On the infinitary proof theory of logics with fixed points

Amina Doumane

Thèse de doctorat de l'Université Sorbonne Paris Cité
Préparée à l'Université Paris Diderot
Ecole Doctorale 386 — Sciences Mathématiques de Paris Centre
Laboratoire IRIF, Equipe PPS

On the infinitary proof theory of logics with fixed points

par

Amina Doumane

Thèse de doctorat en informatique fondamentale

Dirigée par Pierre-Louis Curien, David Baelde et Alexis Saurin

Présentée et soutenue publiquement à Paris Diderot le 27 Juin 2017

David Baelde	Co-directeur de thèse
Arnaud Carayol	Examinateur
Pierre-Louis Curien	Directeur de thèse
Martin Hofmann	Rapporteur
Delia Kesner	Examinateur
Luke Ong	Rapporteur
Alexis Saurin	Co-directeur de thèse
Igor Walukiewicz	Président du jury

Résumé

Logique et informatique La logique, et plus spécifiquement la théorie de la démonstration, a été particulièrement utile en informatique, en particulier en langages de programmation et en vérification formelle.

D'une part, la théorie de la démonstration a profondément révolutionné le domaine des langages de programmation via les deux paradigmes de calcul suivants: la programmation fonctionnelle et la programmation logique. En programmation fonctionnelle, pour garantir que les programmes terminent, il est usuel de considérer des langages typés. Dans ces langages, les programmes correspondent aux preuves, les types aux formules logiques et l'exécution à l'élimination des coupures. Cette dernière propriété est au cœur de cette correspondance, appelée l'isomorphisme de Curry-Howard. En programmation logique, un programme est un ensemble de formules logiques et un calcul consiste à décider si une formule, appelée requête, est une conséquence logique du programme. Le paradigme de calcul est donc la recherche de preuve. Pour que celle-ci soit efficace, le système de preuve sous-jacent doit vérifier les deux propriétés fondamentales que sont l'admissibilité de la coupure et la focalisation. Les deux propriété logiques fondamentales qui assurent le lien entre logique et langages de programmation sont donc l'élimination des coupures et la focalisation. D'autre part, en vérification formelle, la logique est utilisé comme un outil pour décrire les système et leurs propriétés. Même si le model-checking est le paradigme dominant en vérification, une approche, dite preuve-théorique, commence a regagner de l'intérêt. Elle consiste à décrire à la fois le système et la propriété à vérifier par des formules S et P, puis vérifier que le système satisfait la propriété consiste à vérifier si la formule $S \to F$ est prouvable. L'intérêt de cette approche est que la preuve résultante de $S \to F$ peut être utilisée comme un certificat, qui peut être indépendamment communiqué et vérifié.

Logiques à points fixes L'extension des logiques avec des points fixes est particulièrement intéressante en informatique. En programmation, les points fixes permettent un traitement direct, générique et intuitif des types de données (co-)inductifs comme les entiers naturels, listes, stream, etc. En vérification, une des logiques les plus utilisée est le mu-calcul, qui est une logique modale avec points fixes. Il est donc naturel de s'intéresser à la théorie de la démonstration des logiques à points fixes. Systèmes de preuves pour les logiques à points fixes Il existe deux grandes familles de systèmes de preuves pour les logiques à points fixes: les systèmes de preuves finitaires et les systèmes de preuves infinitaires.

- Dans les Systèmes de preuve finitaires, les règles pour les points fixes reflètent les principe de (co-)induction. Ces systèmes de preuve, quoique très largement utilisés, représentent plusieurs inconvénients. Le premier est que la règle de coupure est inévitable, par conséquent ces systèmes ne sont pas adaptés à la recherche de preuve.

Un autre inconvénient est que leur contenu calculatoire n'est souvent pas explicite et il est en général difficile de deviner ce qu'une preuve calcule, de savoir si deux preuves calculent la même fonction, etc.

- Dans les systèmes de preuves infinitaires, les règles pour les points fixes sont de simples règles de "déroulages", mais les preuves peuvent avoir une profondeur infinie. Ils sont plus adaptés à la recherche de preuve, et les règles de déroulage ayant un comportement calculatoire facile à analyser, le contenu calculatoire des preuves infinitaires est souvent explicite.

Même si les systèmes infinitaires existent depuis longtemps, ils ont été utilisés principalement comme des outils techniques et n'ont pas été considérés comme de vrai systèmes de preuves. Ceci est dû à un manque de résultats fondamentaux (élimination des coupures, focalisation) qui nous empêche de les voir comme tel. L'objectif de cette thèse est de pallier à cette lacune dans l'état de l'art, en développant la théorie de la démonstration infinitaire pour les logiques a points fixes, avec deux domaines d'application en vue: les langages de programmation avec types de données (co-)inductifs et la vérification des systèmes réactifs.

Contributions de la thèse Cette thèse peut être résumée comme suit: **Les logiques à points fixes peuvent être équipées de systèmes preuve infinitaires qui ont un réel statut preuve-théorique et peuvent être appliquées à d'autres domaines comme la vérification formelle.**

Naturellement, cette thèse est divisée en deux parties. Dans la première, on argumente le fait que les preuves infinitaires ont effectivement un réel statut preuve-théorique, en montrant qu'ils admettent les propriétés d'élimination des coupures et de focalisation. Dans la deuxième partie, on utilise nos développements sur les preuves infinitaires pour monter de manière constructive la complétude du mu-calcul linéaire relativement à l'axiomatisation de Kozen. Ces deux parties sont précédées par une partie introductive qui expose 1) nos choix de design, 2) un peu de background sur les systèmes de preuves pour les points fixes et 3) un outil technique qui sera très utile tout au long de la thèse: un résultat de traduction entre les systèmes finitaires et infinitaires.

Partie I: Étude preuve-théorique des logiques à points fixes Comme d'habitude, étudier les propriétés preuves théoriques dans un cadre linéaire permet de s'abstraire de beaucoup de bruit et de se focaliser sur les vrai difficultés. Aussi nous avons décidé d'établir les propriétés d'élimination des coupures et de focalisation dans le cadre de la logique linéaire additive multiplicative. Dans le Chapitre 3, nous utilisons une nouvelle technique de preuve pour établir élimination des coupures pour notre système infinitaire. A la fin du Chapitre 3, on indique comment étendre ce résultat pour la logique linéaire classique et la logique classique. Dans le Chapitre 4, nous établissons la propriété de focalisation en analysant les phénomènes qui apparaissent dans le cadre infinitaires comparé au cadre finitaire. Le Chapitre 5 est dédié à un autre problème: élucider le contenu calculatoire des preuves finitaires. Pour ce faire, on muni la logique linéaire multiplicative additive avec une sémantique dénotationnelle, pour laquelle on montre un résultat de complétude partiel. Chemin faisant, on montre la décidabilité et la complétude de l'inclusion sémantique.

Partie II: Complétude constructive pour le mu-calcul linéaire Dans son papier fondateur, Kozen en a proposé une axiomatisation pour le mu-calcul, dont la complétude a été montré 13 ans plus tard par Kaivola dans le cas linéaire et par Walukiewicz dans le cas branchant. Ces preuves de complétude sont basées sur des arguments complexes et non constructifs, qui ne donnent aucun moyen de construire une preuve pour une formule valide. La problématique de constructivité devient centrale lorsqu'on considère les preuves comme des certificats pour les outils de vérification formelle. Nous proposons une nouvelle preuve de complétude pour le mu-calcul linéaire qui est constructive, càd qui construit une preuve pour toute formule valide. Pour ce faire, nous décomposons ce problème difficile en plusieurs sous-problèmes en utilisant la correspondance entre le mu-calcul et les automates. Plus précisément, on relève les transformations d'automates comme la déterminisation et l'élimination de l'alternance au niveau de la logique. Pour résoudre chacun des sous-problèmes, on fait une recherche de preuve dans un système infinitaire, puis on transforme chaque preuve infinitaire obtenue en une preuve dans l'axiomatisation de Kozen en utilisant notre résultat de transformation de la partie introductive.

Abstract

The subject of this thesis is the proof theory of logics with fixed points, such as the μ-calculus, linear-logic with fixed points, etc. These logics are usually equipped with finitary deductive systems that rely on Park's rules for induction. other proof systems for these logics exist, which rely on infinitary proofs, but they are much less developped. This thesis contributes to reduce this deficiency by developing the infinitary proof-theory of logics with fixed points, with two domains of application in mind: programming languages with (co)inductive data types and verification of reactive systems.

This thesis contains three parts. In the first part, we recall the two main approaches to the proof theory for logics with fixed points: the finitary and the infinitary one, then we show their relationships. In the second part, we argue that infinitary proofs have a true proof-theoretical status by showing that the multiplicative additive linear-logic with fixed points admits focalization and cut-elimination. In the third part, we apply our proof-theoretical investigations to obtain a constructive proof of completeness for the linear-time μ-calculus *w.r.t.* Kozen's axiomatization.

Contents

Introduction

"On the unusual effectiveness of logic in computer science"

Even if logic has its roots in Ancient Greece with Aristotle's work, formal logic, as we know it, was born in the early 20th century to confront the crisis in the foundations of mathematics, brought about by the paradoxes of set theory. In his famous program, Hilbert was intending to show that mathematics are consistent and decidable by formalizing them. The works on logic of the time [Gö31, Tar35, Chu36, Tur36] showed that most of Hilbert's goals were impossible to achieve. Since then, mathematics and logic have followed their own paths. Nowadays, logic is considered as a mature and independent research area, with many fields of application. The most striking example of such an application is certainly computer science, which was heavily influenced by logic's development. We discuss in the following two areas of computer science where logic is particularly successful: **programing languages** which were influenced by the proof theoretical side of logic and **formal verification** which benefited most from its model theoretical one.

Logic and programming languages. Proof theory impacted strongly the domain of programming languages through the two following paradigms of computation: **functional programming** and **logic programming**.

- **Functional programming and the Curry-Howard approach.** The prototype of a functional programming language is λ-calculus introduced by Church [Chu32,Bar84]. A well-designed functional programming language is a language where programs should be safe by construction. Safe means that the program does not raise an error during its execution, due for instance to a function called with too few arguments or with arguments with the wrong types. A way to ensure this safety property is to consider type-based programming languages, in which well-typed programs "cannot go wrong" as summarized by Milner's slogan. Type systems have an intriguing similarity with proof systems. Much more than a mere coincidence, this similarity is a general fact known as the **Curry-Howard correspondence**. This correspondence exhibits an isomorphism between a number of programming languages and various proof systems.

  Let us briefly review the three main styles in which proof systems come:

  - *Hilbert-style* systems are given by a set of axioms reflecting the properties of the logical connectives. One can derive from these axioms new theorems using the

modus ponens rule:

$$\frac{A \Rightarrow B \qquad A}{B} \text{ (modus ponens)}$$

– *Natural deduction* was introduced by Gentzen in 1935, and it is the formalism that reproduces most faithfully the mathematical reasoning (hence the attribute "Natural"). It is not axiomatic in the sense that the properties of the connectives are not reflected by axioms, but instead by means of inferences rules. Inference rules for connectives can either introduce a connective or eliminate it. A proof is a tree of formulas, where each node is justified by an introduction or elimination rule and each leaf is called a "hypothesis". Some leaves of the tree are discharged, meaning that they are not to be considered as hypothesis. If all leaves are discharged, then the conclusion holds without hypothesis. We call **cut** a pattern formed by an introduction rule for a connective followed immediately by an elimination rule for this connective. Cuts are somehow detours in the proof. To show that cuts can be eliminated from natural deduction proofs, Gentzen introduced another formalism of proofs: sequent calculus.

– *Sequent calculus* is based on the notion of **sequent** which generalizes that of formula: a sequent has the form $\Gamma \vdash \Delta$ where Γ and Δ are two lists of formulas, and it can be understood as "the conjunction of the elements of Γ implies the disjunction of the elements of Δ" (the symbol $\vdash$ can be thought of as an implication). The inference rules for connectives can either introduce a connective on the right or on the left of the sequent; and a proof is a tree of sequents built using inference rules. In sequent calculus, the cut is not a pattern of rules, as in natural deduction, but it is a proper rule of the system:

$$\frac{\Delta_1 \vdash \Gamma_1, A \qquad A, \Delta_2 \vdash \Gamma_2}{\Delta_1, \Delta_2 \vdash \Gamma_1, \Gamma_2} \text{ (Cut)}$$

Gentzen showed that cuts can be eliminated from sequent calculus proofs using an effective procedure. Prawitz showed the same result for natural deduction proofs.

These results are the heart of the Curry-Howard correspondence, the latter being not simply a mapping between proofs and programs, and between formulas and types, but also, and more importantly, an isomorphism between the cut-elimination procedure (or any other form of proof normalization) and the execution of programs.

The first instance of this correspondence was discovered by Curry, who established in 1958 an isomorphism between Hilbert style proof systems and combinatorial logic, here the use of *modus ponens* corresponds to the application of combinators. Later, in 1969, Howard pointed out the correspondence between minimal logic in natural deduction and simply-typed lambda calculus [How80]. Since then, the Curry-Howard correspondence has been extended to other frameworks, either by enriching the logic (for instance, Girard and Reynolds [Gir72,Rey74] discovered a correspondence between second-order logic and polymorpism, and Griffin [Gri90] discovered that Felleisen's control operator $\mathcal{C}$ [FFKD87] can be typed with $\neg\neg A \Rightarrow A$, which was a stepping stone for the Curry-Howard extension to classical logic); or by exploring other formalisms

of proofs (for instance by considering a Curry-Howard correspondence for sequent calculus [CH00]).

Over time, the meaning of the Curry-Howard correspondence has evolved from the simple observation of an isomorphism between an existing proof system and an existing programming language, to something more "productive". Indeed, one can either start from a typed programming language and extract its isomorphic proof system which may have an interesting logical meaning. Conversely, one could start from a known proof system and try to understand its computational content in order to design its corresponding programming language, this is for instance how a number of calculi were born [Par92, CH00, MM09].

Curry-Howard was also productive in renewing questions related to the semantics of logic, putting an emphasis on the semantics of proofs, in contrast to tranditional thruth or provability semantics. In this sense, **linear logic** [Gir87] arose from Girard's study of the denotational semantics of system F.

- **Logic programming.** This paradigm of computation orginated directly from logical considerations. Here, a program is a set of logical formulas, and a computation is performed by deciding whether a formula, called a query, is a logical consequence of the program or not.

  The most popular example of a logic programming language is Prolog [SS86]. Its formulas are described using first-order Horn clauses, and its execution proceeds as follows: to decide whether a query is a consequence of the program, it adds its negation the program, and searches for a proof of false. This proof-search uses the *SLD resolution* technique [VEK76], which amounts to search for a clause that allows to conclude immediately the goal (initially this goal is false), in which case the proof-search continues with the hypothesis of this clause as goals, etc.

  In [MNPS91], Miller *et al.* characterize, in a generic way, what a Logic Programming language is. This is done through the notion of *uniform proofs*. A uniform proof is one that can be found by a goal-directed search. This generic vision of a logic programming language allowed to reformulate existing languages (such as Prolog) in proof-theoretical terms. It allowed also to design new languages such as λ-Prolog [MN12], by going from first-order Horn clauses to higher-order hereditary Harrop formulas.

  This idea of uniform proofs had another fundamental consequence: when trying to extend it to linear logic, Andreoli came across **focalization** [And92]. While reversible connectives (that is, the connectives that can be applied without risking the loss of provability) are known to behave well in proof-search, the focalization result shows that non-reversible connectives have also good properties. This result has had several applications in both logic (it is at the heart of polarized linear logic [Lau02], and ludics [Gir01, Ter11]) and programming languages much beyond logic programming (it gave new insights on our understanding of evaluation order of programming languages [DJS97], CPS translations, and pattern matching [CM10, Zei09]).

Formal verification. In programming languages, logic is a tool *internal* to the computation paradigm: typing rules are the building blocks to construct (safe) functional programs and formulas are the building blocks of logic programs. In formal verification, logic is used as

an *external* tool to describe existing computer systems, and make formal statements about them.

One of the most successful approaches to verification is **model checking** [CE81]. In this approach, the system is described by a mathematical structure $\mathcal{M}$ and the property to check about it is represented by a formula φ in a certain logic. Then checking whether the system satisfies the property amounts to check whether $\mathcal{M}$ is a model of φ, that is $\mathcal{M} \models \varphi$. In this approach, the choice of the logical language is very crucial. On the one hand, it should be expressive enough to describe complex behaviours of the system. On the other hand, the problem of deciding whether a model satisfies a formula should be decidable, and preferably with a reasonable complexity. Temporal logics [Pnu77] are one of the more widely used specification languages in this sense, and Linear Temporal logic (LTL) and Computation tree logic (CTL*) are cases in point.

The temporal modalities of LTL are $\odot$ (in the next moment of time) $\mathbf{G}$ (always) and $\mathbf{F}$ (eventually); formulas are built using these temporal modalities combined with propositional connectives and atomic propositions. For instance, the LTL formula $\mathbf{FG}\varphi$ means that "eventually, φ will always holds". The models of LTL formulas are linear structures. We can extend this interpretation to branching structures, by setting $\mathcal{M} \models \varphi$ if for every path p of $\mathcal{M}$, we have that $p \models \varphi$. Thus, there is an implicit universal quantification on paths in LTL formulas. The logic CTL* allows both universal and existential quantification, through the path quantifier $\forall$ (in all paths) and $\exists$ (for some path); CTL* formulas are build using these path quantifiers combined with LTL connectives. For instance, the CTL* formula $\exists\mathbf{G}\varphi$ means that "There is a branch of the model where φ always holds". The use of CTL* and LTL goes beyond the academic frame, and finds a considerable industrial success throught tools such as SMV and VIS, FormalCheck, etc, with many applications in major software companies (IBM, Intel, Microsoft, etc).

There is another approach to verification that lives in the shadow of the mainstream model-checking paradigm, which we call the **proof-theoretic approach**, following Walukiewicz [Wal94]. It consists in modeling both the system and the property to check by two formulas φ_S and φ_P respectively, then checking whether the system meets the property is reduced to checking the provability of $\varphi_P \to \varphi_P$. This approach was the prevailing paradigm of verification at the time of the introduction of formal verification [MP81]. However, it ran out of steam for two reasons: first the complexity of the provability problem is way less optimal than that of the satisfiability problem, second the new logical languages introduced since then simply do not admit (well-behaved) proof systems. Recently, new issues about certification have emerged, renewing the interest on the proof-theoretical approach to verification. Indeed, if the model-checking technique outputs a binary answer to the verification problem (either $\mathcal{M} \models \varphi$ or not), the proof-theoretical one outputs a proof (of the formula $\varphi_P \to \varphi_S$) which can be used as a **certificate**, which can be communicated and independently verified [Sha08].

Proof certificates [Mil11] are particularly promising: they are expressed in a well-established language (that of proofs), their production can be (partially at least) mechanized through theorem provers, and they are compositional by essence (via the cut rule).

To sum up:
Logic, and more specifically proof-theory, has been very useful in programming languages and in formal verification.

Fixed point logics in computer science

As discussed earlier, logic has strong links with different areas of computer science. It was therefore natural to try to extend these links to a wide range of logics. Among the most interesting and fruitful extensions, **logics with fixed points** are a case in point. These logics contain two special connectives: the connective μ and its dual ν. A formula of the form $\mu X.F$ (resp. $\nu X.F$) can be understood as the "least (resp. greatest) fixed points of the operator $X \mapsto F(X)$". In the following, we review the use of fixed points logics in the domains of programming languages and formal verification.

Fixed point logics in functional programming. Least and greatest fixed points allow to treat in a direct, generic and intuitive way data (natural numbers, lists, etc) and co-data (streams, infinite trees, etc). The least fixed point operator μ allows to define finite data types such as natural numbers, lists, finite trees, etc. For instance, in linear logic with fixed point, the type of natural numbers can be represented by $\mathsf{Nat} := \mu X.1 \oplus X$, and every natural number n can be represented by the proof $\underline{n}$ inductively defined as follows:

$$\underline{0} \;=\; \dfrac{\dfrac{\dfrac{}{\vdash 1}\,(1)}{\vdash 1 \oplus \mathsf{Nat}}\,(\oplus_1)}{\vdash \mathsf{Nat}}\,(\mu) \qquad\qquad \underline{n+1} \;=\; \dfrac{\dfrac{\dfrac{\underline{n}}{\vdash \mathsf{nat}}}{\vdash 1 \oplus \mathsf{Nat}}\,(\oplus_2)}{\vdash \mathsf{Nat}}\,(\mu)$$

The type of lists of natural numbers can be represented by $\mathsf{List} = \mu X.1 \oplus (\mathsf{Nat} \otimes X)$, and every list l can be represented by the proof $\underline{l}$ inductively defined as follows:

$$\underline{\varepsilon} \;=\; \dfrac{\dfrac{\dfrac{}{\vdash 1}\,(1)}{\vdash 1 \oplus (\mathsf{Nat} \otimes \mathsf{List})}\,(\oplus_1)}{\vdash \mathsf{List}}\,(\mu) \qquad\qquad \underline{n :: l} \;=\; \dfrac{\dfrac{\dfrac{\underline{n}}{\vdash \mathsf{Nat}} \quad \dfrac{\underline{l}}{\vdash \mathsf{List}}}{\vdash \mathsf{Nat} \otimes \mathsf{List}}}{\vdash 1 \oplus (\mathsf{Nat} \otimes \mathsf{List})}\,(\oplus_2)}{\vdash \mathsf{List}}\,(\mu)$$

Dually, the greatest fixed point operator allow to define infinite data types (called co-data), such as streams, infinite trees, etc. For instance, the type of infinite streams of integers can be represented in linear logic with fixed points by $\mathsf{Stream} := \nu X.\mathsf{Nat} \otimes X$, and every stream $n_0 :: n_1 :: n_2 \ldots$ can be represented by the following "proof":

$$\dfrac{\dfrac{\dfrac{\underline{n_0}}{\vdash \mathsf{Nat}} \quad \dfrac{\dfrac{\dfrac{\underline{n_1}}{\vdash \mathsf{Nat}} \quad \dfrac{\vdots}{\vdash \mathsf{Stream}}}{\vdash \mathsf{Nat} \otimes \mathsf{Stream}}\,(\otimes)}{\vdash \mathsf{Stream}}\,(\nu)}{\vdash \mathsf{Nat} \otimes \mathsf{Stream}}\,(\otimes)}{\vdash \mathsf{Stream}}\,(\nu)$$

Treating data and co-data in programming languages as fixed points is not the only possibility. They can either be introduced as constants of the language, as this is the case in System T for instance [GTL89]. This has the drawback of being limited, since we can only use the (co)data that are already provided by the syntax, and cannot declare new ones when needed. Another possibility is to use second-order, which subsumes fixed points (See [Mat99] for an embedding of fixed points in second-order). This has the drawback of being non-intuitive and algorithmically non-optimal. For instance, encoding natural numbers in system F yields Church numbers, for which a linear amount of time is required to compute the predecessor, while this can be performed in constant time in logics with fixed points [Mat99].

