Galmiche. Structural Sharing and Intuitionistic Proof Search. Rapport, LORIA, 2000.

[Mac 71] Saunders Mac Lane. *Categories for the Working Mathematicians*. Graduate Texts in Mathematics. Springer-Verlag, 1971.

[Man 99] Heiko Mantel et Jens Otten. linTAP : a Tableau Prover for Linear Logic. *International Conference TABLEAUX '99*, volume 1617, série *Lecture Notes in Artificial Intelligence*, pages 216–231, Saratoga Springs, NY, USA, 1999.

[Mes 91] José Meseguer et Narciso Marti-Oliet. From Petri nets to Linear Logic. *Mathematical Structures in Computer Science*, 1 :69–101, 1991.

[Mig 97] Pierangelo Miglioli, Ugo Moscato et Mario Ornaghi. Avoiding Duplications in Tableaux Systems for Intuitionistic and Kuroda logic. *Logic Journal of the IGPL*, 5(1) :145–167, 1997.

[Mil 91] Dale Miller, Gopalan Nadathur, Frank Pfenning et Andre Scedrov. Uniform Proofs as a Foundation for Logic Programming. *Annals of Pure and Applied Logic*, 51 :125–157, 1991.

[Min 93] Grigori Mints. Resolution Calculus for the First Order Linear Logic. *Journal of Logic, Language and Information*, 2 :59–83, 1993.

[Nor 90] Bengt Nordström, Kent Petersson et Jan M. Smith. *Programming in Martin-Löf's Type Theory, an Introduction*, volume 7, série *Monographs on Computer Science*. Oxford Press, 1990.

[O'H 99] Peter W. O'Hearn et David J. Pym. The logic of bunched implications. *Bulletin of Symbolic Logic*, 5(2) :215–244, 1999.

[Oka] Mitsuhiro Okada et Kazushige Terui. The Finite Model Property for various fragments of Intuitionnistic Linear Logic. *Journal of Symbolic Logic*. to appear, earlier version available at `http://abelard.flet.keio.ac.jp/person/terui/proofsearch.ps.gz`.

[Oka 99] Mitsuhiro Okada. Phase Semantic Cut-elimination and Normalization Proofs of first- and higher-order Linear Logic. *Theoretical Computer Science*, 227 :333–396, 1999.

[Ono 93] Hiroakira Ono. Semantics for Substructural Logics. Peter Schroeder-Heister et Kosta Došen, éditeurs, *Substructural Logics*, volume 2, série *Studies in Logic and Computation*. Oxford Science Publications, 1993.

[Pin 95] Luis Pinto et Roy Dyckhoff. Loop-Free Construction of Counter-Models for Intuitionistic Propositional Logic. Behara et al., éditeurs, *Symposia Gaussiana*, pages 225–232, 1995.

[PM 93] Christine Paulin-Mohring et Benjamin Werner. Synthesis of ML programs in the system Coq. *Journal of Symbolic Computation*, 15 :607–640, 1993.

[Pra 95] Vaughan R. Pratt. Chu Spaces and their Interpretation as Concurrent Objects. *Computer Science Today : recent trends and developments*, volume 1000, série *Lecture Notes in Computer Science*, pages 392–405, 1995.

[Pra 97] Vaughan R. Pratt. Towards Full Completeness of the Linear Logic of Chu Spaces. *Electronic Notes in Theoretical Computer Science*, 6, 1997. EXPRESS'97 workshop, Santa Margherita Ligure, Italy.

[Ras 63] Helena Rasiowa et Roman Sikorski. *The Mathematics of Metamathematics*, volume 41, série *Monografie Matematyczne*. Polska Akademia Nauk, Warszawa, 1963.

[Rei 85] Wolfgang Reisig. *Petri Nets, An Introduction*, volume 4, série *EATCS Monographs on Theorical Computer Science*. Springer Verlag, 1985.

[Rit 00] Eike Ritter, David J. Pym et Lincoln A. Wallen. On the Intuitionistic Force of Classical Search. *Theoretical Computer Science*, 232(1–2) :299–333, 2000.

[Ros 90] Kimmo I. Rosenthal. *Quantales and their Applications*. Longman Scientific & Technical, 1990.

[Sam 93] Giovanni Sambin. Semantics of Pretopologies. Peter Schroeder-Heister et Kosta Došen, éditeurs, *Substructural Logics*, volume 2, série *Studies in Logic and Computation*. Oxford Science Publications, 1993.

[Sch 91] Harold Schellinx. Some Syntactical Observations on Linear Logic. *Journal of Logic and Computation*, 1(4) :537–559, 1991.