Fixed points in formal verification. In terms of specifications, least fixed points correspond to terminating behaviour, and are used to capture properties asserting that something will happen, such as liveness and progress. Greatest fixed points correspond to infinite behaviours, and are used to capture properties asserting that something happens infinitely often, such as safety or invariance. The most popular examples of temporal logics with fixed points are the linear-time and the branching-time μ-calculi [Koz83], which can be seen respectively as the extension of LTL and CTL* with least and greatest fixed points. The use of the linear-time and the branching time μ-calculi in formal verification has been fruitful for three reasons:

- They are well-balanced between expressiveness and complexity. Compared to their second-order counterparts S1S (monadic second-order logic with one successor, which is equi-expressive to the linear-time μ-calculus) and S2S (monadic second-order logic with two successors, which is equi-expressive to the branching-time μ-calculus), the linear-time and the branching-time μ-calculi have better algorithmic properties. For instance, satisfiability of the linear-time μ-calculus is PSPACE-complete, and it is EXPTIME-complete for the branching-time μ-calculus, while it is non-elementary for S1S and S2S.

- If the balance between expressiveness and complexity is the crucial property for the model-checking paradigm, axiomatizability is the crucial one for the proof-theoretic paradigm. The μ-calculus was equipped with a very natural axiomatization since its introduction by Kozen. This axiomatization was shown to be complete for the branching-time μ-calculus by Walukiewicz [Wal95] and for the linear-time by Kaivola [Kai95].

- There is a fruitful relationship between the μ-calculus and automata theory, at the core of which lies the equivalence between linear-time μ-calculus formulas and alternating parity automata over words (APW), and between branching-time μ-calculus formulas and alternating parity automata over trees (APT). In fact, most of the deep results on the μ-calculus have been (can be) obtained using automata theory. Among these results, the exact complexity of the satisfiability problem [EJ91, Mad94], the strictness of the fixed point alternation hierarchy in the branching-time case [Bra98], the fact that this hierarchy collapses at level 0 in the linear-time case [Lan05], the correspondence with monadic second order logic [Mad94], all build on the correspondence with automata theory.

To sum up:
Logics with fixed points are widely used in the areas of programming languages and of systems verification.

In this thesis, we focus on the proof-theoretical aspects of logics with fixed points. In a proof-theoretical study, we are not only interested by provability (what are the statements that can be established by a given proof system) but also, and more importantly, by the structure of proofs and their interaction. This includes establishing results such as cut-elimination and focalization. We review in the next section the state of the art for the proof theory for logics with fixed points.

Proof systems for logics with fixed points

There are two main families of proof systems with fixed points: finitary proof systems with explicit rules of (co)induction and infinitary proof systems.

Finitary proof systems. In these proof systems, the induction principle is reflected by an explicit rule, called **Park's rule**. Let us recall that the induction principle is based on the characterization of the least fixed point of an operator $X \mapsto F(X)$, $\mu X.F$, as its least pre-fixed point. This means that:

i) $\mu X.F$ is a pre-fixed point, that is $F(\mu X.F) \leq \mu X.F$.

ii) $\mu X.F$ is smaller than any other pre-fixed point S of F, that is:

$$F(S) \leq S \Rightarrow \mu X.F \leq S.$$

These two points yield respectively the following two rules for the μ connective:

$$\frac{\Delta \vdash F[\mu X.F/X], \Gamma}{\Delta \vdash \mu X.F, \Gamma}\ (\mu_{\mathsf{r}}) \qquad \frac{F[S/X] \vdash S}{\mu X.F \vdash S}\ (\mu_{\mathsf{l}})$$

Dually, the coinduction principle, based on the characterization of the greatest fixed point of an operator as its greatest post-fixed point, yields the following two rules:

$$\frac{S \vdash F[S/X]}{S \vdash \nu X.F}\ (\nu_{\mathsf{r}}) \qquad \frac{\Delta, F[\nu X.F/X] \vdash \Gamma}{\Delta, \nu X.F \vdash \Gamma}\ (\nu_{\mathsf{l}})$$

The problem of these rules is that, in general, the cut rule is not admissible in the proof systems using them. To fix that, the following variant of the rules (μ_{l}) and (ν_{r}), containing a "hidden cut", are usually used instead:

$$\frac{\Gamma \vdash S, \Delta \qquad S \vdash F[S/X]}{\Gamma \vdash \nu X.F, \Delta}\ (\nu) \qquad \frac{F[S/X] \vdash S \qquad \Gamma, S \vdash \Delta}{\Gamma, \mu X.F \vdash \Delta}\ (\mu_{\mathsf{r}})$$

Using this precise formulation of the fixed points rules, the admissibility of the cut-rule, and more importantly the cut-elimination property have been shown in several frameworks:

Martin-Löf has shown cut-elimination for an intuitionistic natural deduction system with iterated inductive definitions in the framework of dependent types [ML71]. McDowell and Miller have shown cut-elimination for an intuitionistic sequent calculus system with higher-order quantification and definitions [MM00], then Momigliano and Tiu extended it to the logic Linc [TM12]. Brotherston and Simpson showed the admissibility of the cut rule in the setting of first-order classical logic with inductive definitions [BS07]. In all these results, the syntax of fixed point formulas allows only for purely inductive or coinductive definitions, no interleaving between fixed points of distinct nature being allowed. The first result of cut-elimination in a setting where the use of fixed points is not restricted is due to Mendler, who shows strong normalization for second-order lambda calculus with co-inductive types [Men91]. Baelde shows cut-elimination and focalization for the logic μMALL (linear logic with least and greatest fixed points) [Bae12a].

The fact that the cut rule is unavoidable (either in an explicit or a hidden way), is coherent with the fact that, to show a statement with induction, one should usually generalize it. But this means that proof systems with explicit rules of induction are fundamentally not suited to proof-search, and that there is very little we can do to fix that. Another drawback of proofs using explicit induction is that their computational meaning is usually not explicit. Consider for instance linear logic with fixed points, equipped with explicit induction rules. We have seen that we can express in that framework the type of lists of natural numbers and the type of infinite streams of natural numbers. We can then define a proof (shown below) that concatenates a list and a stream into a stream, by recursing over the list with the invariant $\mathsf{Stream} \multimap \mathsf{Stream}$.

$$\dfrac{\dfrac{\dfrac{}{1 \vdash \mathsf{Stream} \multimap \mathsf{Stream}}\ \ldots,(\mathsf{Ax}) \qquad \dfrac{\dfrac{\dfrac{}{\mathsf{Nat}, \mathsf{Stream} \multimap \mathsf{Stream}, \mathsf{Stream} \vdash \mathsf{Nat} \otimes \mathsf{Stream}}\ \ldots,(\mathsf{Ax})}{\mathsf{Nat}, \mathsf{Stream} \multimap \mathsf{Stream}, \mathsf{Stream} \vdash \mathsf{Stream}}}{\mathsf{Nat} \otimes (\mathsf{Stream} \multimap \mathsf{Stream}) \vdash \mathsf{Stream} \multimap \mathsf{Stream}}\ (\otimes),(\multimap)}}{1 \oplus (\mathsf{Nat} \otimes (\mathsf{Stream} \multimap \mathsf{Stream})) \vdash \mathsf{Stream} \multimap \mathsf{Stream}}\ (\oplus)}{\mathsf{List} \vdash \mathsf{Stream} \multimap \mathsf{Stream}}\ (\mu)$$

It is not trivial to convince oneself that this proof does compute the concatenation function. More generally, it is hard to tell what such proofs compute, when two proofs compute the same function, etc.

Infinitary proof systems. These systems are obtained by using, instead of the Park's rules (ν_r) and (μ_l), the following unfolding rules:

$$\dfrac{\Gamma \vdash F[\nu X.F/X], \Delta}{\Gamma \vdash \nu X.F, \Delta}\ (\nu_\mathsf{r}) \qquad \dfrac{\Gamma, F[\mu X.F] \vdash \Delta}{\Gamma, \mu X.F \vdash \Delta}\ (\mu_\mathsf{l})$$

and allowing infinite derivation trees, called **pre-proofs**. Clearly, these rules do not reflect the difference in nature between μ and ν. More importantly, these pre-proofs are unsound, since one can derive the empty sequent as follows:

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash \mu X.X}\ (\mu_\mathsf{r})}{\vdash \mu X.X}\ (\mu_\mathsf{r}) \qquad \dfrac{\dfrac{\vdots}{\mu X.X \vdash}\ (\mu_\mathsf{l})}{\mu X.X \vdash}\ (\mu_\mathsf{l})}{\vdash}\ (\mathsf{Cut})$$

To get a proof system which is sound, we declare a pre-proof to be a proof if and only if it satisfies the **validity condition**. This condition says, roughly speaking, that in every infinite branch there should be either a least fixed point unfolded infinitely often in the left or a greatest fixed unfolded infinitely often in the right.

There is a natural restriction to infinitary proofs given by regular proofs, that is, proofs that have only finitely many sub-trees. These proofs are called **circular proofs**, since they can be represented as finite trees with loops. This restricted **cyclic proof system** is more suited for a computer science use, as its proofs can be finitely represented and manipulated.

Infinitary and circular proof systems have existed for a long time, but in the shadows, and more as technical tools than as proper proof systems. Indeed, their infinitary nature makes them suitable intermediary objects between the syntax and the semantics, so that we usually find them in completeness proofs. This is the case, for instance, with the μ-calculus **refutations** of Niwinski and Walukiewicz [NW96], which are used as an intermediary proof system in Walukiewicz's proof of completeness of the μ-calculus with respect to Kozen's axiomatization [Wal95].

In the last decades, infinitary proof systems have started to come out from the shadows and have begun to be considered as proper proofs. For instance, Dam and Gurov [DG02] propose a complete infinitary proof system for the μ-calculus with explicit approximations. Dax *et al.* proposed a complete infinitary proof system for the linear-time μ-calculus [DHL06]. Santocanale introduced a circular proof system for the purely additive linear logic with fixed points [San02]. Brotherston came up with an infinitary and a circular proof system for first order classical logic with inductive definitions. This increasing interest in infinitary proofs is due to two facts. On the one hand, they are very suited to proof search, as there is no invariant to guess in their formulation, contrarily to Park's rules. This has been exploited by the theorem prover QuodLibet [AKSW03], which uses the mathematical reasoning corresponding to infinite proofs: the **infinite descent**. This has also been exploited by Dax *et al.* to give an optimal algorithm to check validity for the linear-time μ-calculus formulas, using their complete infinitary proof system [DHL06]. On the other hand, by the Curry-Howard correspondence, new reasoning methods bring new ways of programming. The use of infinitary proofs, in particular circular ones, represents a promising new method of writing (co)inductive programs, allowing to get rid of the cumbersome (co)inductive rules. Another interesting feature of the infinitary proofs is that their computational meaning is more explicit compared to finitary proof systems. For instance, it is clear that the following circular proof computes the concatenation of a list and a stream:

$$\dfrac{\dfrac{\dfrac{}{1, \mathsf{Stream} \vdash \mathsf{Stream}}\,(1),(\mathsf{Ax}) \qquad \dfrac{\dfrac{\dfrac{}{\mathsf{Nat} \vdash \mathsf{Nat}}\,(\mathsf{Ax}) \qquad \dfrac{}{\mathsf{List}, \mathsf{Stream} \vdash \mathsf{Stream}}\,(\star)}{\mathsf{Nat}, \mathsf{List}, \mathsf{Stream} \vdash \mathsf{Nat} \otimes \mathsf{Stream}}\,(\otimes),(\mathsf{Ax})}{\mathsf{Nat} \otimes \mathsf{List}, \mathsf{Stream} \vdash \mathsf{Stream}}\,(\otimes),(\nu)}{1 \oplus (\mathsf{Nat} \otimes \mathsf{List}), \mathsf{Stream} \vdash \mathsf{Stream}}\,(\oplus)}{\dfrac{\mathsf{List}, \mathsf{Stream} \vdash \mathsf{Stream} \qquad (\star)}{\mathsf{List} \vdash \mathsf{Stream} \multimap \mathsf{Stream}}\,(\multimap)}\,(\mu)$$

Despite their compelling interest, few proof-theoretical investigations have been done about infinitary proof theory. For instance, we do not know if the cut-elimination and focal-

ization results hold for the infinitary proof systems mentioned before. The only exception to that is Fortier and Santocanale's work [FS13], in which cut-elimination is established for the infinitary proof system for the purely additive fragment of linear logic with fixed points.

To sum up:
There are two main families of proof systems for logics with fixed points: finitary proof systems with an explicit rule of (co)induction and infinitary proof systems. While the first family has received much attention from the scientific community and is now very well understood, much less work has been done to develop the second one from a proof-theoretical viewpoint.

The aim of this thesis is to contribute to reduce this deficiency by developing the infinitary proof theory of fixed point logics, with two domains of application in mind: programming languages with (co)inductive data types and verification of reactive systems.

Our work

Our thesis can be summarized by the following sentence:

Our thesis:
Logics with fixed points can be equipped with infinitary proof systems having a true proof-theoretical status which can be fruitfully applied to other domains such as formal verification.

Naturally, this thesis is split in two parts. One where we argue that infinitary proof systems have indeed a real proof-theoretical status, by focusing on the cut-elimination and focalization properties. In the other part, we show an application of our proof-theoretical investigations on infinitary proofs, by showing in a constructive way, and using our developments on infinitary proofs, a completeness proof for the linear-time μ-calculus with respect to Kozen's axiomatization. Before developing these two parts containing our main contributions, we start by an introductory part, where we expose 1) our design choices, 2) some background on proof systems and fixed-points logics and 3) a technical tool which will be very useful all along the thesis: a translation results between finitary and circular proofs.

Part 0: Design choices, background and technical tools

Design choices. As we aim at a proof-theoretical investigation of our logics with fixed points, we choose to work with the formalism of sequent calculus which is the most suited to analyze the structure of proofs and to establish technical results such as cut-elimination. There are many variants of sequent calculus, which differ mainly by the way sequents are presented: sequents as sets of formulas, as multisets of formulas, as lists of formulas or as **occurrences of formulas**. These last two formulations are the most widely used when we aim at a computational interpretation of proofs, we made therefore the choice to use formulas occurrences. Although this could seem insignificant, this choice had surprising consequences.

The most striking one is a **translation result** from circular proofs to finitary ones, which relies heavily on this precise formulation of sequents.

An invaluable technical tool: the translation procedure. Usually it is not difficult to go from proofs with explicit induction rules (we call them simply finitary proofs) to circular ones: the induction principle is reflected by a cycle. The converse is a hard problem, that has been conjectured by Brotherstone and Simpson in the framework of first order logic with inductive definitions [BS07]. We do not have a full translation result from circular proofs to finitary ones, but we provide a sufficient condition on circular proofs, that allows to translate them into finitary ones. The translation procedure we describe relies on the presentation of sequents as sets of formula occurrences. It is surprising how such a presentation, inspired from proofs-as-programs considerations, has brought new insights on this difficult problem.

Here is a raodmap for this introductory part:

- Chapter 1: We recall the logics that we will be interested in along the thesis (the propositional classical logic LK, the linear logics MALL and LL, the linear temporal logic LTL and its fragment that contains only the next operator that we call $\mathsf{LK}\odot$). We recall the semantics and proof systems for each of these logics.

- Chapter 2: First, we give some background on lattice theory and the fundamental fixed point theorems of the literature. Second, we show how to extend the syntax of a logic with least and greatest fixed points, and study some properties of the resulting formulas.

  Then we show the two standard approaches, discussed earlier, to incorporate fixed points to a proof system S (which can be any of the proof systems of Chapter 1):

  - The first one consists in adding Park's rules, and the obtained proof system is denoted $\mu\mathsf{S}$. For instance, the extension of the multiplication additive linear logic MALL with fixed points is denoted $\mu\mathsf{MALL}$.
  - The second approach consists in adding to S the unfolding rules for fixed points and allowing infinite derivations. The obtained proof system is denoted $\mu\mathsf{S}^\infty$. Then we introduce the circular proof system $\mu\mathsf{S}^\omega$, whose proofs are finite graphs. This system can be seen as a fragment of $\mu\mathsf{S}^\infty$ since its proofs are in correspondence with the regular $\mu\mathsf{S}^\infty$ proofs. For instance, $\mu\mathsf{MALL}^\infty$ and $\mu\mathsf{MALL}^\omega$ denote respectively the infinitary and the circular proof systems for MALL extended with fixed points.

  Finally, we study the relationship between the finitary proof system $\mu\mathsf{S}$ and the circular one $\mu\mathsf{S}^\omega$. For that, we show that:

  - Every finitary proof can be translated into a circular one.
  - Conversely, we show that if a circular proof satisfies a property that we call the **translatability criterion**, then it can be translated into a finitary proof. This translation result will be used, as a crucial step, twice in the thesis.

Part I: Proof-theoretical study of infinitary proofs

Our goal is to study the proof-theoretical aspects of infinitary proofs, by showing in particular the two main results of cut-elimination and focalization. As usual, studying proof-theoretical properties in a linear framework allows to get away from the noise introduced by the other logics, and focus on the real difficulties. That is why we concentrate on $\mu\mathsf{MALL}^\infty$, the infinitary proof system for linear logic with fixed points, showing that it admits both cut-elimination and focalization.

As mentioned before, the only known cut-elimination result in an infinitary setting is due to Fortier and Santocanale in the restricted setting of purely additive linear logic. The topological argument they use does not scale to richer settings such as $\mu\mathsf{MALL}^\infty$, and we had to develop new proof techniques to deal with the multiplicative connective. We hint on how to extend this result to richer logics such as $\mu\mathsf{LL}^\infty$, and $\mu\mathsf{LK}^\infty$ for instance, the full development is left to a future work.

Focalization has never been addressed in an infinitary setting. We show it in the framework of $\mu\mathsf{MALL}^\infty$, stressing out the phenomena that arise in the infinitary setting compared to the finitary one.

We dedicated the rest of the part to another problem: elucidating the computational meaning of proofs in the finitary proof system with explicite induction. As discussed earlier, the computational meaning of proof systems with explicit induction rules is usually not clear. For instance, it is hard to tell what such proofs compute, when two proofs compute the same function, etc. Addressing this issue requires to step back from the finite, syntactic proof system under consideration and to start considering its **semantics**. In the rest of the part, we investigate the semantics of the finitary proof system $\mu\mathsf{MALL}$.

As our domain of interpretation of proofs, we consider ludics [Gir01] which can be regarded as a variant of game semantics, where the basic objects are well-behaved strategies, called **designs**. Girard introduced ludics with the aim of bringing closer together syntax and semantics in the study of proofs and proved a full completeness result with respect to proofs of a polarized variant of MALL. Extending this interpretation to all of linear logic, including exponentials, has been challenging and required to deal with non-determinism [BF11]. As we shall see, accounting for least and greatest fixed points is much easier, and can essentially be done in Girard's original framework. Still, we shall work in Terui's reformulation of ludics, **computational ludics** [Ter11], since it is more convenient to work with and slightly more general, for instance the objects of computational ludics may contain cut while Girard's original designs are cut-free: this happens to be very handy when working with greatest fixed point.

We provide $\mu\mathsf{MALL}$ with a denotational semantics, interpreting proofs by designs and formulas by particular sets of designs called behaviours. Then we prove a completeness result for the class of "essentially finite designs", which are those designs performing a finite computation followed by a copycat. On the way to completeness, we establish decidability and completeness of semantic inclusion.

Here is a roadmap for Part I:

- Chapter 3: We show that $\mu\mathsf{MALL}^\infty$ admits the cut elimination property. We show this result using an unusual semantical argument. We discuss the extension of this result to the proof systems $\mu\mathsf{LL}$, $\mu\mathsf{LK}$ and $\mu\mathsf{LK}\odot$.
- Chapter 4: We show that $\mu\mathsf{MALL}^\infty$ admits the focalization property.
- Chapter 5: We give a denotational semantics for $\mu\mathsf{MALL}$ in Ludics. We investigate the problem of completeness, and give a partial result by going through an infinitary proof system and using the translation result from Chapter 2.

Part II: Constructive completeness for the linear-time μ-calculus

We give a new proof of completeness for the linear-time μ-calculus with respect to Kozen's axiomatization ($\mu\mathsf{LK}\odot$ is the sequent calculus style of this axiomatization). Our proof has the advantage of being *constructive*. This means that if a formula is valid, we show how to build in an effective way a $\mu\mathsf{LK}\odot$ proof of it. Earlier proofs of completeness were not constructive in the sense that they show that a valid formula *must* have a proof, but they do not specify a way to obtain it. Indeed, their arguments rely on proofs by contradiction, and it is known to be difficult, when not impossible, to extract constructive and algorithmic content from proofs by contradiction.

To get this constructive completeness result, we generalize an idea that has being used in earlier proofs of completeness [Wal95, Kai95], which consists in introducing a sub-class $\mathcal{C}_1$ of the class of μ-calculus formulas $\mathcal{C}_0$, and establishing the following two results:

1) For every valid formula φ_0 in $\mathcal{C}_0$, there is a valid formula φ_2 in $\mathcal{C}_1$ such that $\varphi_1 \vdash \varphi_0$ is provable in $\mu\mathsf{LK}\odot$.

2) Every valid formula of $\mathcal{C}_1$ is provable. This is the completeness result restricted to $\mathcal{C}_1$.

Completeness is proved by combining 1) and 2) via a cut rule:

$$\dfrac{\dfrac{2)}{\vdash \varphi_2} \qquad \dfrac{1)}{\varphi_2 \vdash \varphi_1}}{\vdash \varphi_1}\ (\mathsf{Cut})$$

Our idea is to introduce, instead of one intermediary class, several intermediary classes. To design these classes, we will take advantage of the correspondence between the linear-time μ-calculus formulas and alternating parity automata over words (APW). More precisely, we can encode every APW $\mathcal{A}$ by a μ-calculus formula $[\mathcal{A}]$. The image of APW by this encoding gives us our first class $\mathcal{C}_1$. We introduce the other classes in the same way:

- The class $\mathcal{C}_2$ is the image of non-deterministic parity automata (NPW).
- The class $\mathcal{C}_3$ is the image of non-deterministic Büchi automata (NBW).
- The class $\mathcal{C}_4$ is the image of deterministic Büchi automata (DBW).

Since we have $DBW \subseteq NBW \subseteq NPW \subseteq APW$ we have also that $\mathcal{C}_4 \subseteq \mathcal{C}_3 \subseteq \mathcal{C}_2 \subseteq \mathcal{C}_1$. To show completeness, we will show that:

- For every $3 < i \leq 0$, if φ_i is a valid formula in $\mathcal{C}_i$, then there is a valid formula φ_{i+1} in $\mathcal{C}_{i+1}$ such that $\varphi_{i+1} \vdash \varphi_i$ is provable in $\mu\mathsf{LK}\odot$.
- Every valid formula in $\mathcal{C}_4$ is provable.

Using these results, and provided that they are proved in a constructive way, we obtain a constructive completeness proof for the linear-time μ-calculus.

Let us mention that these equivalences are well-known in the automata side, since they correspond respectively to alternation elimination, parity simplification and determinization results. Our goal is to lift them to the provability level. To do so, we go through the circular proof system $\mu\mathsf{LK}\odot^\omega$: we perform first a proof-search in $\mu\mathsf{LK}\odot^\omega$, then we **transform in an effective way** the obtained circular proof into a $\mu\mathsf{LK}\odot$ one, using our translation procedure from Chapter 2. This yields a constructive proof for the full linear-time μ-calculus.

Here is a roadmap of Part II:

- Chapter 6: We discuss and analyze earlier proofs of completeness for the linear-time and the branching-time μ-calculus. Then we give the roadmap for our proof of completeness for the linear-time case.
- Chapter 7: We recall the syntax and semantics for the linear-time μ-calculus. Then we reintroduce, for clarity, the proof system $\mu\mathsf{LK}$ which is the target of the completeness proof, and $\mu\mathsf{LK}\odot^\omega$, the intermediary proof system.
- Chapter 8: We give some background on automata over infinite words, and recall the different classes of automata that we shall work with.
- Chapter 9: We recall the correspondence between the linear-time μ-calculus formulas and alternating parity automata over words (APW). Then we show that this equivalence can be lifted to the level of provability.
- Chapter 10: we recall the correspondence between APW and NPW, and the one between NPW and NBW. Then we show that these equivalences can also be lifted to the provability level.
- Chapter 11: We show that Büchi language inclusions are provable in $\mu\mathsf{LK}\odot$.
- Chapter 12: Building on the results obtained in Chapters 9-11, we show our constructive completeness result.

We end up this thesis by a concluding Chapter 12.2, in which we expose the perspectives of our work.

Chapter 1

Background on proof theory

In this chapter, we recall the syntax, the semantics and the sequent calculus for the different logics that we will study in the course of this thesis: classical logic, linear logic and linear temporal logic.

In Chapter 2, our goal will be to extend, in a generic way, a logic and its proof systems with least and greatest fixed points. With this in mind, we will try to present uniformly the aforementioned logics. This means in particular that some notions which are not strictly necessary for the rest of the thesis will be nevertheless introduced.

We have chosen to work with sequent calculus for its elegance, and since it is an excellent framework to study the structure of proofs. There are many presentations of sequents in the litterature: sequents as sets of formulas, as lists of formulas, as multisets of formulas, etc. We have chosen to introduce first our proof systems (Sections 1.2, 1.3 and 1.4) with the formalism of sequents as lists. In Section 1.5, we discuss and compare the different presentations of sequents that exist in the litterature. The one that meets the most our future needs is the presentation of sequents as **sets of formula occurrences**, we introduce it in details in 1.5.2 and show how to interpret the proof systems introduced earlier with this specific presentation of sequents.

1.1 Syntax, semantics and proof systems

The syntax of a logic can be described by a signature which is a set of symbols coming with their arities. Formulas are built inductively using these symbols.

Definition 1.1. A ***signature*** $\mathcal{L}$ is a pair (S, ar) of a set of symbols S and a function $ar : S \to \omega$ that assigns to every symbol an integer called its ***arity***.

The set of ***formulas*** $(\varphi, \psi, \dots)$ over the signature $\mathcal{L}$, denoted $\mathcal{F}_{\mathcal{L}}$, is defined inductively as follows:

$$\varphi = s(\varphi_1, \dots, \varphi_n) \qquad \text{where} \qquad s \in S \text{ and } ar(s) = n.$$

Formulas are sequences of symbols without a particular meaning. To give them a meaning, many approaches exist, but we only consider the two following in this thesis.

The first one is a **semantical** approach. It is based on a set $\mathcal{U}$ of mathematical structures called **models**, and a relation $\models \subseteq \mathcal{U} \times \mathcal{F}_{\mathcal{L}}$ which links models to formulas, we usually write $\mathcal{M} \models \varphi$, and say that φ is **true** in the model $\mathcal{M}$. When a formula is true in every model, we say that it is **valid**.

The second approach is **syntactic**. It associates the meaning of formulas with the roles that they can play in inferences rules. Inference rules come in different styles, but we will focus only in the **sequent calculus** presentation. The specificity of sequent calculus is to generalize the notion of formula into a richer notion which is that of a sequent.

Definition 1.2. A ***sequent*** is given by two lists of formulas Γ and Δ and is denoted $\Gamma \vdash \Delta$.

A sequent $\Gamma \vdash \Delta$ can be understood as follows: the conjunction of the formulas of Γ implies the disjunction of the formulas of Δ.

Depending on use, sequents may be defined in a different way. We discuss this later in Section 1.5.

A proof system in sequent calculus is given by a set of **inference rules**. The latter are the building blocks of sequent calculus ***proofs***, which are the well formed trees obtained using inference rules.

Since we are carrying out a proof-theoretical study of logics in this thesis, we will focus in this chapter on the proof systems for the logics we are interested in and simply recall their semantics.

1.2 Propositional classical logic

1.2.1 LK formulas

The formulas of propositional classical logic are given by the following syntax:

Definition 1.3. Let $\mathcal{P} = \{\mathfrak{p}, \mathfrak{q}, \dots\}$ be a set of atoms. LK ***formulas*** are given by:

$$\varphi, \psi ::= \mathfrak{p} \mid \mathfrak{p}^\perp \mid \bot \mid \top \mid \varphi \vee \psi \mid \varphi \wedge \psi$$

In other words, the signature of propositional classical logic is $\mathcal{L}_{\mathsf{LK}} = (S_{\mathsf{LK}}, ar_{\mathsf{LK}})$ where $S_{\mathsf{LK}} = \{\mathfrak{p}, \mathfrak{p}^\perp, \vee, \wedge \mid \mathfrak{p} \in \mathcal{P}\}$ and:

$$ar_{\mathsf{LK}}(\vee) = ar_{\mathsf{LK}}(\wedge) = 2, \quad ar_{\mathsf{LK}}(\top) = ar_{\mathsf{LK}}(\bot) = 0, \quad \forall \mathfrak{p} \in \mathcal{P}, ar_{\mathsf{LK}}(\mathfrak{p}) = ar_{\mathsf{LK}}(\mathfrak{p}^\perp) = 0.$$

The signature of a logic being trivially inferable from the syntax of its formulas, we will not explicitly show it for the upcoming logics.

Definition 1.4. ***Negation*** is the involution on LK formulas written $\varphi^\perp$ and satisfying:

$$(\varphi \vee \psi)^\perp = \psi^\perp \wedge \varphi^\perp \qquad \top^\perp = \bot \qquad (\mathfrak{p})^\perp = \mathfrak{p}^\perp$$

1.2.2 Semantics of LK formulas

Let $(\{\mathsf{t}, \mathsf{f}\}, \vee, \wedge, \neg\}$ be the boolean lattice, whose top is t, whose bottom is f, and $\vee$, $\wedge$ and $\neg$ are respectively the usual join, meet and complement operations. We interpret LK formulas in this boolean lattice as follows:

Definition 1.5. A ***model*** is a subset $\mathcal{M} \subseteq \mathcal{P}$ of atoms. The ***interpretation*** $\|\varphi\|^{\mathcal{M}}$ of an LK formula φ under a model $\mathcal{M}$ is a boolean defined by induction on φ as follows:

$$\begin{array}{llll} \|\varphi \vee \psi\|^{\mathcal{M}} & = & \|\varphi\|^{\mathcal{M}} \vee \|\psi\|^{\mathcal{M}} & \|\top\|^{\mathcal{M}} = \mathsf{t} \\ \|\varphi \wedge \psi\|^{\mathcal{M}} & = & \|\varphi\|^{\mathcal{M}} \wedge \|\psi\|^{\mathcal{M}} & \|\bot\|^{\mathcal{M}} = \mathsf{f} \\ \|\mathfrak{p}^{\perp}\|^{\mathcal{M}} & = & \neg\|\mathfrak{p}\|^{\mathcal{M}} & \|\mathfrak{p}\|^{\mathcal{M}} = \mathsf{t} \text{ if } \mathfrak{p} \in \mathcal{M} \\ & & & \quad = \mathsf{f} \text{ otherwise} \end{array}$$

We say that a formula φ is ***true*** in a model $\mathcal{M}$ and we write $\mathcal{M} \models \varphi$ if $\|\varphi\|^{\mathcal{M}} = \mathsf{t}$. A formula is said to be ***valid*** if for every model $\mathcal{M}$, we have $\varphi \models \mathcal{M}$.

We generalize the notion of validity to a sequent as follows: we say that a ***sequent*** $\Gamma \vdash \Delta$ ***is valid*** if the formula $(\wedge\Gamma)^{\perp} \vee (\vee\Delta)$ is.

1.2.3 The proof systems LK, $\mathsf{LK}_{\mathsf{rev}}$ and $\mathsf{LK}_{\mathsf{os}}$

Definition 1.6. LK is the proof system whose rules are shown in Figure 1.1.

The rules of LK can be classified into three categories: the **logical rules**, which are the rules decomposing the top-level connective of a formula in their conclusion sequent, the **identity rules**, which are the axiom and cut rules, these rules aim at recognizing two formulas as being identical, and ensure that the deductive relation is reflexive and transitive. Finally, the **structural rules**, which are contraction, weakening and exchange, and which restructure the sequent.

Admissibility of the cut rule. One of the most fundamental properties of LK is the admissibility of the cut rule, which means that the sequents which are provable in LK can be also proved without using the cut rule.

Theorem 1.1 (Gentzen [Gen35])**.** *A sequent* $\Gamma \vdash \Delta$ *is provable in* LK *if and only if it is provable in the system* LK *without the cut rule.*

An immediate consequence of this result is the coherence of LK. Indeed, if for a formula F, both F and $F^{\perp}$ where provable, then the empty sequent would be also provable by combing the proofs of F and $F^{\perp}$ via a cut. Since the empty sequent has no cut free proof, this contradicts Theorem 1.1.

Corollary 1.1. *In* LK *we cannot prove both a formula and its negation.*

The proof of the admissibility of the cut rule, due to Gentzen, shows more than that: not only the classical sequent calculus with and without the cut rule prove the same theorems, but on top of that, we can transform, via an **effective procedure**, every proof in the calculus with the cut rule into a proof of the same conclusion without cuts. This procedure is described by a set of rewriting rules, which can be found for example in [Gen35]. Figure 1.2 shows the example of the $(\vee_{\mathsf{l}}) - (\wedge_{\mathsf{r}})$ rewriting rule. This discovery is at the heart of what we call the **Curry-Howard correspondence**: the proof of a formula φ can be seen as a program of type φ, the cut elimination steps can be seen as the execution steps of this program and the cut-free proof obtained after cut elimination corresponds to the result of the program.

Identity rules

$$\frac{}{\varphi \vdash \varphi}\ (\mathsf{Ax}) \qquad \frac{}{\vdash \varphi, \varphi^{\perp}}\ (\mathsf{Ax}) \qquad \frac{}{\varphi, \varphi^{\perp} \vdash}\ (\mathsf{Ax})$$

$$\frac{\Gamma_1 \vdash \varphi, \Delta_1 \qquad \Gamma_2 \vdash \varphi^{\perp}, \Delta_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\ (\mathsf{Cut}) \qquad \frac{\Gamma_1, \varphi \vdash \Delta_1 \qquad \Gamma_2, \varphi^{\perp} \vdash \Delta_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\ (\mathsf{Cut})$$

$$\frac{\Gamma_1 \vdash \varphi, \Delta_1 \qquad \Gamma_2, \varphi \vdash \Delta_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\ (\mathsf{Cut}) \qquad \frac{\Gamma_1, \varphi \vdash \Delta_1 \qquad \Gamma_2 \vdash \varphi, \Delta_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\ (\mathsf{Cut})$$

Structural rules

$$\frac{\Gamma \vdash \Delta}{\Gamma \vdash \varphi, \Delta}(\mathsf{W_r}) \qquad \frac{\Gamma \vdash \varphi, \varphi, \Delta}{\Gamma \vdash \varphi, \Delta}(\mathsf{C_r}) \qquad \frac{\Gamma \vdash \Delta_1, \psi, \varphi, \Delta_2}{\Gamma \vdash \Delta_1, \varphi, \psi, \Delta_2}(\mathsf{Ex_r})$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, \varphi \vdash \Delta}(\mathsf{W_l}) \qquad \frac{\Gamma, \varphi, \varphi \vdash \Delta}{\Gamma, \varphi \vdash \Delta}(\mathsf{C_l}) \qquad \frac{\Gamma_1, \psi, \varphi, \Gamma_2 \vdash \Delta}{\Gamma_1, \varphi, \psi, \Gamma_2 \vdash \Delta}(\mathsf{Ex_l})$$

Logical rules

$$\frac{\Gamma_1 \vdash \varphi, \Delta_1 \qquad \Gamma_2 \vdash \psi, \Delta_2}{\Gamma_1, \Gamma_2 \vdash \varphi \wedge \psi, \Delta_1, \Delta_2}\ (\wedge_{\mathsf{r}}) \qquad \frac{\Gamma \vdash \varphi, \Delta \qquad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \varphi \wedge \psi, \Delta}\ (\wedge_{\mathsf{r}}) \qquad \frac{\Gamma, \varphi, \psi \vdash \Delta}{\Gamma, \varphi \wedge \psi \vdash \Delta}\ (\wedge_{\mathsf{l}})$$

$$\frac{\Gamma_1, \varphi \vdash \Delta_1 \qquad \Gamma_2, \psi \vdash \Delta_2}{\Gamma_1, \Gamma_2, \varphi \vee \psi \vdash \Delta_1, \Delta_2}\ (\vee_{\mathsf{l}}) \qquad \frac{\Gamma, \varphi \vdash \Delta \qquad \Gamma, \psi \vdash \Delta}{\Gamma, \varphi \vee \psi \vdash \Delta}\ (\vee_{\mathsf{l}}) \qquad \frac{\Gamma \vdash \varphi, \psi, \Delta}{\Gamma \vdash \varphi \vee \psi, \Delta}\ (\vee_{\mathsf{r}})$$

$$\frac{\Gamma \vdash \varphi_i, \Delta}{\Gamma \vdash \varphi_1 \vee \varphi_2, \Delta}\ (\vee_{\mathsf{r}}) \qquad \frac{\Gamma, \varphi_i \vdash \Delta}{\Gamma, \varphi_1 \wedge \varphi_2 \vdash \Delta}\ (\wedge_{\mathsf{l}})$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, \top \vdash \Delta}\ (\top_{\mathsf{l}}) \qquad \frac{}{\Gamma \vdash \top, \Delta}\ (\top_{\mathsf{r}}) \qquad \frac{}{\vdash \top}\ (\top_{\mathsf{r}})$$

$$\frac{}{\bot \vdash}\ (\bot_{\mathsf{l}}) \qquad \frac{}{\Gamma, \bot \vdash \Delta}\ (\bot_{\mathsf{l}}) \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \bot, \Delta}\ (\bot_{\mathsf{r}})$$

Figure 1.1: Inference rules for LK.

Reversible and non-reversible Rules. We can classify the rules of LK in two categoris: the reversible and the non-reversible ones. A rule is ***reversible*** means that if its conclusion is provable, so are its premisses. For example, the following right rule of disjunction is reversible:

$$\frac{\Gamma \vdash \varphi, \psi, \Delta}{\Gamma \vdash \varphi \vee \psi, \Delta}\ (\vee_{\mathsf{r}})$$

$$\dfrac{\dfrac{\Gamma_1, A \vdash \Delta_1 \qquad \Gamma_1, B \vdash \Delta_1}{\Gamma_1, A \vee B \vdash \Delta_1}\,(\vee_\mathsf{l}) \qquad \dfrac{\Gamma_2 \vdash A, \Delta_2}{\Gamma_2 \vdash A \vee B, \Delta_2}\,(\vee_\mathsf{r})}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\,(\mathsf{Cut})$$

$$\downarrow$$

$$\dfrac{\Gamma_1, A \vdash \Delta_1 \qquad \Gamma_2 \vdash A, \Delta_2}{\Gamma_1, \Gamma_2 \vdash \Delta_1, \Delta_2}\,(\mathsf{Cut})$$

Figure 1.2: The $(\vee_\mathsf{l}) - (\wedge_\mathsf{r})$ cut-elimination rule

Indeed, we can derive its premisse from its conclusion using the following derivation:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma \vdash \varphi^\perp, \varphi}\,(\mathsf{Ax})}{\Gamma \vdash \varphi^\perp, \varphi, \psi, \Delta}\,(\mathsf{Ex_r}),(\mathsf{W_r}) \qquad \dfrac{\dfrac{}{\Gamma \vdash \psi^\perp, \psi}\,(\mathsf{Ax})}{\Gamma \vdash \psi^\perp, \varphi, \psi, \Delta}\,(\mathsf{Ex_r}),(\mathsf{W_r})}{\Gamma \vdash \varphi^\perp \wedge \psi^\perp, \varphi, \psi, \Delta}\,(\wedge_\mathsf{r}) \qquad \Gamma \vdash \varphi \vee \psi, \Delta}{\Gamma \vdash \varphi, \psi, \Delta}\,(\mathsf{Cut})$$

An example of a ***non-reversible*** rule is the following right rule of disjunction:

$$\dfrac{\Gamma \vdash \varphi_i, \Delta}{\Gamma \vdash \varphi_1 \vee \varphi_2, \Delta}\,(\vee_\mathsf{r})$$

Indeed, the sequent $\top \vdash \top \vee \bot$ is provable in LK, but applying the last rule $(\vee_\mathsf{r})$, choosing the right disjunct, leads to the sequent $\top \vdash \bot$ which is not provable.

The same holds for the following right conjunction rules, where the first is reversible and the second is not:

$$\dfrac{\Gamma \vdash \varphi, \Delta \qquad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \varphi \wedge \psi, \Delta}\,(\wedge_\mathsf{r}) \qquad\qquad \dfrac{\Gamma_1 \vdash \varphi, \Delta_1 \qquad \Gamma_2 \vdash \psi, \Delta_2}{\Gamma_1, \Gamma_2 \vdash \varphi \wedge \psi, \Delta_1, \Delta_2}\,(\wedge_\mathsf{r})$$

Despite this, the reversible and non-reversible versions of all the rules are inter-derivable. For instance, if the reversible version of the $(\vee_\mathsf{r})$ is available, we can derive the non-reversible version as follows:

$$\dfrac{\dfrac{\Gamma \vdash \varphi_2, \Delta}{\Gamma \vdash \varphi_1, \varphi_2, \Delta}\,(\mathsf{W_r})}{\Gamma \vdash \varphi_1 \vee \varphi_2, \Delta}\,(\vee_\mathsf{r}) \qquad\qquad \dfrac{\dfrac{\dfrac{\Gamma \vdash \varphi_1, \Delta}{\Gamma \vdash \varphi_2, \varphi_1, \Delta}\,(\mathsf{W_r})}{\Gamma \vdash \varphi_1, \varphi_2, \Delta}\,(\mathsf{Ex_r})}{\Gamma \vdash \varphi_1 \vee \varphi_2, \Delta}\,(\vee_\mathsf{r})$$

Thus, if we are only concerned with provability, and if we are not interested in proofs themselves and their interaction (this will be the case for example in Part II of this thesis), it is preferable to work with a sub-system of LK, which has exactly the same expressive power as LK (that is, it proves the same sequents), but which is more suited for proof search, we call it $\mathsf{LK_{rev}}$. In this system, contraction and weakening are integrated to the rules, thus they do not appear explicitly in the system. The cut rule being admissible, it disappears also, and we keep only the logical reversible rules.