[SH 93] Peter Schroeder-Heister et Kosta Došen, éditeurs. *Substructural Logics*, volume 2, série *Studies in Logic and Computation*. Oxford Science Publications, 1993.

[Sha 92] Natarajan Shankar. Proof Search in the Intuitionistic Sequent Calculus. *11^{th} Conference on Automated Deduction*, volume 607, série *Lecture Notes in Artificial Intelligence*, pages 522–536, Saratoga Springs, June 1992.

[Sta 79] Richard Statman. Intuitionistic Propositional Logic is Polynomial-Space Complete. *Theoretical Computer Science*, 9 :67–72, 1979.

[Tam 94] Tanel Tammet. Proof Strategies in Linear Logic. *Journal of Automated Reasoning*, 12 :273–304, 1994.

[Tam 96] Tanel Tammet. A Resolution Theorem Prover for Intuitionistic Logic. *13^{th} Conference on Automated Deduction*, volume 1104, série *Lecture Notes in Artificial Intelligence*, pages 2–16, NJ, USA, 1996.

[Tro 88] Anne S. Troelstra et Dirk van Dalen. *Constructivism in Mathematics, an Introduction*, volume 121, série *Studies in Logic and the foundations of Mathematics*. North-Holland, 1988.

[Tro 92] Anne S. Troelstra. *Lectures on Linear Logic*, volume 29, série *Center for the Study of Language and Information*. CSLI, 1992.

[Tro 96] Anne S. Troelstra et Helmut Schwichtenberg. *Basic Proof Theory*, volume 43, série *Cambridge tracks in Theoretical Computer Science*. Cambridge University Press, 1996.

[Vor 96] Andrei Voronkov. Proof-search in Intuitionistic Logic based on Constraint Satisfaction. *International Workshop TABLEAUX'96*, volume 1071, série *Lecture Notes in Artificial Intelligence*, pages 312–327, Terrasini, Italy, 1996.

[Wal 89] Lincoln A. Wallen. *Automated Proof Search in Non-Classical Logics*. MIT Press, 1989.

[Wei 98] Klaus Weich. Decisions Procedures for Intuitionistic Logic by Program Extraction. *International Conference TABLEAUX'98*, volume 1397, série *Lecture Notes in Artificial Intelligence*, pages 292–306, Oisterwijk, The Netherlands, May 1998.

[Yet 90] David N. Yetter. Quantales and (noncommutative) Linear Logic. *Journal of Symbolic Logic*, 55 :41–64, 1990.

Résumé

Les logiques sont de puissants outils qui permettent la spécification de systèmes informatiques et la preuve de l'adéquation de leurs implantations avec ces spécifications. Dans le cadre des logiques sous-structurelles, nous mettons en place des outils de démonstration automatique et de construction de contre-modèles. Ces logiques intègrent la notion de ressource : au niveau de la recherche de preuve, la gestion des ressources permet la mise en place de procédures plus efficaces ; au niveau de l'interprétation sémantique, la notion de ressource permet de construire des modèles fidèles et complets.

Nous établissons un lien entre la notion syntaxique de réfutation et la notion sémantique de contre-modèle. Nous en déduisons des méthodes de démonstration de la propriété des modèles finis ainsi que des algorithmes de construction de contre-modèles. En logique intuitionniste propositionnelle, la gestion fine des ressources permet d'en déduire une implantation efficace de la recherche de preuves. En logique intuitionniste linéaire, les modèles à base de ressources permettent une preuve élégante de la propriété des modèles finis. Nous établissons un lien entre la sémantique des ressources et la sémantique à base de réseaux de Petri, ce qui permet de raffiner les résultats de complétude partiels connus jusqu'alors.

Mots-clés: Déduction automatique et recherche de preuve, sémantique et modèles, logiques intuitionniste et linéaire, gestion efficace des ressources.

Abstract

Logics can be used as powerful tools for specifying computer systems and proving the soundness of their implementations with respect to these specifications. In the field of substructural logics, we develop tools and methods for automated deduction and counter-model generation. These logics involve the notion of resource : at the level of proof-search, the management of resources enables more efficient procedures ; at the semantic level, resource models provide sound and complete interpretations.

We develop a link between the syntactic notion of refutation and the semantic notion of counter-model. We deduce methods for proving the finite model property and algorithms for building counter-models. In propositional intuitionistic logic, we are able to provide an efficient implementation of a proof-search procedure, based on a fine management of resources. In intuitionistic linear logic, resource based models constitute the core of an elegant proof of the finite model property. Furthermore, we establish a link between resource models and Petri net based models, from which we improve the preceding partial completeness results.

Keywords: Automated deduction and proof search, models and semantic, intuitionistic and linear logics, efficient resource management.