Identity rules

$$\frac{}{\Gamma, \varphi \vdash \varphi, \Delta}\ (\mathsf{Ax}) \qquad \frac{}{\Gamma \vdash \varphi, \varphi^{\perp}, \Delta}\ (\mathsf{Ax}) \qquad \frac{}{\Gamma, \varphi, \varphi^{\perp} \vdash \Delta}\ (\mathsf{Ax})$$

Structural rules

$$\frac{\Gamma_1, \psi, \varphi, \Gamma_2 \vdash \Delta}{\Gamma_1, \varphi, \psi, \Gamma_2 \vdash \Delta}(\mathsf{Ex_l}) \qquad \frac{\Gamma \vdash \Delta_1, \psi, \varphi, \Delta_2}{\Gamma \vdash \Delta_1, \varphi, \psi, \Delta_2}(\mathsf{Ex_r})$$

Logical rules

$$\frac{\Gamma, \varphi \vdash \Delta \quad \Gamma, \psi \vdash \Delta}{\Gamma, \varphi \vee \psi \vdash \Delta}\ (\vee_{\mathsf{l}}) \qquad \frac{\Gamma \vdash \varphi, \Delta \quad \Gamma \vdash \psi, \Delta}{\Gamma \vdash \varphi \wedge \psi, \Delta}\ (\wedge_{\mathsf{r}}) \qquad \frac{\Gamma \vdash \varphi, \psi, \Delta}{\Gamma \vdash \varphi \vee \psi, \Delta}\ (\vee_{\mathsf{r}})$$

$$\frac{\Gamma, \varphi, \psi \vdash \Delta}{\Gamma, \varphi \wedge \psi \vdash \Delta}\ (\wedge_{\mathsf{l}}) \qquad \frac{\Gamma \vdash \Delta}{\Gamma, \top \vdash \Delta}\ (\top_{\mathsf{l}}) \qquad \frac{}{\Gamma \vdash \top, \Delta}\ (\top_{\mathsf{r}})$$

$$\frac{}{\Gamma, \bot \vdash \Delta}\ (\bot_{\mathsf{l}}) \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \bot, \Delta}\ (\bot_{\mathsf{r}})$$

Figure 1.3: Inference rules for $\mathsf{LK_{rev}}$.

Definition 1.7. $\mathsf{LK_{rev}}$ is the proof system whose rules are shown in Figure 1.3.

We can give yet another presentation of the proof system LK, this time restricting the shape of sequents. In this presentation, we allow only sequents of the form $\varepsilon \vdash \Gamma$, where ε is the empty list, which we write simply as $\vdash \Gamma$, and adapt the LK rules accordingly. We call the obtained proof system $\mathsf{LK_{os}}$, for one-sided LK.

Definition 1.8. $\mathsf{LK_{os}}$ is the proof system whose rules are shown in Figure 1.4.

This system is equivalent to LK in the sense that if $\vdash \Gamma$ is provable in $\mathsf{LK_{os}}$ then it is also provable in LK. Conversely, if $\Gamma \vdash \Delta$ is provable in LK, then $\vdash \Gamma^{\perp}, \Delta$ is provable in LK. The advantage of $\mathsf{LK_{os}}$ is that is has half as many rules as LK.

The proof systems for propositional classical logic presentend above are all sound and complete, this means that:

Theorem 1.2. *A sequent is valid if and only if it is provable in* LK *(resp.* $\mathsf{LK_{rev}}$*, resp.* $\mathsf{LK_{os}}$*).*

Lafont's critical pair. The cut elimination procedure for LK is non-deterministic in a strong sense: a proof may be reduced to two different proofs which are completely different, we say that it is **non-confluent**. An example of such situation is given by the following derivation, called **Lafont's critical pair**. This proof will reduce either to π_1 or π_2 depending

Identity rules

$$\frac{}{\vdash \varphi, \varphi^\perp}\ (\mathsf{Ax}) \qquad \frac{\vdash \varphi^\perp, \Delta_1 \quad \vdash \varphi, \Delta_2}{\vdash \Delta_1, \Delta_2}\ (\mathsf{Cut})$$

Structural rules

$$\frac{\vdash \Delta}{\vdash \varphi, \Delta}(\mathsf{W}) \qquad \frac{\vdash \varphi, \varphi, \Delta}{\vdash \varphi, \Delta}(\mathsf{C}) \qquad \frac{\vdash \Delta_1, \psi, \varphi, \Delta_2}{\vdash \Delta_1, \varphi, \psi, \Delta_2}(\mathsf{Ex})$$

Logical rules

$$\frac{\vdash \varphi_i, \Delta}{\vdash \varphi_1 \vee \varphi_2, \Delta}\ (\vee) \qquad \frac{\vdash \varphi, \Delta \quad \vdash \psi, \Delta}{\vdash \varphi \wedge \psi, \Delta}\ (\wedge) \qquad \frac{\vdash \varphi, \psi, \Delta}{\vdash \varphi \vee \psi, \Delta}\ (\vee) \qquad \frac{\vdash \varphi, \Delta_1 \quad \vdash \psi, \Delta_2}{\vdash \varphi \wedge \psi, \Delta_1, \Delta_2}\ (\wedge)$$

$$\frac{}{\vdash \top, \Delta}\ (\top) \qquad \frac{}{\vdash \top}\ (\top) \qquad \frac{\vdash \Delta}{\vdash \bot, \Delta}\ (\bot)$$

Figure 1.4: Inference rules for $\mathsf{LK_{os}}$.

on the direction we choose to reduce cuts.

$$\frac{\dfrac{\dfrac{\pi_1}{\vdash \Gamma}}{\vdash \Gamma, \Gamma}\ (\mathsf{W})}{\vdash \Gamma}\ (\mathsf{C}) \quad {}^\star\!\leftarrow \quad \frac{\dfrac{\dfrac{\pi_1}{\vdash \Gamma}}{\vdash \varphi^\perp, \Gamma}\ (\mathsf{W}) \qquad \dfrac{\dfrac{\pi_2}{\vdash \Gamma}}{\vdash \varphi, \Gamma}\ (\mathsf{W})}{\vdash \Gamma, \Gamma}\ (\mathsf{Cut})}{\vdash \Gamma}\ (\mathsf{C}) \quad \rightarrow^\star \quad \frac{\dfrac{\dfrac{\pi_2}{\vdash \Gamma}}{\vdash \Gamma, \Gamma}\ (\mathsf{W})}{\vdash \Gamma}\ (\mathsf{C})$$

If we think about cut elimination as a computational process transforming a proof into a result, this last observation is problematic since it means that a computation does not give a unique result. One way to get rid of such critical pairs and to retrieve more determinism is to control the use of structural rules. This is one of the innovations of linear logic, which we introduce in the next section.

1.3 Linear Logic

In $\mathsf{LK_{os}}$, some rules require identical contexts in all their premises, which they supperpose in the conclusion sequent, we call them **additive** rules. This is the case of the following rules for instance:

$$\frac{\vdash \varphi, \Delta \quad \vdash \psi, \Delta}{\vdash \varphi \wedge \psi, \Delta}\ (\wedge) \qquad \frac{\vdash \varphi_i, \Delta}{\vdash \varphi_1 \vee \varphi_2, \Delta}\ (\vee)$$

Other rules on the contrary juxtapose the different contexts coming from their premises; we call them **multiplicative** rules. This is the case of the following rules for instance:

$$\frac{\vdash \varphi, \Delta_1 \qquad \vdash \psi, \Delta_2}{\vdash \varphi \wedge \psi, \Delta_1, \Delta_2}\ (\wedge) \qquad\qquad \frac{\vdash \varphi, \psi, \Delta}{\vdash \varphi \vee \psi, \Delta}\ (\vee)$$

It turns out that the additive and the multiplicative rules of every $\mathsf{LK}_{\mathsf{os}}$ connective are inter-derivable. But these rules are inter-derivable only through the weakening and contraction rules. If we remove these structural rules from LK, the additive and multiplicative versions of $\mathsf{LK}_{\mathsf{os}}$ rules are not equivalent anymore, and it becomes necessary to reflect this distinction in the syntax, by providing an additive and a multiplicative version for each connective. For conjunction, the additive version is denoted $\&$ and the multiplicative one is denoted $\otimes$. For disjunction, the additive version is denoted $\oplus$ and the multiplicative one is denoted $\parr$. Finally, each unit comes also in two versions: $\top$ and 0 are the additive versions of $\top$ and $\bot$ respectively, and 1 and $\bot$ are their multiplicative versions. The proof system obtained from LK by removing contraction and weakening, and by providing an additive and multiplicative version for each connective is called the **multiplicative additive linear logic**, and it is denoted MALL.

1.3.1 MALL formulas

Definition 1.9. Let $\mathcal{P} = \{\mathfrak{p}, \mathfrak{q}, \dots\}$ be a set of atoms. MALL ***formulas*** are given by:

$$\varphi, \psi ::= \mathfrak{p} \mid \mathfrak{p}^\perp \mid 0 \mid \bot \mid \top \mid 1 \mid \varphi \oplus \psi \mid \varphi \parr \psi \mid \varphi \,\&\, \psi \mid \varphi \otimes \psi$$

To define negation in LK, we relied on the De Morgan laws, which set the dual of conjunction to be the disjunction, and the dual of true to be false. Now that we have two versions for each connective, we have to figure out which version of a connective is dual to a given version. For that purpose, we make use of the cut elimination rule, our goal being to have a proof system that admits cut-elimination. If we set for instance the dual of $\&$ to be $\parr$, the following cut-elimination step $(\&) - (\parr)$ would imply the use of structural rules, which are not available in MALL.

$$\dfrac{\dfrac{\vdash A, B, \Gamma}{\vdash A \parr B, \Gamma}\ (\parr) \qquad \dfrac{\vdash A^\perp, \Delta \qquad \vdash B^\perp, \Delta}{\vdash A^\perp \,\&\, B^\perp, \Delta}\ (\&)}{\vdash \Gamma, \Delta}\ (\mathsf{Cut})$$

$$\downarrow$$

$$\dfrac{\dfrac{\dfrac{\vdash A, B, \Gamma \qquad \vdash A^\perp, \Delta}{\vdash B, \Gamma, \Delta}\ (\mathsf{Cut}) \qquad \vdash B^\perp, \Delta}{\vdash \Gamma, \Delta, \Delta}\ (\mathsf{Cut})}{\vdash \Gamma, \Delta}\ (\mathsf{C}),(\mathsf{Ex})$$

The choice of dual connectives which permits us to write the cut elimination rules without invoking structural rules is the one that sets the dual of $\&$ to be $\oplus$, and the dual of $\parr$ to be $\otimes$. If we apply this analysis to all the other connectives, we get the following definition of negation:

Identity and structural rules

$$\frac{}{\vdash \varphi, \varphi^\perp}\ (\mathsf{Ax}) \qquad \frac{\vdash \varphi^\perp, \Delta_1 \quad \vdash \varphi, \Delta_2}{\vdash \Delta_1, \Delta_2}\ (\mathsf{Cut}) \qquad \frac{\vdash \Delta_1, \psi, \varphi, \Delta_2}{\vdash \Delta_1, \varphi, \psi, \Delta_2}\ (\mathsf{Ex})$$

Logical rules

$$\frac{\vdash \varphi_i, \Delta}{\vdash \varphi_1 \oplus \varphi_2, \Delta}\ (\oplus_i) \qquad \frac{\vdash \varphi, \Delta \quad \vdash \psi, \Delta}{\vdash \varphi \,\&\, \psi, \Delta}\ (\&) \qquad \frac{\vdash \varphi, \psi, \Delta}{\vdash \varphi \parr \psi, \Delta}\ (\parr) \qquad \frac{\vdash \varphi, \Delta_1 \quad \vdash \psi, \Delta_2}{\vdash \varphi \otimes \psi, \Delta_1, \Delta_2}\ (\otimes)$$

$$\frac{}{\vdash \top, \Delta}\ (\top) \qquad \frac{}{\vdash 1}\ (1) \qquad \frac{\vdash \Delta}{\vdash \bot, \Delta}\ (\bot)$$

Figure 1.5: Inference rules for MALL.

Definition 1.10. ***Negation*** is the involution on formulas written $\varphi^\perp$ and satisfying

$$(\varphi \parr \psi)^\perp = \psi^\perp \otimes \varphi^\perp \quad \bot^\perp = 1 \quad 0^\perp = \top \quad (\varphi \oplus \psi)^\perp = \psi^\perp \,\&\, \varphi^\perp \quad (\mathfrak{p})^\perp = \mathfrak{p}^\perp$$

1.3.2 The proof system MALL

Definition 1.11. MALL is the proof system whose rules are shown in Figure 1.5.

The logic MALL is a very weak logic, but it has a bunch of good properties. Its simplicity makes it a very good nucleus for many different extensions. For instance, by adding second order variables and $\forall$ and $\exists$ quantification on them, we get a very powerful logic: MALL2 ([Gir01]). Later in this thesis (Part II), we will focus on another extension of MALL, obtained by adding least and greatest fixed points.

1.3.3 LL formulas

Another well-known extension of MALL is the full linear logic, denoted LL. In LL, we will reintroduce weakening and contraction, but in a very controlled way. More precisely, these rules can be applied in LL only on formulas marked by **exponential** connectives ? and !. Thus, the syntax of LL is the following:

Definition 1.12. Let $\mathcal{P} = \{\mathfrak{p}, \mathfrak{q}, \dots\}$ be a set of atoms. LL ***formulas*** are given by:

$$\varphi, \psi ::= \mathfrak{p} \mid \mathfrak{p}^\perp \mid 0 \mid \bot \mid \top \mid 1 \mid \varphi \oplus \psi \mid \varphi \parr \psi \mid \varphi \,\&\, \psi \mid \varphi \otimes \psi \mid !\varphi \mid ?\psi$$

We define negation of LL formulas by extending that of MALL with the equation $(?\varphi)^\perp = !\varphi^\perp$.

1.3.4 The proof system LL

Definition 1.13. LL is the proof system whose rules are those shown in Figure 1.5 together with the following rules, called ***exponential rules***:

$$\frac{\vdash ?\varphi, ?\varphi, \Delta}{\vdash ?\varphi, \Delta}\ (\mathsf{C}) \qquad \frac{\vdash \Delta}{\vdash ?\varphi, \Delta}\ (\mathsf{W}) \qquad \frac{\vdash \varphi, \Delta}{\vdash ?\varphi, \Delta}\ (?) \qquad \frac{\vdash \varphi, ?\Delta}{\vdash !\varphi, ?\Delta}\ (!)$$

Exponential rules allow to recover the expressive power of LK: LL proofs can be embedded in LK by erasing the exponential connectives and identifying the two versions of each connective by the original LK connective. Conversely, LK proofs can be encoded by LL ones ([Lau02]), and this encoding is sound and complete for provability.

With LL we recover the expressivity of LK, in addition we earn the confluence of the calculus.

1.3.5 Semantics of MALL and LL formulas

If we are seeking for a complete semantics for linear logic, the one that we used for LK is not appropriate. For instance, the sequent $\vdash 1 \parr 1$, which would be valid if we interpret 1 and $\parr$ by their corresponding classical connectives, is not provable in MALL. We present a semantics for linear logic, called **phase semantics**, which is sound and complete for MALL.

Definition 1.14. A ***phase space*** is given by a tuple $(M, ., 1_M, \bot)$, where $(M, ., 1_M)$ is commutative monoid and $\bot \subseteq M$.

Let $X, Y \subseteq M$. We define $X.Y$ and $X^\bot$ as follows:

$$X.Y = \{x.y \mid x \in X \text{ and } y \in Y\} \qquad \text{and} \qquad X^\bot = \{y \mid X.\{y\} \subseteq \bot\}$$

A subset $X \subseteq M$ is called a ***fact*** if $X^{\bot\bot} = X$.

A ***phase model*** is a tuple $(M, ., 1_M, \bot, v)$ where $(M, ., 1_M, \bot)$ is a phase space and v is a valuation that maps atoms to facts of $(M, ., 1_M, \bot)$.

A typical example of phase models is the **syntactic phase space**, we recall it in the following.

Example 1.1. Let M be the set of finite multisets of MALL formulas. Let . be the operation of concatenation of two multisets and 1 be the empty multiset, and let $\bot$ be the set of provable multisets. The phase space $(M, ., 1, \bot)$ is called the ***syntactic phase space*** of MALL.

Definition 1.15. Let $\mathcal{M} = (M, ., 1_M, \bot, v)$ be a phase model. The ***interpretation*** $\|\varphi\|^{\mathcal{M}}$ of a formula φ under $\mathcal{M}$ is defined by:

$$\|\varphi \otimes \psi\|^{\mathcal{M}} = (\|\varphi\|^{\mathcal{M}}.\|\psi\|^{\mathcal{M}})^{\bot\bot} \qquad \|\mathfrak{p}\|^{\mathcal{M}} = v(\mathfrak{p})$$

$$\|\varphi \,\&\, \psi\|^{\mathcal{M}} = \|\varphi\|^{\mathcal{M}} \cap \|\psi\|^{\mathcal{M}} \qquad \|\varphi^\bot\|^{\mathcal{M}} = (\|\varphi\|^{\mathcal{M}})^\bot$$

$$\|\bot\|^{\mathcal{M}} = \bot \qquad \|\top\|^{\mathcal{M}} = M$$

A formula φ is ***true*** in $\mathcal{M}$, and we write $\mathcal{M} \models \varphi$, if and only if $1_M \in \|\varphi\|^{\mathcal{M}}$. It is said to be ***valid*** if for every model $\mathcal{M}$, we have $\mathcal{M} \models \varphi$.

The interpretation of a formula in the syntactic phase model provides us with insight into the meaning of phase semantics interpretation. Morally, the interpretation of a formula in this space is the set of contexts that make it provable. Here, we can see clearly the difference between the two versions of conjunction for example. In the interpretation of $\otimes$, we find this idea of juxtaposition that we addressed while introducing multiplicatives, which is reflected by the use of composition operation of the monoid. The additive nature of &, which superposes contexts, is reflected by the operation of intersection used to interpret it.

Phase semantics interpretation is correct and complete for MALL, as stated by the following theorem.

Theorem 1.3 (Girard [Gir87])**.** *A* MALL *formula is valid if and only if it is provable in* MALL.

To interpret the exponentials, we have to enrich the notion of a model as follows.

Definition 1.16. Let $(M, ., 1_M, \bot)$ be a phase space. A submonoid J of M is ***weakly idempotent*** if:

$$\forall a \in J,\ \{a\}^{\bot\bot} \subseteq \{a.a\}^{\bot\bot}$$

An ***enriched phase model*** $\mathcal{M}$ is given by a phase model $(M, ., 1_M, \bot, v)$ and a set $I = \bot^{\bot} \cap J$ where J is a submonoid of M satisfying the weak idempotency property.

We extend the definition of the interpretation under an enriched phase model to LL formulas by:

$$\|?\varphi\|^{\mathcal{M}} = (I \cap \|\varphi\|^{\mathcal{M}})^{\bot\bot}$$

A formula φ is ***true*** in $\mathcal{M}$ and we write $\mathcal{M} \models \varphi$ if and only if $1_M \in \|\varphi\|\mathcal{M}$. It is said to be ***valid*** if for every enriched phase model $\mathcal{M}$ we have that $\mathcal{M} \models \varphi$.

This semantics is sound and complete for LL.

Theorem 1.4 (Girard [Gir87])**.** *A* LL *formula is valid if and only if it is provable in* LL.

1.4 Linear temporal logic

We have seen that we can associate to each LK formula one of the two truth values t (for true) or f (for false). This universal aspect of truth is sometimes very poor, in the situations where we want to be more specific about the characteristics of truth. For instance, for a proposition such as "life is good", we may want to express that "Tomorrow, life is good" or "Alexis thinks that life is good". By adding expressiveness to classical logic in this way, we get what we call a **modal logic**, and expressions such as "Tomorrow" or "Amina thinks that" are called **modalities**. When the modal operators express properties about time, we call them **temporal modalities**. We can then add to classical logic modal operators to represent and reason about the temporal dimension. The obtained logics are particularly suited to describe the behaviour of **reactive systems**, which are open systems constantly in interaction with their environment. With temporal modalities, we can express for instance *accessibility* properties, such as "the system will eventually reach a good state", *safety* properties such as "The system will never be in a bad state" and many other properties such as *invariance* and *progress*.

The **Linear Temporal Logic** (LTL) is among the most studied temporal logics, we present its syntax in the following.

1.4.1 LTL formulas

Definition 1.17. Let $\mathcal{P} = \{\mathfrak{p}, \mathfrak{q}, \dots\}$ be a set of atoms. LTL ***formulas*** are given by:

$$\varphi, \psi := \mathfrak{p} \mid \mathfrak{p}^\perp \mid \bot \mid \top \mid \varphi \vee \psi \mid \varphi \wedge \psi \mid \odot\varphi \mid \mathbf{F}\varphi \mid \mathbf{G}\varphi \mid \varphi\mathbf{U}\psi \mid \varphi\mathbf{R}\psi$$

The negation of LTL formulas is the involution extending that of LK with the following equations:

$$(\odot\varphi)^\perp = \odot\varphi^\perp \qquad (\mathbf{F}\varphi)^\perp = \mathbf{G}\varphi^\perp \qquad (\varphi\mathbf{U}\psi)^\perp = \varphi^\perp\mathbf{R}\psi^\perp$$

Let us briefly outline the informal meaning of each temporal operator of LTL:

- $\odot\varphi$: φ in true in the next moment in time.
- $\mathbf{F}\varphi$: φ will eventually become true.
- $\mathbf{G}\varphi$: φ will always be true.
- $\psi\mathbf{U}\varphi$: ψ has to hold at least until φ, which holds at the current or a future position.
- $\psi\mathbf{R}\psi$: φ has to be true until and including the point where ψ first becomes true, if ψ never becomes true, φ must remain true forever.

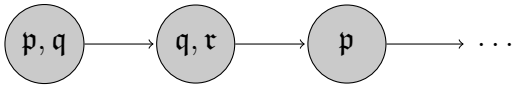
Further in this thesis, we will consider a fragment of LTL, which we call LK$\odot$. This logic extends LK only with the modality $\odot$. We introduce it below:

Definition 1.18. Let $\mathcal{P} = \{\mathfrak{p}, \mathfrak{q}, \dots\}$ be a set of atoms. LK$\odot$ ***formulas*** are given by:

$$\varphi, \psi := \mathfrak{p} \mid \mathfrak{p}^\perp \mid \bot \mid \top \mid \varphi \vee \psi \mid \varphi \wedge \psi \mid \odot\varphi$$

1.4.2 Semantics of LTL formulas

The models of LTL formulas are the ω-words over the alphabet $\Sigma = 2^{\mathcal{P}}$. Intuitively, every position of such a word corresponds to an instant of time, and a letter at some position represents the set of atoms which are true at the corresponding instant of time. In the example below, the atoms $\mathfrak{p}$, $\mathfrak{q}$ are true at instant 0, the atoms $\mathfrak{q}$, $\mathfrak{r}$ are true at instant 1, etc.



Definition 1.19. A ***model*** $\mathcal{M}$ is an infinite word over $\Sigma := 2^{\mathcal{P}}$, ie $\mathcal{M} \in \Sigma^\omega$. The ***interpretation*** $\|\varphi\|^{\mathcal{M}}$ of an LTL formula φ over a model $\mathcal{M}$ is a set of integers defined by induction on φ as follows:

$$\|\mathfrak{p}\|^u = \{i \in \omega \mid \mathfrak{p} \in u_i\} \qquad \|\varphi \vee \psi\|^u = \|\varphi\|^u \cup \|\psi\|^u \qquad \|\varphi^\perp\|^u = \omega \setminus \|\varphi\|^u$$

$$\begin{aligned}
\|\odot\varphi\|^u &= \{i \in \omega \mid i+1 \in \|\varphi\|^u_\rho\} \\
\|\mathbf{F}\varphi\|^u &= \{i \in \omega \mid \exists j \geq i,\ j \in \|\varphi\|^u\} \\
\|\psi\mathbf{U}\varphi\|^u &= \{i \in \omega \mid \exists j \geq i, j \in \|\varphi\|^u \wedge \forall i \leq k \leq j,\ k \in \|\psi\|^u\}
\end{aligned}$$

We say that a formula φ is ***true in a model*** $\mathcal{M}$ and we write $\mathcal{M} \models \varphi$ if $0 \in \|\varphi\|^{\mathcal{M}}$. A formula is said to be ***valid*** if $\forall \mathcal{M},\ \mathcal{M} \models \varphi$.

The interpretation of an LTL formula with respect to a model is the set of instants of time where it is true in this model. A formula is valid in a model, if it is true at the instant 0 of this model. For instance, the formula $\odot\mathfrak{r}$ is true in the previous model.

The semantics of LK can be seen as a particular case of the semantics of LTL. Indeed, every model of LK can be seen as a word of length 1, in other words, we have only one instant of time in LK.

1.4.3 Proof systems for LTL

The logic LTL does not have a well established proof theory. Many proof systems for LTL exist, but they do not fulfil the conditions prescribed to be satisfactory in a Curry-Howard approach: they either use a proof theoretic artifacts in the form of an ω-rule, have no cut-elimination or only for fragments of the logic [Bor09] or have cut-admissibility proved by semantic methods but without syntactic cut-elimination [BL08].

We introduce below the proof system LK$\odot$. This proof system enjoys all the expected good properties: soundness, completeness and cut-elimination. We will not establish formaly all these properties about it, since its main interest in this thesis will be to serve as nucleus for the futur extensions with fixed points.

Definition 1.20. LK$\odot$ is the proof system whose rules are those shown in Figure 1.3 together with the following rule:

$$\frac{\Delta \vdash \Sigma}{\Gamma, \odot\Delta \vdash \odot\Sigma, \Theta}\ (\odot)$$

The logic LK$\odot$ is obviously much weaker that LTL, since it can speak only about what could happen in a fixed number of instants of time. Further in this thesis, we will enrich LK$\odot$ formulas with least and greatest fixed points operators, which yields a logic called the **linear-time μ-calculus.** This enriched logic is not only much more expressive than LTL, but it will have a very beautiful proof theory.

1.5 On the shape of sequents

In the previous sections, we have presented our proof systems with sequents which are lists of formulas. This presentation is not the only one, and many other alternative definitions exist in the literature. We review some of them in the following.

1.5.1 Review on the shape of sequents

We discuss in the following some of the most used presentations of sequents:

1. **Sequents as lists of formulas:** This is the original definition of Gentzen and the one that we used in the previous sections. Its main drawback is its rigidity, since we have to use constantly the exchange rule. Since this original definition, other alternative presentations of sequents have been proposed to avoid the use of this rule.

2. **Sequents as multi-sets of formulas:** When we quotient the order of formulas in the previous presentation of sequents, we get sequents which are multi-sets of formulas. Doing so, we obviously do not need the exchange rule. If both presentations are perfectly equivalent from the provability point of view, they are not at the level of proofs. Indeed, from a Curry-Howard perspective where proofs correspond to programs, the multi-sets presentation of sequents induces a quotient on proofs, identifying some of them which correspond to programs having completely incomparable computational content. For instance, the following two proofs correspond respectively to the λ-terms $\lambda xy.x$ and $\lambda xy.y$, they will be nevertheless identified in the sequents as multi-sets presentation (we have annotated formulas with the variables x and y so that the correspondence with λ-terms is clearer).

$$\dfrac{\dfrac{}{A^x \vdash A}\ (\mathsf{Ax})}{A^x, A^y \vdash A}\ (\mathsf{W_l}) \qquad\qquad \dfrac{\dfrac{\dfrac{}{A^y \vdash A}\ (\mathsf{Ax})}{A^y, A^x \vdash A}\ (\mathsf{W_l})}{A^x, A^y \vdash A}\ (\mathsf{Ex_l})$$

   The same observations hold for the presentation of sequents as sets of formulas. Note that these two presentations do not require to adapt the inference rules, but only to interpret them differently, that is the contexts $\Gamma, \Delta, \ldots$ appearing in the rules should be thought of as multi-sets (resp. sets) of formulas, not as lists of formulas.

3. **Sequents as sets of named formulas:** In this presentation, we will also not have an ordering on the formulas of a sequent, but we will distinguish two *occurrences* of the same formula by giving them distinct names, sequents become then sets of named formulas. More precisely, given a set of names $\mathcal{N}$, a sequent is defined to be a set of pairs (φ, x) where φ is a formula and x a name. Every pair (φ, x), denoted φ_x, is called a ***named formula*** or a ***formula occurrence***. This presentation requires an adaptation of the inference rules. Identity rules should be applied when the underlying formulas are equal:

$$\dfrac{}{\vdash \varphi_x, \varphi_y^\perp}\ (\mathsf{Ax}) \qquad\qquad \dfrac{\vdash \varphi_x, \Gamma \qquad \vdash \varphi_y^\perp, \Delta}{\vdash \Gamma, \Delta}\ (\mathsf{Cut})$$

   And every inference rule of the following form, where φ is the principal formula of (r) and φ^j the sub-formulas of φ:

$$\dfrac{\{\vdash [\varphi^j]_{j\in I_i}, \Gamma_i\}_{i\in I}}{\vdash \varphi, \Gamma}\ (\mathsf{r})$$

   becomes as follows, where Γ and Γ_i are sets of formulas occurrences, and the names x^j should not appear in Γ_i and Δ_i:

$$\dfrac{\{\vdash \{\varphi^j_{x^j}\}_{j\in I_i}, \Delta_i\}_i}{\vdash \varphi_x, \Delta}\ (\mathsf{r}, x \to [[x^j]_j]_i)$$

The mention $(x \to [[x^j]_j]_i)$ that appears next to the name of the rule is here to remember that the causality link between the name x and the names x^j. This is essential to define unambiguously the cut-elimination steps. For instance, the following rules:

$$\frac{\vdash \varphi, \psi, \Gamma}{\vdash \varphi \parr \psi, \Gamma}\ (\parr) \qquad \frac{\vdash \varphi, \Gamma \quad \vdash \psi, \Gamma}{\vdash \varphi \,\&\, \psi, \Gamma}\ (\&)$$

become:

$$\frac{\vdash \varphi_y, \psi_z, \Gamma}{\vdash (\varphi \parr \psi)_x, \Gamma}\ (\parr, x \to [[y,z]]) \qquad \frac{\vdash \varphi_y, \Gamma \quad \vdash \psi_z, \Gamma}{\vdash (\varphi \,\&\, \psi)_x, \Gamma}\ (\&, x \to [[y],[z]])$$

This last presentation is equivalent to the original one that uses lists of formulas, at the level of provability as well as the level of proofs. It is the most commonly used since it correponds to the way proofs are annotated by variables, in order to obtain λ-terms, but it is usually treated in an implicit way. It will be also of special importance in our work. Indeed, if in finitary proof systems (that is proof systems in which proofs are finite trees) the well-formation of a proof is simply guaranteed by the well-application of the inference rules, this is not the case anymore for the infinitary proof systems we will deal with further in this thesis. More precisely, they rely on the future and the transformation it undergoes all along the proof.

This presentation, although useful, is heavy to manipulate. This is the reason why we will present in the next section a concrete implementation of it, by choosing an appropriate set of names. This will allow us to work with formula occurrences in an easy and transparent way.

1.5.2 Sequents as sets of formula occurrences

Our treatment of occurrences relies on the choice of a specific set of names, which is the set of words over the alphabet $\{l, r, i\}$.

Definition 1.21. Let Σ be the alphabet $\{l, r, i\}$. An ***address*** is a word over the alphabet Σ. The empty word is denoted by ε. A formula occurrence, we say simply ***occurrence***, is given by a formula φ and an address α, and written φ_α.

The letters l, r and i stands for "left", "right" and "inside" respectively. Whenever we will decompose a formula in an inference rule, the address of each of its subformulas will be extended by r if it its the right subformula, by l if it its the left subformula and by i if its is the immediate subformula when the connective of the original formula is unary.

The alphabet Σ is enough for our needs since we focus only on signatures having symbols of at most arity 2. But if we have more than binary symbols, the alphabet Σ should be enriched by as many directions as needed.

Convention 1.1. We reserve the symbols $\varphi, \psi, \ldots$ to formulas and $G, H, \ldots$ to occurrences.

To treat in a transparent way occurrences in our proof systems, we have to extend the operations on formulas to occurrences. We define in the following the negation of occurrences and show how to apply the connectives of the logic to occurrences.

Definition 1.22. We define a ***duality*** over Σ^* as the morphism $(\bullet)^\perp$ satisfying:

$$l^\perp = r, \qquad r^\perp = l \qquad \text{and} \qquad i^\perp = i.$$

Negation of occurrences is defined by:

$$(\varphi_\alpha)^\perp = (\varphi^\perp)_{\alpha^\perp}$$

Connectives are extended to ***occurrences*** as follows:

- For any binary symbol $\star$, we set $F \star G = (\varphi \star \psi)_\alpha$ if $F = \varphi_{\alpha l}$ and $G = \psi_{\alpha r}$.
- For any unary symbol $\triangle$ we set $\triangle\varphi = (\triangle\varphi)_\alpha$ if $\varphi = \varphi_{\alpha i}$.

The inference rules seen in earlier sections can be interpreted in this setting of sequents as sets of occurrences without any need of adaptation. For instance, in the following rules, instead of seeing $F \parr G, \Gamma$ as a list of formulas whose head is the formula $F \parr G$, we can see it as a set containing an occurrence of the form $F \parr G$, etc.

$$\frac{\vdash F, G, \Gamma}{\vdash F \parr G, \Gamma}\ (\parr) \qquad\qquad \frac{\vdash F, \Gamma \qquad \vdash G, \Gamma}{\vdash F \,\&\, G, \Gamma}\ (\&)$$

These two rules actually denote the following rules, where we explicited the addresses of the occurrences F and G by setting $F = \varphi_{\alpha l}$ and $G = \psi_{\alpha r}$:

$$\frac{\vdash \varphi_{\alpha l}, \psi_{\alpha r}, \Gamma}{\vdash (\varphi \parr \psi)_\alpha, \Gamma}\ (\parr) \qquad\qquad \frac{\vdash \varphi_{\alpha l}, \Gamma \qquad \vdash \psi_{\alpha r}, \Gamma}{\vdash (\varphi \parr \psi)_\alpha, \Gamma}\ (\&)$$

The cut rule stays also unchanged since we have defined the operation of negation on occurrences. We need to be more specific about the axiom rule, since we want to apply it to occurrences as soon as their underlying formulas are equal, but not necessarily their addresses. For that we introduce the following useful notation.

Definition 1.23. Two occurrences $F = \varphi_\alpha$ and $G = \psi_\beta$ are said to be ***structurally equivalent***, and we write $F \equiv G$ if $\varphi = \psi$.

Axiom rules have to be exchanged against these two "up-to relocating" versions:

$$\frac{F \equiv G}{F \vdash G}\ (\mathsf{Ax}) \qquad \frac{F \equiv G}{\vdash F, G^\perp}\ (\mathsf{Ax}) \qquad \frac{F \equiv G}{F, G^\perp \vdash}\ (\mathsf{Ax})$$

The idea behind the use of named formulas is to guarantee that every formula appearing in a proof can be traced back in a unique way to a conclusion formula or to a cut formula. In other words, we do not want two occurrences to generate the same occurrence. This can happen if the addresses of the occurrences appearing in a sequent are extensions one of the other, as for the following example, where the occurrence φ_l of the premise comes both from the occurrence $(\varphi \oplus \top)_\varepsilon$ and the occurrence φ_l of the conclusion:

$$\frac{\vdash \varphi_l}{\vdash (\varphi \oplus \top)_\varepsilon, \varphi_l}\ (\oplus_1)$$

To prevent that, we impose a disjointness condition on our sequents.

Definition 1.24. If α, α' are two addresses such that α is a prefix of α', then we write $\alpha \sqsubseteq \alpha'$ and we say that α' is a ***sub-address*** of α. We say that two ***addresses*** α and β are ***disjoint*** if $\alpha \not\sqsubseteq \beta$ and $\beta \not\sqsubseteq \alpha$. We say that two ***occurrences*** are ***disjoint*** when their addresses are. A ***sequent*** is a pair of two sets of pairwise disjoint occurrences.

Proviso 1.1. From now on, the sequents of the inference rules of LK, $\mathsf{LK}_{\mathsf{rev}}$, $\mathsf{LK}_{\mathsf{os}}$, MALL, LL and $\mathsf{LK}\odot$ are seen as (pairs of two) sets of pairwise disjoint occurrences, and the axiom rule is replaced by the axiom rule up to renaming.

Example 1.2. We show in the following an example of a $\mathsf{LK}_{\mathsf{os}}$ proof with our formalism of sequents, where α is an address.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash \mathfrak{p}_{\alpha ll}, \mathfrak{p}^{\perp}_{\alpha rr}}\,(\mathsf{Ax})}{\vdash \mathfrak{p}_{\alpha ll}, \mathfrak{q}_{\alpha rl}, \mathfrak{p}^{\perp}_{\alpha rr}}\,(\mathsf{W}) \qquad \dfrac{\dfrac{}{\vdash \mathfrak{q}^{\perp}_{\alpha lr}, \mathfrak{q}_{\alpha rl}}\,(\mathsf{Ax})}{\vdash \mathfrak{q}^{\perp}_{\alpha lr}, \mathfrak{q}_{\alpha rl}, \mathfrak{p}^{\perp}_{\alpha rr}}\,(\mathsf{W})}{\vdash (\mathfrak{p} \wedge \mathfrak{q}^{\perp})_{\alpha l}, \mathfrak{q}_{\alpha rl}, \mathfrak{p}^{\perp}_{\alpha rr}}\,(\wedge)}{\vdash (\mathfrak{p} \wedge \mathfrak{q}^{\perp})_{\alpha l}, (\mathfrak{q} \vee \mathfrak{p}^{\perp})_{\alpha r}}\,(\vee)}{\vdash ((\mathfrak{p} \wedge \mathfrak{q}^{\perp}) \vee (\mathfrak{q} \vee \mathfrak{p}^{\perp}))_{\alpha}}\,(\vee)$$

We show an other example in MALL, where α and $\beta^{\perp}$ are two disjoint addresses:

$$\dfrac{\dfrac{\dfrac{}{\vdash \mathfrak{p}_{\beta l}, \mathfrak{p}^{\perp}_{r\beta}}\,(\mathsf{Ax})}{\vdash (\mathfrak{p} \parr \mathfrak{p}^{\perp})_{\beta}}\,(\parr) \qquad \dfrac{}{\vdash (\mathfrak{p} \parr \mathfrak{p}^{\perp})_{\alpha}, (\mathfrak{p} \parr \mathfrak{p}^{\perp})^{\perp}_{\beta^{\perp}}}\,(\mathsf{Ax})}{\vdash (\mathfrak{p} \parr \mathfrak{p}^{\perp})_{\alpha}}\,(\mathsf{Cut})$$

Remark 1.1. Note that all the addresses appearing in a proof are the sub-addresses of the addresses of the conclusion occurrences or the cut occurrences.

We state in the following an easy condition that guarantees the disjointness condition in sequents. If not otherwise stated, we assume that this condition is satisfied.

Proviso 1.2. All along this thesis, we suppose that the addresses of the conclusion occurrences together with the addresses of cut occurrences are all pairwise disjoint.

By Remark 1.1, every proof satisfying the proviso satisfies automatically the disjointness condition on sequents. Proviso 1.2 is always achievable, since we have an infinite supply of disjoint addresses, for instance $\{\ r^n l\ :\ n > 0\ \}$. One may thus pick addresses from that supply for the conclusion sequent of the derivation, and then carry the remaining supply along proof branches, splitting it on branching rules, and consuming a new address for cut rules. Clearly, every proof satisfying the proviso satisfies also the disjointness condition on sequents.

This proviso is practical and easy to state, but it is sometimes too rigid. In particular, it is not always stable by the transformations that we will perform on our proofs, for example cut-elimination and focalization. In these specific situations, we will relax it into a condition which is preserved by these transformations.

We introduce the last definition concerning occurrences, which is the operation of substitution. It will be helpful to keep the same transparency when we will deal with the rules of fixed points, since they use substitution.

Definition 1.25. ***Substitution of occurrences*** is defined as follows:

$$(\varphi_\alpha)[\psi_\beta/X] = (\varphi[\psi/X])_\alpha$$

We end up this section by a last convention, which allows us to be flexible in our use of occurrences.

Definition 1.26. Let $F = \varphi_\alpha$ be a formula occurrence and β be an address. We define F_β to be $F_\beta := \varphi_\beta$. We say that we ***relocated*** F in β. We define $\overline{F}$ to be $\overline{F} := \varphi$. The function $\overline{\cdot}$ is called the ***forgetful function***.

Convention 1.2. All along this thesis, whenever we define a notion $N(F)$ over occurrences, if we write $N(\varphi)$ one should read $N(\varphi_\varepsilon)$. Conversely, if we define a notion $N(\varphi)$ over formulas, if we write $N(F)$ one should read $N(\overline{F})$.

Chapter 2

Fixed-points in proof theory

This chapter is dedicated to the extension of various logics and the associated proof systems with fixed points.

We start in Section 2.1 by recalling some background on lattice theory and the fundamental fixed point theorems of the literature. These theoretical results are important for two reasons. On the one hand, we will use them several times in our technical proofs. On the other hand, they will give us good intuitions about fixed point, which will guide the design of proof systems for fixed points.

The first step towards conceiving these proof systems is to extend the syntax of a logic with fixed points statements. We show how to perform such extensions in Section 2.2. The formulas of those logics enjoy, beside the usual notion of subformula, another form of subformula called Fischer-Ladner subformulas. We discuss and compare these two notions in Section 2.3.

In Section 2.4, we show how to enrich a proof system S with rules for fixed points. We focus on two approaches. The first one consists in using Park's rules, which are rules directly inspired by the (co)-induction principle. The obtained proof system is called $\mu\mathsf{S}$ and its proofs are finite trees, just as the proof system we have seen in Chapter 1. The second approach is inspired by the infinite descent proof technique. It consists in adding to S the unfolding rules for fixed points and allowing infinite derivations. To get a sound proof system, these derivations are equipped with a validity condition, and the infinite trees satisfying it are called infinite proofs. The obtained proof system is denoted by $\mu\mathsf{S}^\infty$. If an infinite proof is regular, we can represent it by a finite graph. We call these graphs circular proofs, and the corresponding proof system $\mu\mathsf{S}^\omega$, the circular proof system. We relate the finitary and the circular proof systems at the end of the chapter, by showing that every $\mu\mathsf{S}$ proof can be translated into a $\mu\mathsf{S}^\omega$ proof. Conversely, we show that if a $\mu\mathsf{S}^\omega$ proof satisfies a condition that we call the translatability criterion, we can translate it into a $\mu\mathsf{S}$ proof.

2.1 Fixed points theorems

The least and greatest fixed points of an operator F over a set E do not always exist. The **Knaster-Tarski theorem** states a condition on the set E (being a complete lattice) and the operator F (monotonicity), so that the existence of extremal fixed points is guaranteed. We introduce these notions and the Knaster-Tarski theorem in the following.

Definition 2.1. A ***complete lattice*** is given by a set U and a partial order relation $\leq$ on U, such that every subset A of U has a ***supremum*** (meet) and an ***infimum*** (join) with respect to $\leq$, denoted respectively $\bigvee A$ and $\bigwedge A$.

Taking the meet and the join of the empty set, one gets the ***bottom*** and ***top*** elements of the lattice, that is the least and greatest elements of the lattice.

Example 2.1. A common example of a complete lattice is the powerset 2^X of a set X, that is the set of all subsets of X, ordered by the inclusion relation $\subseteq$. The supremum of a set $\mathcal{C}$ of subsets of X is their union $\cup\mathcal{C}$ and the infimum of $\mathcal{C}$ is their intersection $\cap\mathcal{C}$.

Definition 2.2. Let $(U, \leq)$ be a complete lattice and $F : U \to U$ an operator on U.

- The operator F is ***monotonic*** if $\forall x, y \in U$, $x \leq y$ implies $F(x) \leq F(y)$.
- An element $x \in U$ is a ***pre-fixed point*** of F if $F(x) \leq x$.
- An element $x \in U$ is a ***post-fixed point*** of F if $x \leq F(x)$.
- An element $x \in U$ is a ***fixed point*** of F if $F(x) = x$.

Theorem 2.1 (Knaster-Tarski [Tar55])**.** *Let $(U, \leq)$ be a complete lattice and F a monotonic operator on it. The set of fixed points of F, equipped with $\leq$ forms a complete lattice. Moreover, the least element of this set, called the* **least fixed point of** *F and denoted $\mathsf{lfp}(F)$, is the meet of the set of pre-fixed points of F.*

$$\mathsf{lfp}(F) = \bigcap \{S \mid F(S) \leq S\}$$

Dually, the greatest element of the set of fixed points of F, called the **greatest fixed point of** *F and denoted $\mathsf{gfp}(F)$, is the join of the set of post-fixed points of F.*

$$\mathsf{gfp}(F) = \bigcup \{S \mid S \leq F(S)\}$$

The Knaster-Tarski theorem guarantees the existence of extremal fixed points, but it does not give any concrete way to construct them. For that, we usually use a more "constructive" theorem, due to Kleene [Kle52], where we relax the condition on the order relation, and strengthen that of the operator.

Definition 2.3. Let $(U, \leq)$ be an ordered set. A subset S of U is said to be ***directed*** if it is nonempty and if for every elements a and b in S there is a c in S such that $a \leq c$ and $b \leq c$. We say that $(U, \leq)$ is a ***complete partial order (cpo)*** if each of its directed subsets has a supremum.

Definition 2.4. Let $(U, \leq)$ be an ordered set and F an operator on U. We say that F is ***Scott-continuous*** if it preserves all directed suprema, that is if for every directed subset S of U with supremum in U its image has a supremum in U, and that supremum is the image of the supremum of S.

Remark 2.1. Every Scott-continuous operator is monotonic.

Theorem 2.2 (Kleene [Kle52]). *Let $(U, \leq)$ be a cpo and F a Scott-continuous operator on U. Let $\bot$ be the bottom of U.*

The **ascending Kleene chain** *of F is the chain*

$$\bot \leq F(\bot) \leq F(F(\bot)) \leq \cdots \leq F^n(\bot) \leq \ldots$$

obtained by iterating F starting from $\bot$. The least fixed point of F is obtained by taking the meet of this chain:

$$\mathsf{lfp}(F) = \bigvee_{n \in \omega} F^n(\bot)$$

In practice, Scott-continuity is not always guaranteed or it is hard to prove. We present below a theorem due to Cousot and Cousot [CC79] which has the same flavour as Kleene's theorem, but with the hypothesis of Knaster-Tarski one. The idea is that, to obtain the least fixed point of a monotonic operator on a complete lattice, one should iterate it starting from $\bot$, not always ω times as it is the case for Kleene's theorem, but up to an ordinal which is "big enough" for the lattice.

Definition 2.5. Let U be a set. The ***ordinal associated*** to U, λ_U, is the least ordinal whose cardinality is strictly greater than the cardinality $\mathsf{card}(U)$.

Definition 2.6. Let $(U, \leq)$ be a complete lattice and F a monotone operator on it. We define the ***ascending iterations*** $F^{\uparrow\delta}(\bot)$ of F starting from $\bot$ as follows:

- $F^{\uparrow 0}(\bot) = \bot$;
- $F^{\uparrow\delta}(\bot) = F(F^{\uparrow\lambda}(\bot))$ for every successor ordinal $\delta = \lambda + 1$;
- $F^{\uparrow\delta}(\bot) = \bigvee_{\lambda<\delta} F^{\uparrow\lambda}(\bot)$ for every limit ordinal δ.

Dually, we define the ***descending iterations*** $F^{\downarrow\delta}(\top)$ of iterations of F starting from $\top$ as follows:

- $F^{\downarrow 0}(\top) = \top$;
- $F^{\downarrow\delta}(\top) = F(F^{\downarrow\lambda}(\top))$ for every successor ordinal $\delta = \lambda + 1$;
- $F^{\downarrow\delta}(\top) = \bigwedge_{\lambda<\delta} F^{\downarrow\lambda}(\top)$ for every limit ordinal δ.

Theorem 2.3 (Cousot and Cousot [CC79]). *Let $(U, \leq)$ be a complete lattice and F be a monotonic operator on U. The sequence $(F^{\uparrow\delta}(\bot))_{\delta\in\lambda_U}$ is a stationary increasing chain, its limit $f^{\uparrow\lambda_U}(\top)$ is the least fixed point of F. Dually, the sequence $(F^{\downarrow\delta}(\top))_{\delta\in\lambda_U}$ is a stationary decreasing chain, its limit $F^{\downarrow\lambda_U}(\top)$ is the greatest fixed point of F.*

2.2 Formulas with fixed points

In this section we will see how to extend the formulas of a given logic with least and greatest fixed points. This requires to consider a set of variables $\mathcal{V}$ that will help to construct these fixed points statements.

Definition 2.7. Let $\mathcal{L} = (S, ar)$ be a signature. The ***signature*** $\mu\mathcal{L}$ is defined as follows:

- The symbols of $\mu\mathcal{L}$ are $S \cup \{\mu X., \nu X. | X \in \mathcal{V}\} \cup \{X, X^\perp | X \in \mathcal{V}\}$.
- The arity function extends that of $\mathcal{L}$ by setting:

$$ar(\mu X.) = ar(\nu X.) = 1 \text{ and } ar(X) = ar(X^\perp) = 0.$$

Intuitively, the formula $\mu X.\varphi$ denotes the least fixed point of the operator "$X \mapsto \varphi(X)$", and the formula $\nu X.\varphi$ denotes the greatest fixed point of this operator.

We usually write $\sigma X.\varphi$ when we do not want to be specific about the nature of the fixed point.

The symbols $\mu X.$ and $\nu X.$ are variable binders. The notions of free variables, bound variables and capture free substitution are as usual.

In the Knaster-Tarski theorem, the monotonicity of the operator is crucial. We reflect this semantical condition by a syntactic constraint on formulas as follows.

Definition 2.8. Let $\mu\mathcal{L}$ be a signature. A formula of $\mu\mathcal{L}$ is said to be ***monotonic*** if no negation of a variable $X^\perp$ is under the scope of a binder μX or νX.

Note that the abscence of implication in the logics we are considering is important for this definition of monotonicity.

We will show in further semantical investigations (Chapter 5, Chapter 7), that this syntactic condition of monotonicity indeed guaranties the monotonicity of the operator that will be associated to the formula. In the rest of this thesis, we work only with monotonic formulas and simply call them formulas.

In all the logics presented in Chapter 1 (LK, LL, MALL, $\mathsf{LK}\odot$), we defined negation as an involution satisfying a number of equations. We extend these definitions in the presence of fixed points as follows.

Definition 2.9. Let $\mathcal{L}$ be a signature in $\{\mathsf{LK}, \mathsf{LL}, \mathsf{MALL}, \mathsf{LK}\odot\}$. ***Negation*** is the involution on formulas written $\varphi^\perp$ and satisfying the conditions of Chapter 1 together with:

$$(\mu X.F)^\perp = \nu X.F^\perp \qquad (X)^\perp = X$$

Example 2.2. Negation of the $\mu\mathsf{LK}\odot$ formula $\mu X.(X \vee \odot Y^\perp) \wedge \mathfrak{p}$ is $\nu X.(X \wedge \odot Y^\perp) \vee \mathfrak{p}^\perp$.

Remark 2.2. Notice that negation does not dualize the fixed point variables. Doing so, the negation of a (monotonic) formula is a (monotonic) formula.

Remark 2.3. This definition of negation is practical when we are concerned only with closed formulas, which will be the case most of the time in this thesis. If we are also interested by open formulas, the two equations of Definition 2.9 should be replaced by the following one:

$$(\mu X.F)^\perp = \nu X.F^\perp[X/X^\perp]$$

This alternative definition dualizes only free variables, and keeps unchanged the bound ones.

2.3 On the notion of subformula

Formulas with fixed points support two notions of subformula. The first notion is the usual one: a formula ψ is the subformula of φ, and we write $\varphi \leq \psi$, if the syntactic tree of ψ is a sub-tree of the syntactic tree of φ. In general, the subformula of a closed formula may contain free variables. The second one is specific to formulas with fixed points, it is a sort of subformula up to unfolding, so that the subformulas of a closed formula *w.r.t.* this notion are also closed, we call it **Fischer-Ladner** subformula. In the following, we introduce these two kinds of subformula, taking advantage of the notion of occurrence.

2.3.1 The usual notion of subformula

We first define the syntactic tree of a formula, then we introduce the *sub-occurrences* of a formula. This notion is finer-grained than that of a sub-formula. For instance, the variable X is a sub-formula of $\varphi = (\nu X.X) \otimes X$, but in φ there are two appearences or *occurrences* of this variable, one is bound and the other is free in φ. We want to be able to distinguish these two occurrences of X, and the notion of occurrence, introduced in Definition 1.21 is well-suited for that purpose.

Definition 2.10. The ***tree of a formula*** φ is the labelled tree $\tau(\varphi)$ defined inductively as follows, where $\star$ is a binary symbol, $\triangle$ is a unary symbol and $\circ$ is a nullary symbol.

$$\tau(\varphi \star \psi) = \begin{array}{ccc} & \star & \\ {\scriptstyle l}\swarrow & & \searrow{\scriptstyle r} \\ \tau(\varphi) & & \tau(\psi) \end{array} \qquad \tau(\triangle\varphi) = \begin{array}{c} \triangle \\ \downarrow i \\ \tau(\varphi) \end{array} \qquad \tau(\circ) = \circ$$

Example 2.3. The trees of the μMALL formula $\Phi = (\nu X.X) \otimes X$ and the μLK formula $\Psi = \mu X.\nu Y.\, \odot X \wedge \odot Y$ are the following:

$$\tau(\Phi) = \begin{array}{ccc} & \otimes & \\ {\scriptstyle l}\swarrow & & \searrow{\scriptstyle r} \\ \nu X. & & X \\ \downarrow i & & \\ X & & \end{array} \qquad \tau(\Psi) = \begin{array}{c} \mu X. \\ \downarrow i \\ \nu Y. \\ \downarrow i \\ \wedge \\ {\scriptstyle l}\swarrow \quad \searrow{\scriptstyle r} \\ \odot \qquad \odot \\ \downarrow i \qquad \downarrow i \\ X \qquad Y \end{array}$$

We will use Φ and Ψ in the upcoming examples of this section without recalling their definitions.

Definition 2.11. We define the ***relation*** $\rightharpoonup$ on occurrences as follows, where $\star$ and Δ are respectively a binary and a nullary symbols:

$$\begin{aligned}(\varphi \star \psi)_\alpha &\rightharpoonup \varphi_{\alpha l}\\ (\varphi \star \psi)_\alpha &\rightharpoonup \psi_{\alpha r}\\ (\triangle\varphi)_\alpha &\rightharpoonup \varphi_{\alpha i}\end{aligned}$$

A ***sub-occurrence*** of an occurrence F is any occurrence G such that $F \rightharpoonup^* G$, where $\rightharpoonup^*$ is the reflexive transitive closure of $\rightharpoonup$. The ***sub-formulas*** of an occurrence F are the formulas obtained by forgetting the addresses of the sub-occurrences of F. We denote by $\leq$ the ***subformula ordering*** on formulas, that is $\psi \leq \varphi$ if the formula ψ is a subformula of φ. We denote by $Sub(F)$ (resp. $Sub_o(F)$) the set of sub-formulas (resp. sub-occurrences) of F.

We use the expression "ψ_β is an occurrence of ψ in φ_α" to say that ψ_β is a sub-occurrence of φ_α.

Example 2.4. We have $\Phi_\varepsilon \rightharpoonup X_r$ and $\Phi_\varepsilon \rightharpoonup (\nu X.X)_l \rightharpoonup X_{li}$. In particular Φ_ε has two occurrences of the variable X which are X_r and X_{li}, the first one is free the other is bound.

Proposition 2.1. *Let $F = \varphi_\alpha$ be an occurrence. The following holds:*

i) If ψ_β is a sub-occurrence of F, then $\beta = \alpha.p$, where p is a path in $\tau(\varphi)$ from the root to some node n.

ii) The sub-tree of $\tau(\varphi)$ rooted in n is $\tau(\psi)$. As a consequence, any sub-occurrence G of F can be mapped in a unique way to a node of $\tau(\varphi)$, we denote it $N_F(G)$.

iii) Conversely, any node n of $\tau(\varphi)$ defines a unique sub-occurrence ψ_β of F: the subtree rooted in n is the tree of the formula ψ and $\beta = \alpha.p$ where p is the path from the root of $\tau(\varphi)$ to the node n. We denote by $S_F(n)$ the occurrence ψ_β.

iv) One has $S_F(N_F(G)) = G$ and $N_F(S_F(n)) = n$.

Remark 2.4. As a consequence of Proposition 2.1, for any occurrence F, the sets $Sub(F)$ and $Sub_o(F)$ are finite.

Definition 2.12. Let F be an occurrence and let X_β be a sub-occurrence of F. We say that X_α is ***a free*** occurrence of X in F if the path between the root of $\tau(F)$ and $N_F(X_\alpha)$ does not contain a node labelled σX. It is called ***a bound*** occurrence otherwise. We denote by $\mathsf{fv}_o(F)$ the set of free variable occurrences in F.

Let X_α be a bound occurrence of X in F, we call the ***binder*** of X_β the closest node of $\tau(F)$ to $N_F(X_\alpha)$, which is labelled σX. If n is the binder of X_α, we call ***definition of*** X_α the occurrence $S_F(n)$ and we denote it $D_F(X_\alpha)$.

Remark 2.5. The underlying formula of $D_F(X_\alpha)$ is of the form $\sigma X.\psi$.

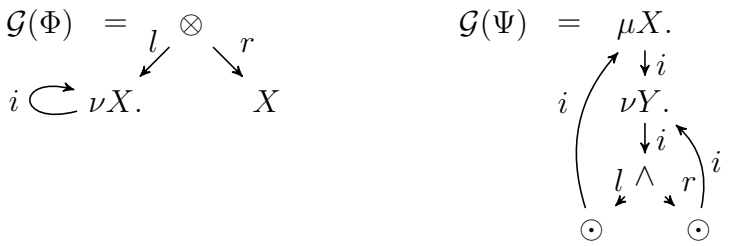
Example 2.5. The definition of the variable occurrence X_{li} in Φ_ε is $(\nu X.X)_l$. The definition of X_{iili} in Ψ_ε is Ψ_ε and that of Y_{iiri} is $(\nu Y.\odot X \wedge \odot Y)_i$

2.3.2 Fischer-Ladner subformulas

We have seen that the usual notion of sub-formula is obtained by traversing the tree of a formula. In the same way, the notion of Fischer-Ladner subformula can be obtained by traversing the graph of the formula, that is the graph obtained by identifying the nodes of bound variable occurrences with their binders.

Definition 2.13. The ***graph of a formula*** φ, denoted $\mathcal{G}(\varphi)$, is the graph obtained from $\tau(\varphi)$ by identifying the nodes of bound variable occurrences with their binders.

Example 2.6. The graphs of the formulas Φ and Ψ of example 2.3 are the following:



Remark 2.6. The graph $\mathcal{G}(\varphi)$ has the structure of a tree with "back-edges".

Terminology 2.1. When we speak about the relative positions of two nodes of $\mathcal{G}(\varphi)$ (above, below, closer to the root, ...) it is relatively to its underlying tree structure. For example, a node n is above a node m in $\mathcal{G}(F)$, if n is the ancestor of m in $\tau(\varphi)$.

Definition 2.14. We define the ***relation*** $\rightarrow$ on occurrences as follows:

$$\begin{array}{lcll} (\varphi \star \psi)_\alpha & \rightarrow & \varphi_{\alpha l} & \\ (\varphi \star \psi)_\alpha & \rightarrow & \psi_{\alpha r} & \\ (\triangle\varphi)_\alpha & \rightarrow & \varphi_{\alpha i} & \text{if } \triangle \neq \sigma X \\ (\sigma X.\varphi)_\alpha & \rightarrow & (\varphi[\sigma X.\varphi/X])_{\alpha i} & \end{array}$$

A FL-***suboccurrence*** of an occurrence F is any occurrence G such that $F \rightarrow^* G$, where $\rightarrow^*$ is the reflexive transitive closure of $\rightarrow$. The FL-***subformulas*** of an occurrence F are the formulas obtained by forgetting the addresses of the FL-suboccurrences of F. The ***Fischer-Ladner closure*** of an occurrence, denoted $\mathsf{FL}(F)$, is the set of its FL-subformulas.

Example 2.7. We set $\Theta = \nu Y.\, \odot\, \Psi \vee \odot Y$. One has:

$$\Psi_\varepsilon \rightarrow \Theta_i \rightarrow (\odot\Psi \vee \odot\Theta)_{ii} \rightarrow (\odot\Psi)_{iil} \rightarrow \Psi_{iili} \rightarrow \Theta_{iilii} \rightarrow (\odot\Psi \vee \odot\Theta)_{iiliii}$$

The Fischer-Ladner closure of Ψ is $\{\Psi, \Theta, \odot\Psi \vee \odot\Theta, \odot\Theta, \odot\Psi\}$. We use Θ in the next examples without recalling its definition.

Proposition 2.2. *Let φ_α be an occurrence. If ψ_β is a* FL*-suboccurrence of φ_α, then $\beta = \alpha.p$, where p is a path in $\mathcal{G}(\varphi)$ from the root to some node n, we denote this node by $\mathcal{N}_F(\psi_\beta)$.*

Example 2.8. The node of the FL-suboccurrence $(\odot\Psi \vee \odot\Theta)_{iliiii}$ of Ψ_ε is the node labelled $\wedge$ in $\mathcal{G}(\Psi)$.

We can relate the two notions of subformula and FL-subformula as follows: for every FL-subformula G of F, the node of G in $\mathcal{G}(F)$ is also a node of $\tau(F)$, and we have seen that we can associate to every node of $\tau(F)$ a sub-occurrence H of F. In this way, we can associate to every FL-subformula G of F the subformula H, which will be called the *body* of G.

Definition 2.15. Let G be a FL-suboccurrence of F. The node $n = \mathcal{N}_F(G)$ is also a node of $\tau(F)$. The ***body*** of G, denoted $\mathsf{Body}_F(G)$, is the occurrence $S_F(n)$, that is the suboccurrence corresponding to the node n of $\tau(F)$.

Example 2.9. The body of $(\odot\Psi \wedge \odot\Theta)_{iiliii}$ is the occurrence $(\odot X \wedge \odot Y)_{ii}$ because the sub-occurrence corresponding to the node $\wedge$ in $\tau(\Psi)$ is $(\odot X \wedge \odot Y)_{ii}$.

Definition 2.16. Let F be an occurrence and G be a sub-occurrence of F. The ***recursive substitution*** of G, denoted by $[\![G]\!]_F$, is defined inductively as follows:

$$[\![G]\!]_F = G[[\![D_F(X_\alpha)]\!]_F/X]_{X_\alpha \in fv_o(G)}$$

The recursive substitution takes a sub-occurrence G and replaces its free variables by their definitions, that is to say the subformulas of F that bind them, but these subformulas contain potentially free variables themselves, then the recursive substitution replaces these free variables by their definitions, and it continues this way. This definition is well-formed because it is inductive on the distance of $N_F(G)$ to the root: clearly the node of the definition of a free variable X_α of G, $N_F(D_F(X_\alpha))$, is closer to the root than $N_F(G)$ because $N_F(D_F(X_\alpha))$ is in the path between the root and $N_F(G)$.

Example 2.10. The recursive substitution of the sub-occurrence $(\odot Y)_{iir}$ is $(\odot\Theta)_{iir}$. Indeed, the definition of Y_{iiri} is $(\nu Y.(\odot X) \vee (\odot Y))_i$ which contains in its turn the free variable X_{iili}. The definition of X_{iili} is Ψ_ε, which is closed. Hence the recursive substitution of $(\nu Y.(\odot X) \vee (\odot Y))_i$ is Θ.

The following proposition shows the exact relation between a FL-suboccurrence G of F and its body: when we perform a recursive substitution on the body of G, we get exactly G, up to relocation.

Proposition 2.3. *Let G be a FL-suboccurrence of an occurrence F. We have that:*

$$[\![\mathsf{Body}_F(G)]\!]_F \equiv G$$

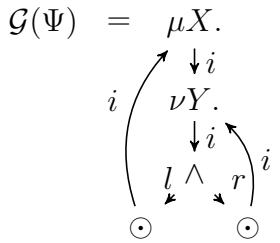
Proof. The proof is by induction on the distance from the node $\mathcal{N}_F(G)$ to the root of $\mathcal{G}(F)$. If $\mathcal{N}_F(G)$ is the root itself, then the equality is trivial since $\mathsf{Body}_F(G) = F$ and $[\![F]\!]_F = F$. Suppose that we have proved the property for G and suppose that $G = H \star K$. If the node $n_H := \mathcal{N}_F(H)$ is above $n_G := \mathcal{N}_F(G)$, that is the edge (n_G, n_H) is a back-edge, then the property is already proved for H since n_H is closer to the root than n_G. Otherwise, ie. when n_H is under n_G, this means that $\mathsf{Body}_F(G)$ is of the form $\mathsf{Body}_F(G) = \mathsf{Body}_F(H) \star T$ (notice that we do not have necessarily $\mathsf{Body}_F(G) = \mathsf{Body}_F(H) \star \mathsf{Body}_F(K)$ since the node of K in $\mathcal{G}(K)$ may be above n_G, hence T is some variable occurrence Y_γ and $\mathsf{Body}_F(K)$ is the definition of Y_γ). One then has $[\![\mathsf{Body}_F(G)]\!]_F = [\![\mathsf{Body}_F(H)]\!]_F \star [\![T]\!]_F$. By induction hypothesis one has $[\![\mathsf{Body}_F(G)]\!]_F \equiv G$, then $[\![\mathsf{Body}_F(H)]\!]_F \star [\![T]\!]_F \equiv H \star K$, hence $[\![\mathsf{Body}_F(H)]\!]_F \equiv H$. The same reasoning applies for unary and nullary symbols. □

Since $\mathsf{Body}_F(G)$ ranges over sub-occurrences of F, which is a finite set, a consequence of Proposition 2.3 is the following corollary:

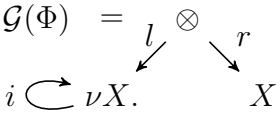
Corollary 2.1. *The Fischer-Ladner closure of a formula is finite.*

2.3.3 Comparing FL-subformulas

Further in this thesis, we shall compare the subformulas or the FL-subformulas of a given formula *w.r.t.* the subformula ordering $\leq$. For the first problem, that is comparing two elements G, H of $Sub(F)$ *w.r.t.* $\leq$, one can rely on some graphical observations: if the node of G is under the node of H in the tree of F, one can conclude that $G \leq H$. Since the elements of $\mathsf{FL}(F)$ correspond to the nodes of the graph of F, one may wonder if the same property happens for the FL-subformulas of F, but this is not the case. Indeed, in the graph of the formula Ψ of Example 2.6 shown below, the node of the FL-suboccurrence Θ_i, which is labelled $\nu Y.$, is under the node of Ψ_ε which is the root, however, $\Psi \leq \Theta$.



One cannot even say that $\leq$ and the graph ordering are inverted, indeed, in the graph of the formula Φ, the node of $(\nu X.X)_i$, which is the node labelled $\nu X.$ is under the node of Φ_ε which is the root, and one has $\nu X.X \leq \Phi$.

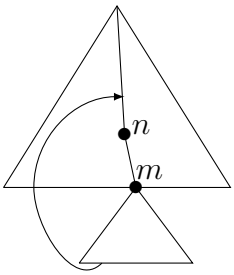


In Proposition 2.4, given two FL-suboccurrences G and H of F, we show some situations when we can assert that $G \leq H$ or $H \leq G$ using only the relative positions of the nodes of H and G in the graph of F.

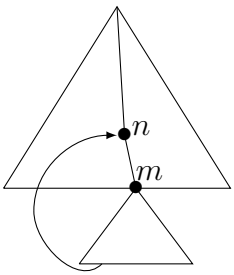
Terminology 2.2. In the following, if n is a node in $\mathcal{G}(F)$, we use the expression *sub-tree of* n to speak about the sub-tree of $\tau(F)$ rooted in n, from which the nodes labelled with variables are deleted. This sub-tree is obviously also a sub-tree of $\mathcal{G}(F)$. For instance, the sub-tree of the node labelled $\wedge$ in $\mathcal{G}(\Psi)$ in the above example is the sub-graph of $\mathcal{G}(\Psi)$ containing the node labelled $\wedge$ and the two nodes labelled $\odot$.

Proposition 2.4. *Let G, H be two FL-suboccurrences of F, let $n = \mathcal{N}_F(G)$ and $m = \mathcal{N}_F(H)$ be their respective corresponding nodes in $\mathcal{G}(F)$. Suppose that m is under n.*

- *If there is no back-edge going from the sub-tree of m to a node between n and m, including n (in other words, all the back-edges starting from the subtree of m can only stay in this subtree or point to nodes strictly above n), then one has $H \leq G$.*



- *There is a back edge starting from the subtree of m and pointing to the node n. In this case, one has $G \leq H$.*



Proof. Let $B = \mathsf{Body}_F(G)$ and $C = \mathsf{Body}_F(H)$. Since m is under n, we have that $C \leq B$.

- Consider the case where all the back-edges going out of the subtree of m point to nodes strictly above n. This means that $\mathsf{fv}_o(C) \subseteq \mathsf{fv}_o(B)$. By Proposition 2.3, we have:
$$G \equiv [\![B]\!]_F = B[[\![D_F(X_\alpha)]\!]_F/X]_{X_\alpha \in fv_o(B)}$$
and
$$H \equiv [\![C]\!]_F = C[[\![D_F(X_\alpha)]\!]_F/X]_{X_\alpha \in fv_o(C)} = C[[\![D_F(X_\alpha)]\!]_F/X]_{X_\alpha \in fv_o(B)}$$
Since $C \leq B$ we have also that $H \leq G$.

- If there is a back edge starting form the subtree of m and pointing n, this means that B is of the form $\sigma Y.D$, and that there is an occurrence Y_β free in C and bound by this σY. note that the definition of Y_β, $D_F(Y_\beta)$, is B. By proposition 2.3, $H = C[[\![D_F(X_\alpha)]\!]_F/X]_{X_\alpha \in fv_o(C)}$ and $G = [\![B]\!]_F = [\![D_F(X_\alpha)]\!]_F$, thus $G \leq H$.

□

Now, instead of comparing two FL-subformulas of a given formula, we address the problem of finding the minimum of a whole set of its FL-subformulas. The existence of the minimum is not always guaranteed, but in Proposition 2.5 and 2.6, we show two sufficient conditions for the minimum to exist.

Definition 2.17. A ***thread*** of F is a sequence $t = (F_i)_{i \in o}$, where $o \in \omega + 1$, $F_0 = F$ and $\forall i \in o$ such that $i + 1 \in o$ either $F_i \to F_{i+1}$ or $F_i = F_{i+1}$.

Definition 2.18. Let F be an occurrence and $t = (F_i)_{0\leq i\leq k}$ be a finite thread starting from F. For all $0 \leq i \leq k$, we set $n_i = \mathcal{N}_F(F_i)$. The thread t is said to be ***straight*** if n_{i+1} is the son of n_i in $\tau(F)$.

Example 2.11. The thread $(\Psi_\varepsilon, \Theta_i, (\odot\Psi \wedge \odot\Theta)_{ii}, (\odot\Theta)_{iir})$ is straight while $(\Psi_\varepsilon, \Theta_i, (\odot\Psi \wedge \odot\Theta)_{ii}, (\odot\Theta)_{iir}, \Theta_{iiri})$ is not.

Proposition 2.5. *If t is a straight thread then it admits a minimum* w.r.t. $\leq$.

Proof. We set $t = (F_i)_{0\leq i\leq k}$ and for all $0 \leq i \leq k$ we set $n_i = \mathcal{N}_F(F_i)$. We construct a sequence of nodes $(n_{k_i})_{0\leq i\leq l}$ starting from n_k *i.e.*, $k_0 = k$, and by applying the following algorithm:

- If there is no back-edge from the subtree of n_{k_i} to a node strictly above n_{k_i}, then halt.
- Otherwise, there is a back-edge pointing to a node strictly above n_{k_i}, call it $n_{k_{i+1}}$.

This algorithm always halts because we cannot go beyond n_0. We show first by a decreasing induction on k_i that $F_{k_l} \leq F_{k_i}$, for every $0 \leq i \leq l$.

- The base case is when $k_i = k_l$, the result is obvious since $F_{k_l} \leq F_{k_l}$.
- Suppose that $F_{k_l} \leq F_{k_i}$ for some $0 < i \leq l$ and let us show that $F_{k_l} \leq F_{k_{i-1}}$. Note that by construction, there is a back-edge from the sub-tree of $n_{k_{i-1}}$ pointing to n_{k_i}. We are then in the second case of Proposition 2.4, thus we have that $F_{k_i} \leq F_{k_{i-1}}$. We conclude this case using transitivity of the relation $\leq$.

Let us show now that $F_{k_l} \leq F_i$ for every $0 \leq i \leq n$. There are two cases to analyse:

- If $0 \leq i < k_l$, then n_i is above n_{k_l} and by construction there is no back-edge starting from the subtree of n_{k_l} and pointing to a node above n_{k_l}, in particular there is no back-edge pointing to a node above i (including n_i), thus we are in the fisrt case of Proposition 2.4, thus $F_{k_l} \leq F_i$.
- If $k_l \leq i \leq n$, then by construction there is j such that n_i is a node between n_{k_j} and $n_{k_{j+1}}$. By construction, there is a back-edge from the sub-tree of n_{k_j} pointing to $n_{k_{j+1}}$. Since the subtree of n_i contains that of n_{k_j}, there is a back-edge from the subtree of n_i pointing to $n_{k_{j+1}}$. Thus we are in the second case of Proposition 2.4, and we have $F_{k_{j+1}} \leq F_i$. Since we have $F_{k_l} \leq F_{k_{j+1}}$, and by transitivity of $\leq$, we have also that $F_{k_l} \leq F_i$.

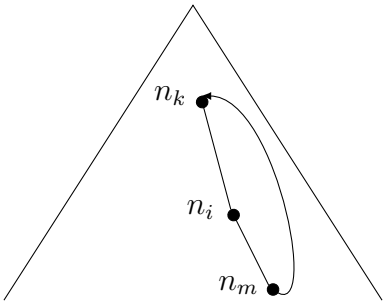
□

The other situation where the minimum of a set of FL-subformulas of F exists is when the set of their corresponding nodes in $\mathcal{G}(F)$ forms a strongly connected subgraph of $\mathcal{G}(F)$.

Proposition 2.6. *Let $\{F_i\}_{i\in I}$ be a set of FL-suboccurrences of the occurrence F and for all $i \in I$, let $n_i = \mathcal{N}_F(F_i)$ be the node of F_i in $\mathcal{G}(F)$. If the subgraph of $\mathcal{G}(F)$ restricted to the set of nodes $\{n_i\}_{i\in I}$ forms a strongly connected graph, then $\{F_i\}_{i\in I}$ admits a minimum* w.r.t. $\leq$.

Proof. The first observation is that among the nodes $\{n_i\}_{i\in I}$, there is a node n_k which is the nearest one to the root. Indeed, the underlying tree structure of $\mathcal{G}(F)$ prevents two distinct nodes to be nearest to the root. If this were the case, these two nodes being co-accessible, there would be a third node, nearer to the root, which allows them to be co-accessible, contradiction.

Let us first show the proposition in the particular case where the nodes $\{n_i\}_{i\in I}$ form an elementary cycle, we depict this situation below, where n_m is the predecessor of n_k in this cycle.



Let us show that $F_k \leq F_i$, for every $i \in I$. Since there is a back-edge from n_m to n_k, and since n_m is a node of the subtree of n_i for every $i \in I$, then there is a back-edge from the subtree of n_i to n_k. We are then in the case 2) of Proposition 2.4, thus we have that $F_k \leq F_i$.

We generalize this result to every cycle. More precisely, we show that if $\{n_i\}_{i\in J\subseteq I}$ forms a cycle $C = n_k, \ldots, n_k$ where n_k is the nearest node of C to the root, then $F_k \leq F_i$ for every $i \in J$. We proceed by induction on the number of elementary cycles of C. The case where C is itself an elementary cycle was treated above. Suppose that C is of the form $C = n_k \ldots m, C', m', \ldots, n_k$, where $C' = n_m \ldots n_m$ is an elementary cycle and where n_m is the nearest node to the root among its elements. Let C'' be the cycle obtained from C by collapsing the cycle C' to the node n_m, that is $C' = n_k, \ldots, m, n_m, m', \ldots, n_k$. The cycles C' and C'' have strictly less elementary cycles than C, thus by induction hypothesis we have that:

$$\begin{array}{ll} \forall n_i \in C', & F_m \leq F_i \\ \forall n_i \in C'', & F_k \leq F_i \quad (\star) \end{array}$$

Since $n_m \in C''$, we have that $F_k \leq F_m$, thus by transitivity of $\leq$, we have also that:

$$\forall n_i \in C', \quad F_k \leq F_i \quad (\dagger)$$

Since for every $n \in C$, either $n \in C'$ or $n \in C''$, we conclude using $(\star)$ and $(\dagger)$. □

Definition 2.19. Let $t = (F_i)_{i\in\omega}$ be a thread. We denote by $\overline{t}$ the sequence $(\overline{F_i})_{i\in\omega}$, that is the sequence obtained by forgetting the addresses of the occurrences of t. We denote by $\mathsf{Inf}(t)$ the elements of $\overline{t}$ that appears infinitely often in $\overline{t}$.

Remark 2.7. The elements of $\overline{t}$ are in the Fischer-Ladner closure of F_0, hence they are finitely many.

A thread starting from an occurrence F can be seen as a path in the graph $\mathcal{G}(F)$. The nodes of this path that occur infinitely often form a connected component.

Proposition 2.7. *Let F be an occurrence and $t = (F_i)_{i\in\omega}$ be a thread starting from F. For all $i \in \omega$, let $n_i = \mathcal{N}_F(F_i)$ be the node corresponding to the occurrence F_i in $\mathcal{G}(F)$. The sequence $p = (n_i)_{i\in\omega}$ forms a path in $\mathcal{G}(F)$. Let $\mathsf{Inf}(p)$ be the set of nodes that occurs infinitely often in p. The following facts hold:*

- *The set $\mathsf{Inf}(p)$ contains a node n such that n is in the path from the root to any other element of $\mathsf{Inf}(p)$, in other words n is the closest element of $\mathsf{Inf}(p)$ to the root, we denote it $\mathsf{min}(\mathsf{Inf}(p))$.*
- *The set $\mathsf{Inf}(t)$ admits a minimum* w.r.t. *the subformula ordering $\leq$, we denote it by $\mathsf{min}(\mathsf{Inf}(t))$.*
- *Moreover, if $k \in \omega$ is such that $n_k = \mathsf{min}(\mathsf{Inf}(p))$, then $\overline{F_k} = \mathsf{min}(\mathsf{Inf}(t))$.*

Proof. This proposition follows immediately from Proposition 2.6, by observing that the elements of $\mathsf{Inf}(p)$ forms a cycle. We have seen in the proof of Proposition 2.6 that the formulas corresponding to the nodes of the cycle admit a minimum, and that this minimum is the formula of the node which is the nearest to the root. □

Example 2.12. Consider the thread starting from Ψ_ε that goes always to the right:

$$t = \Psi_\varepsilon \to \Theta_i \to (\odot\Psi \wedge \odot\Theta)_{ii} \to (\odot\Theta)_{iir} \to \Theta_{iiri} \to (\odot\Psi \wedge \odot\Theta)_{iirii} \to (\odot\Theta)_{iiriir} \dots$$

One has:

$$\bar{t} = \Psi \to \Theta \to (\odot\Psi \wedge \odot\Theta) \to (\odot\Theta) \to \Theta \to (\odot\Psi \wedge \odot\Theta) \to (\odot\Theta) \dots$$

and $\mathsf{Inf}(t) = \{\Theta, (\odot\Psi \wedge \odot\Theta), \odot\Theta\}$. The minimum of $\mathsf{Inf}(t)$ *w.r.t.* $\leq$ is Θ.

The thread t describes the path $n_\mu(n_\nu n_\wedge n_{\odot_r})^\omega$, where $n_\mu, n_\nu, n_\wedge$ are respectively the nodes labelled $\mu X.$, $\nu Y.$, $\wedge$, and $n_{\odot_r}$ is the right node labelled $\odot$. $\mathsf{Inf}(p)$ is the set of nodes $\{n_\nu, n_\vee, n_{\odot_r}\}$, and the closest node of $\mathsf{Inf}(p)$ to the root is the node n_ν which is the node of Θ_i.

2.4 Proof systems for logics with fixed points

We have seen in Section 2.2 how to extend the signature $\mathcal{L}$ of a logic with least and greatest fixed points. This allows us to write inductive and coinductive definitions over $\mathcal{L}$. Now we want to reason about these definitions, that is to enrich a proof system S for this logic, to get one which is able to manipulate least and greatest fixed points formulas.

We will consider two ways to obtain such proof systems. The standard one, is to add to S Park's fixed point rules, which are rules for fixed points directly inspired by Knaster-Tarski's characterisation of extremal fixed points, we denote the resulting system $\mu\mathsf{S}$. The proof system $\mu\mathsf{S}$ is close to the proof systems we deal with usually in proof theory, in particular its proofs are finite trees. The advantage of this extension is that we can apply to it the well established tools and methods of reasoning of finitary proof theory. But the drawback of $\mu\mathsf{S}$, as we shall see, is that it is not suitable for proof search in practice.

Another alternative is to use rules for fixed points which are weaker than Park's rules, and to admit, as a counterpart to this simplification, infinitary proof structures. To get a

$$\frac{\vdash F[\mu X.F/X], \Gamma}{\vdash \mu X.F, \Gamma}\ (\mu) \qquad \frac{\vdash S^{\perp}, F[S/X]}{\vdash S^{\perp}, \nu X.F}\ (\nu)$$

Figure 2.1: Park's rules for least and greatest fixed point in a one-sided sequent calculus.

$$\frac{\Delta \vdash F[\mu X.F/X], \Gamma}{\Delta \vdash \mu X.F, \Gamma}\ (\mu_{\mathrm{r}}) \qquad \frac{S \vdash F[S/X]}{S \vdash \nu X.F}\ (\nu_{\mathrm{r}})$$

$$\frac{F[S/X] \vdash S}{\mu X.F \vdash S}\ (\mu_{\mathrm{l}}) \qquad \frac{\Delta, F[\nu X.F/X] \vdash \Gamma}{\Delta, \nu X.F \vdash \Gamma}\ (\nu_{\mathrm{l}})$$

Figure 2.2: Park's rules for least and greatest fixed point in a two-sided sequent calculus.

proof system which is sound, this proof system should be equipped with a validity condition which will ensure the well-foundedness of inductive reasoning, and only infinitary derivations satisfying this condition can be declared as proper proofs. The obtained proof system is denoted by $\mu\mathsf{S}^{\infty}$. We will be also interested in a restriction of this system to proofs which are finitely representable: we call these proofs **circular proofs** and the obtained proof system $\mu\mathsf{S}^{\omega}$.

In this section, we treat generically the extension of the proof systems introduced in Chapter 1 with least and greatest fixed points along the two approaches discussed above. A system S will then denote any proof system in $\{\mathsf{MALL}, \mathsf{LL}, \mathsf{LK}, \mathsf{LK}\odot\}$, but the approaches shown later can be adapted to many other proof systems.

2.4.1 Finitary proof system

The first possibility to extend a proof system with least and greatest fixed points is to add Park's fixed point rules. The obtained proof system is called **finitary** because its proofs are finite trees.

Definition 2.20. If S is a one-sided (resp. two-sided) proof system over a signature $\mathcal{L}$, we denote by $\mu_{\mathsf{c}}\mathsf{S}$ the proof system over the signature $\mu\mathcal{L}$, whose rules are obtained by adding to S the rules of Figure 2.1 (resp. Figure 2.2).

We comment on the two-sided rules of Figure 2.1, but the same observations hold dually for the one-sided ones. Park's rules reflect the Knaster-Tarski characterization of the least fixed point of an operator as its least pre-fixed point, and dually for the greatest fixed point. Indeed, the rule (μ_{l}) can be understood along the common interpretation of implication as inclusion as:

"If $F(S) \subseteq S$ then $\mu X.F \subseteq S$"

$$\frac{\vdash \Gamma, S \qquad \vdash S^\perp, F[S/X]}{\vdash \Gamma, \nu X.F}\ (\nu) \qquad\qquad \frac{\vdash F[\mu X.F], \Gamma}{\vdash \mu X.F, \Gamma}\ (\mu)$$

Figure 2.3: Rule for fixed points with an integrated cut.

Which means that $\mu X.F$ is smaller than any pre-fixed point S of F. The rule (μ_r) completes this characterisation by reflecting the fact that $\mu X.F$ is itself a pre-fixed point.

Remark 2.8. From now on we will consider only the one-sided formulation of our proof systems since this presentation is perfectly equivalent to the two-sided one for the systems we are considering.

All the systems we are considering here (MALL, LL, LK, LK$\odot$) enjoy the cut elimination property. Is this property preserved by adding Park's rules? The answer is no, at least with this precise formulation of the rules. Actually, the cut rule is not even admissible in these proof systems. Indeed, consider the sequent $\vdash 0, 0, \nu X.X$ which is provable in μ_cMALL using the following derivation:

$$\dfrac{\dfrac{}{\vdash 0, 0, \top}\ (\top) \qquad \dfrac{\dfrac{}{\vdash 0, \top}\ (\top)}{\vdash 0, \nu X.X}\ (\nu)}{\vdash 0, 0, \nu X.X}\ (\mathsf{Cut})$$

If we remove the cut rule from μ_cMALL, we cannot derive it anymore: the rule (ν) cannot be applied because the context around $\nu X.X$ contains more than a single formula, and no rule on 0 is available. Similar examples can be developed for the other systems. To get proof systems which admit the cut-elimination property, we have to consider a rule for greatest fixed points (Figure 2.3) obtained by aggregating the former rule with a cut as follows:

$$\dfrac{\vdash \Gamma, S \qquad \dfrac{\vdash S^\perp, F[S/X]}{\vdash S^\perp, \nu X.F}\ (\nu)}{\vdash \Gamma, \nu X.F}\ (\mathsf{Cut})$$

Definition 2.21. If S is a proof system over a signature $\mathcal{L}$, we denote by $\mu\mathsf{S}$ the proof system over the signature $\mu\mathcal{L}$, whose rules are obtained by adding to S the rules of Figure 2.3.

Remark 2.9. In $\mu\mathsf{S}$, the rule (ν) of $\mu_\mathsf{c}\mathsf{S}$ is still derivable. Indeed, when we take Γ to be $S^\perp$ in the rule (ν) of $\mu\mathsf{S}$, we can close the left premise using an axiom rule, the right one being the premise of the rule (ν) in $\mu_\mathsf{c}\mathsf{S}$.

Before discussing the cut-elimination property for $\mu\mathsf{S}$, we show some proof constructions that will be useful later on, in particular to properly define the cut-elimination rules. The first construction is **functoriality**, which is a construction used to derive the following rule:

$$\frac{\vdash F, G^\perp}{\vdash B[F/X], B^\perp[G^\perp/X]}\ (\mathsf{F_B})$$

In functional programming terms, it corresponds to a **map** function: its type is $(G \to F) \to (B(G) \to B(F))$. To derive this construction in $\mu\mathsf{S}$, the original proof system S should admit all the **finite η-expansions**, we define this property in the following:

Definition 2.22. A proof system S admits ***finite η-expansions*** if for every n-ary connective c the following rule, denoted η_c, is derivable:

$$\frac{\{\vdash F_i, G_i\}_{1\leq i\leq n}}{\vdash c[F_1,\dots,F_n], c^\perp[G_1,\dots,G_n]}\ (\eta_c)$$

Example 2.13. We show below how to derive $\eta_\otimes$ in MALL.

$$\dfrac{\dfrac{\vdash F_1, G_1 \quad \vdash F_2, G_2}{\vdash F_1 \otimes F_2, G_1, G_2}\ (\otimes)}{\vdash F_1 \otimes F_2, G_1 \parr G_2}\ (\parr)$$

It is easy to show that all the proof systems we are considering admit finite η-expansions.

Proposition 2.8. *The proof systems* MALL, LL, LK *and* $\mathsf{LK}\odot$ *admit finite η-expansions.*

Now we can define the functoriality construction in $\mu\mathsf{S}$, which can be seen as a generalisation of finite η-expansions:

Definition 2.23. Let B be an occurrence and π be a proof of $\vdash F, G$. We define the ***proof*** $\mathsf{F}_\mathsf{B}(\pi)$ of conclusion $\vdash B[F/X], B^\perp[G/X]$ by induction on the maximum depth of the occurrences of X in B, as follows:

- If $X \notin \mathsf{fv}(B)$ then $B^\perp[G/X] = (B[F/X])^\perp$ and $\mathsf{F}_\mathsf{B}(\pi)$ is an instance of the axiom rule.
- If $B \equiv X$ then $\mathsf{F}_\mathsf{B}(\pi)$ is π up to relocation of F and G in the addresses of B and $B^\perp$ respectively.
- If $B = c(B_1,\dots,B_n)$ where $c \notin \{\mu,\nu\}$ we perform an η-expansion on c and conclude by induction hypothesis.

$$\mathsf{F}_\mathsf{B}(\pi) \quad = \quad \dfrac{\left\{\ \dfrac{\mathsf{F}_{\mathsf{B}_i}(\pi)}{\vdash B_i[F/X], B_i^\perp[G/X]}\ \right\}_{1\leq i\leq n}}{\vdash c[B_1[F/X],\dots,F_n[F/X]], c^\perp[B_1^\perp[G/X],\dots,B_n^\perp[G/X]]}\ (\eta_c)$$

- If $B = \mu Y.C$, then $\mathsf{F}_\mathsf{B}(\pi)$ is the following derivation, where S stands for $(\mu Y.C[F/X])^\perp$.

$$\dfrac{\dfrac{}{\vdash S^\perp, S}\ (\mathsf{Ax}) \qquad \dfrac{\mathsf{F}_{\mathsf{C}[\mathsf{S}^\perp/\mathsf{Y}]}(\pi)}{\vdash C[S^\perp/Y][F/X], C^\perp[S/Y][G/X^\perp]}\ (\mu)}{\vdash S^\perp, C^\perp[S/Y][G/X^\perp]}\ (\nu)$$
$$\vdash \mu Y.C[F/X], \nu Y.C^\perp[G/X]$$

- The case where $B = \nu X.C$ is treated symmetrically.

$$\dfrac{\vdash \Gamma, C \qquad \dfrac{\vdash \Delta, C^\perp, S \qquad \vdash S^\perp, F[S/X]}{\vdash C^\perp, \Delta, \nu X.F}\,(\nu)}{\vdash \Gamma, \Delta, \nu X.F}\,(\mathsf{Cut}) \quad \rightarrow \quad \dfrac{\dfrac{\vdash \Gamma, C \qquad \vdash \Delta, C^\perp, S}{\vdash \Gamma, \Delta, S}\,(\mathsf{Cut}) \qquad \vdash S^\perp, F[S/X]}{\vdash \Gamma, \Delta, \nu X.F}\,(\nu)$$

$$\dfrac{\vdash \Gamma, C \qquad \dfrac{\vdash C^\perp, \Delta, F[\mu X.F/X]}{\vdash C^\perp, \Delta, \mu X.F}\,(\mu)}{\vdash \Gamma, \Delta, \mu X.F}\,(\mathsf{Cut}) \quad \rightarrow \quad \dfrac{\dfrac{\vdash \Gamma, C \qquad \vdash C^\perp, \Delta, F[\mu X.F/X]}{\vdash \Gamma, \Delta, F[\mu X.F/X]}\,(\mathsf{Cut})}{\vdash \Gamma, \Delta, \mu X.F}\,(\mu)$$

Figure 2.4: Auxiliary reduction rules $(\mu)/(\mathsf{Cut})$ and $(\nu)/(\mathsf{Cut})$ in $\mu\mathsf{S}$.

Remark 2.10. We usually write the functoriality construction as a rule named $(\mathsf{F_B})$. Indeed, $\mathsf{F_B}(\pi)$ starts with a derivation that decomposes B and $B^\perp$ using η-expansions until reaching the occurrences of X where it plugs π up to some renamings. This part of the derivation does not depend on π, that is why we write $\mathsf{F_B}(\pi)$ as follows, where $\vdash F, G$ is the conclusion of π:

$$\mathsf{F_B}(\pi) := \dfrac{\dfrac{\pi}{\vdash F, G}}{\vdash B[F/X], B^\perp[G/X]}\,(\mathsf{F_B})$$

Another useful rule, derivable in $\mu\mathsf{S}$, is the unfolding of greatest fixed points.

Proposition 2.9. *The following rule is derivable in* $\mu\mathsf{S}$*:*

$$\dfrac{\vdash F[\nu X.F/X], \Gamma}{\vdash \nu X.F, \Gamma}\,(\nu_{\mathsf{u}})$$

Proof. It suffices to apply the rule (ν) and to take $F[\nu X.F/X]$ as its invariant, then we can derive the following:

$$\dfrac{\vdash F[\nu X.F/X], \Gamma \qquad \dfrac{\dfrac{\dfrac{}{\vdash F^\perp[(\nu X.F)^\perp/X], F[\nu X.F/X]}\,(\mathsf{Ax})}{\vdash (\nu X.F)^\perp, F[\nu X.F/X]}\,(\mu)}{\vdash F^\perp[(\nu X.F)^\perp/X], F[F[\nu X.F/X]/X]}\,(\mathsf{F_F})}{\vdash \nu X.F, \Gamma}\,(\nu)$$

□

Let us go back to the cut-elimination property for $\mu\mathsf{S}$. Given a proof system S admitting the cut-elimination property, with rewriting rules $\mathcal{R}$, we show how to extend $\mathcal{R}$ to get a rewriting system for $\mu\mathsf{S}$. For that, we need three new rewriting rules: the auxiliary reduction rules $(\mu)/(\mathsf{Cut})$ and $(\nu)/(\mathsf{Cut})$ shown in Figure 2.4 and the principal $(\mu)/(\nu)$ reduction rule shown in Figure 2.5. Notice that this last reduction makes use of the functoriality construct.

$$\dfrac{\dfrac{\dfrac{\Pi_L}{\vdash \Gamma, B[(\mu X.B)/X]}}{\vdash \Gamma, \mu X.B}\,(\mu) \qquad \dfrac{\dfrac{\Pi_R}{\vdash \Delta, S} \qquad \dfrac{\Theta}{\vdash S^\perp, B^\perp[S/X]}}{\vdash \Delta, (\mu X.B)^\perp}\,(\nu)}{\vdash \Gamma, \Delta}\,(\mathsf{Cut})$$

$$\downarrow$$

$$\dfrac{\dfrac{\Pi_R}{\vdash \Delta, S} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\Theta}{\vdash S^\perp, B^\perp[S/X]} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\Theta}{\vdash S^\perp, B^\perp[S/X]} \qquad \dfrac{}{\vdash S^\perp, S}\,(\mathsf{Ax})}{\vdash S^\perp, (\mu X.B)^\perp}\,(\nu)}{\vdash B[S^\perp/X], B^\perp[(\mu X.B)^\perp/X]}\,(\mathsf{F_B}) \qquad \dfrac{\Pi_L}{\vdash B[(\mu X.B)/X], \Gamma}}{\vdash B[S^\perp/X], \Gamma}\,(\mathsf{Cut})}{\vdash S^\perp, \Gamma}\,(\mathsf{Cut})}{\vdash \Gamma, \Delta}\,(\mathsf{Cut})$$

Figure 2.5: Principal cut-elimination rule $(\mu)/(\nu)$ in $\mu\mathsf{S}$.

A candidate of reducibility argument has been used to show that μMALL equipped with these cut-elimination rules, enjoys cut elimination [Bae12b]. This argument can be adapted to obtain the same result for the other logics considered in this thesis.

From the Curry-Howard viewpoint, this cut-elimination result allows us see the proof systems presented above as programming languages, handling inductive and coinductive definitions.

Usually, in a proof search perspective, proving the cut-elimination property, in particular the admissibility of the cut rule, is a very big step toward reducing the non-determinism of the proof search. This is not really the case when we consider the proof system $\mu\mathsf{S}$ introduced above. Indeed, one has to figure out the invariant S of the (ν) rule, and in general there is no hint on how to choose it. We present in the next section a proof system which is better-suited for proof-search in the presence of least and greatest fixed points.

2.4.2 Infinitary proof system

If we replace in $\mu\mathsf{S}$ the rule (ν) by the unfolding rule (ν_u) (Proposition 2.9), we get a proof system where proof search is easier. The obtained proof system is however much weaker than $\mu\mathsf{S}$. For instance, $\nu X.X$ which is derivable in $\mu\mathsf{S}$, is no longer derivable. To offset the weakness of the unfolding rules, we allow in this new system infinite derivations, which we call $\mu\mathsf{S}^\infty$ **pre-proofs**.

Definition 2.24. A *$\mu\mathsf{S}^\infty$ pre-proof* is a possibly infinite tree, coinductively generated by the rules of S and the rules of Figure 2.6.

The problem of pre-proofs is that they are unsound. For example, it is easy to derive

$$\frac{\vdash F[\nu X.F/X], \Gamma}{\vdash \nu X.F, \Gamma}\ (\nu) \qquad \frac{\vdash F[\mu X.F/X], \Gamma}{\vdash \mu X.F, \Gamma}\ (\mu)$$

Figure 2.6: Fixed point rules for the $\mu\mathsf{S}^\infty$ proof system.

the empty sequent using the following pre-proof.

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash (\nu X.X)_i}\ (\nu)}{\vdash (\nu X.X)_\varepsilon}\ (\nu) \qquad \dfrac{\dfrac{\vdots}{\vdash (\mu X.X)_i}\ (\mu)}{\vdash (\mu X.X)_\varepsilon}\ (\mu)}{\vdash}\ (\mathsf{Cut})$$

In order to obtain proper proofs from pre-proofs, we add a validity condition that reflects the nature of our two fixed points.

Definition 2.25. A thread t (Definition 2.19) is said to be a ***ν-thread*** (we say sometimes that it is ***valid***) if $\mathsf{min}(\mathsf{Inf}(\bar{t}))$ is a ν-formula. Let $\gamma = (\vdash \Gamma_i)_{i\in\omega}$ be an infinite branch in a $\mu\mathsf{S}^\infty$ pre-proof. We say that γ is ***valid*** if there is a ν-thread $t = (F_i)_{i>j}$ where $j \in \omega$ which is not stationary, and such that $\forall i > j$, $F_i \in \Gamma_i$.

Definition 2.26. The ***proofs of*** $\mu\mathsf{S}^\infty$ are those pre-proofs in which every infinite branch is valid.

Remark 2.11. Note that a thread that validates a branch can start at any point of this branch. In particular, it can start from a cut formula.

This validity condition has its roots in parity games and is very natural for infinitary proof systems with fixed points. It is commonly found in deductive systems for modal μ-calculi, for instance in the proof system introduced by Dax *et al.* in [DHL06], which yields a sound and complete sequent calculus for the linear-time μ-calculus. We find a dual condition in the **refutations** for the modal μ-calculus [Wal95]. A refutation of a formula φ can be seen as infinite proof of $\neg\varphi$, it is therefore natural that the validity condition for refutations is the dual of our validity condition. Santocanale uses the same condition for his circular proofs [San02], but his formulation is simpler due to the restriction to purely additive logic. Indeed, his sequents have exactly two formulas, one on the right and one on the left. Thus, every infinite branch has **exactly** two threads: a right thread and a left one.

Let us give an intuition on this validity condition. We can think about greatest fixed points as objects that we can iterate infinitely many times and about least fixed points as objects that can be iterated only finitely many times. The validity condition says that an infinite branch cannot be supported by the unfolding of least fixed points only, but there should be at least a greatest fixed point formula that is unfolded there infinitely many times. Actually, the condition is more complicated than that because of the interleaving of fixed points. Let us take for instance the formula $\varphi = \mu X.\nu Y.X \oplus Y$ and the following $\mu\mathsf{MALL}^\infty$

pre-proof, where $\psi = \nu Y.\varphi \oplus Y$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\vdash (\varphi \oplus \psi)_{iilii}}(\oplus_1)}{\vdash \psi_{iili}}(\nu)}{\vdash \varphi_{iil}}(\mu)}{\vdash (\varphi \oplus \psi)_{ii}}(\oplus_1)}{\vdash \psi_i}(\nu)}{\vdash \varphi_\varepsilon}(\mu)$$

In the (unique) branch of this pre-proof, there is a (unique) thread that unfolds infinitely often the ν-formula ψ. But in this same thread, we unfold also the μ-formula φ which "dominates" the formula ψ: the μ-connective is more external than the ν-connective in φ. That is why, in the validity condition of a thread, we ask the **minimal** formula, among those occurring infinitely often, to be a ν-formula.

Remark 2.12. The above validity is interpreted in the two-sided case as follows. A thread t is said to be a ***μ-thread*** if $\mathsf{min}(\mathsf{Inf}(\bar{t}))$ is a μ-formula. Let $\gamma = (\Delta_i \vdash \Gamma_i)_{i\in\omega}$ be an infinite branch in a $\mu\mathsf{S}^\infty$ pre-proof. A ***thread*** $t = (F_i)_{i>j}$ is a right (resp. left) thread of γ if $\forall i > j$, $F_i \in \Gamma_i$ (resp. $F_i \in \Delta_i$). The branch γ is said to be ***valid*** if it either contains a right ν-thread or a left μ-thread, which is not stationary.

Sometimes, we will need to compare two infinitary proofs. Due to our explicit treatment of occurrences, the syntactic equality is too restrictive. What we need actually is an equality up-to renaming which we define in the following. But first, let us first introduce the notion of renamings and some useful operations on them.

Definition 2.27. A ***renaming*** is a bijection $b : \mathcal{A} \to \mathcal{A}'$ between two sets of pairwise disjoint addresses.

Let b be a renaming. We define $b^\uparrow : \Sigma^\star \to \Sigma^\star$ to be the partial function on addresses defined by:

$$b^\uparrow(\alpha) = b(\beta).\gamma \quad \text{if } \alpha = \beta.\gamma \text{ where } \beta \in \mathtt{Dom}(b)$$

Let b, b' be two renamings. We define $b \bullet b'$ to be the $b^\uparrow \circ b'$, where $\circ$ is function composition.

Let Γ be a sequent, and b be a renaming. We define $\Gamma \bullet b$ to be the sequent obtained from Γ by replacing every occurrence φ_α by $\varphi_{b^\uparrow(\alpha)}$.

Remark 2.13. Let b be a renaming of domain $\mathcal{A}$. Notice that if no address from $\mathcal{A}$ extends strictly an address appearing in Γ, then for every $\mu\mathsf{S}^\infty$ rule (r) of conclusion Γ and of premises $\Gamma_1, \ldots, \Gamma_n$, the following is still an occurrence of the rule (r):

$$\dfrac{\Gamma_1 \bullet b \quad \ldots \quad \Gamma_n \bullet b}{\Gamma \bullet b}(r)$$

Note that one may break a $\mu\mathsf{S}^\infty$ rule by applying a renaming b if an address in the domain of b extends an address in the conclusion sequent. For instance, if $b = \{i \mapsto \varepsilon\}$ and if we

consider the following instance of the rule (μ), then applying b to both the conclusion and the premise will not yield a licit rule (μ), since both occurrences will have the address ε.

$$\frac{\vdash (\varphi[\mu X.\varphi])_i}{\vdash (\mu X.\varphi)_\varepsilon}\ (\mu)$$

Definition 2.28. Let Π be a $\mu\mathsf{S}^\infty$ pre-proof. We define the ***base addresses*** of Π to be the set of addresses of the conclusion occurrences together with the cut occurrences of Π, we denote it by $\mathcal{A}(\Pi)$. Due to our Proviso 1.2, $\mathcal{A}(\Pi)$ is a set of pairwise disjoint addresses.

Let $b : \mathcal{A}(\Pi) \to \mathcal{A}'$ be a renaming. We define $\Pi \bullet b$ to be the pre-proof obtained by replacing the address of every conclusion occurrence and every cut occurrences by its image by b, and propagating this substitution all along the proof, that is to say, by replacing every sequent Γ in Π by $\Gamma \bullet b$.

Two $\mu\mathsf{S}^\infty$ proofs Π, Π' are said to be ***equal up-to renaming*** and we write $\Pi \equiv \Pi'$, if there is a renaming $b : \mathcal{A}(\Pi) \to \mathcal{A}(\Pi')$ such that $\Pi \bullet b = \Pi'$.

By Remark 2.13, and since no address in $\mathcal{A}(\Pi)$ extends strictly an address in Π (all the addresses of Π are extensions of the addresses of $\mathcal{A}(\Pi)$), $\Pi \bullet b$ is indeed a $\mu\mathsf{S}^\omega$ proof.

2.4.3 Circular proof system

As we aim to see proofs as programs, in particular as objects that can be effectively manipulated and stored in a finite memory, we will be interested in a system whose proofs are **finitely** representable, but when unfolded they yield $\mu\mathsf{S}^\infty$ proofs. We call this system the **circular** proof system $\mu\mathsf{S}^\omega$. Concretely, $\mu\mathsf{S}^\omega$ proofs have the shape of finite graphs, extending the tree shape of usual finitary proofs by the possibility of cycling, hence the name **circular** for this system.

Due to our explicit treatment of occurrences, we will need a **renaming** rule, which will permit us to come back to a sequent already seen, but with the right addresses. We define this rule as follows:

Definition 2.29. Let $b = \{\alpha_i \mapsto \beta_i\}_{i=1}^n$ be a renaming. We call ***renaming rule*** the following rule:

$$\frac{\vdash (\varphi_1)_{\beta_1}, \dots (\varphi_n)_{\beta_n}}{\vdash (\varphi_1)_{\alpha_1}, \dots (\varphi_n)_{\alpha_n}}\ (b)$$

As for $\mu\mathsf{S}^\infty$, we will define first an unsound proof structures called $\mu\mathsf{S}^\omega$ ***pre-proofs***. In a second step we will equip them with a validity condition.

Definition 2.30. We call the ***arity*** of an inference rule r the number of its premises, we denote it by $Ar(r)$. A tuple $(G, \lambda, \rho, \sigma)$, where:

- G is a finite set of vertices,
- λ is a labeling of vertices by sequents,
- ρ is a labeling of vertices by inference rules of $\mu\mathsf{S}^\infty$ and the renaming rule,
- $\sigma_g : Ar(\rho(g)) \to G$ is a successor function associated to each $g \in G$,

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(\star)}{\vdash \varphi_{ililil}, \varphi_{ililir}}\,(b)}{\vdash \varphi_{ilil}}\,(\nu),(\vee)}{(\star) \vdash \varphi_{ilil}, \varphi_{ilir}}\,(\mathsf{W})}{\vdash \varphi_{il}}\,(\nu),(\vee) \qquad \dfrac{\dfrac{}{\vdash \top_{irl}}\,(\top) \qquad \dfrac{(\dagger)}{\vdash \psi_{irr}}\,(b')}{\vdash (\top \wedge \psi)_{ir}}\,(\wedge)}{(\dagger) \vdash \psi_{\varepsilon}}\,(\nu),(\wedge)$$

Figure 2.7: Example of a circular pre-proof

is said to be ***well-typed*** if the following typing constraint holds:

- For every $g \in G$, if $r = \rho(g)$, $\Gamma = \lambda(g)$ and $\Gamma_i = \lambda(\sigma_g(i))$, then the following inference
$$\dfrac{\{\vdash \Gamma_i\}_{1 \leq i \leq Ar(r)}}{\vdash \Gamma}\,(r)$$
is an instance of either a $\mu\mathsf{S}^\infty$ rule or of the renaming rule.
- For every $g \in G$, if $\rho(g)$ is a renaming rule and $g' = \sigma_g(1)$, then $\rho(g')$ is not a renaming rule.

A $\mu\mathsf{S}^\omega$ ***pre-proof*** (also called ***circular pre-proof***) is a tuple $(G, \lambda, \rho, \sigma, c)$ of a well-typed tuple $(G, \lambda, \rho, \sigma)$ and a distinguished vertex $c \in G$ of P called its ***conclusion***.

Remark 2.14. The first condition of well-typedness ensures that we are applying licit $\mu\mathsf{S}^\infty$ rules. The second condition prevents us in particular from having a loop of renaming rules.

There is a natural graph that can be associated to a circular pre-proof, which is the graph induced by the successor relation:

Definition 2.31. Let $\Pi = (G, \lambda, \rho, \sigma, c)$ be a circular pre-proof. We define the ***graph of*** Π to be the graph $\mathcal{G}(\Pi) = (G, \to)$, whose vertices are the elements of G and $g \to g'$ iff $g' = \sigma_g(i)$ for some $1 \leq i \leq Ar(\rho(g))$. The pre-proof Π is said to be ***sourced*** if c is a source of $\mathcal{G}$, that is, for every vertex g, $c \to^\star g$.

We assume that all our circular pre-proofs are sourced. We are going to draw them as in the usual way: vertices are represented by the sequents labeling them, the inference rule of a vertex is indicated to the right of a horizontal bar above this vertex, and its successors are above the horizontal bar.

Example 2.14. Let $\varphi = \nu X.X \vee X$, and $\psi = \nu Y.\varphi \wedge (\top \wedge Y)$. Let us consider the circular pre-proof of Figure 2.7, where $b = \{ililil \mapsto ilil, ililir \mapsto ilir\}$ and $b' = \{irl \mapsto \varepsilon\}$. In this pre-proof, the $(\star)$ appearing in the premise of the renaming rule (b) means that the successor of the sequent $\vdash \varphi_{ililil}, \varphi_{ililir}$ is $\vdash \varphi_{ilil}, \varphi_{ilir}$, also marked by $(\star)$. Similarly, the $(\dagger)$ means that the successor of $\vdash \psi_{irr}$ is $\vdash \psi_\varepsilon$. One can imagine that there is a loop between the two symbols $(\star)$ and $(\dagger)$ respectively. We call it sometimes a back-edge.

Notice that this circular pre-proof has the shape of a tree, built using the inference rules of the system, except that there is a loop, or a back-edge from some leafs labeled by a

renaming rule to inner nodes of the derivation. Most of the time, our circular proofs will have this particular shape of trees with back-edges.

Circular pre-proofs can be seen as a sub-class of infinitary proofs. Indeed, if we unfold a circular pre-proof, applying the right renamings when encountering a renaming rule, then we get a $\mu\mathsf{S}^\infty$ pre-proof. There is a subtlety with the cuts, due to our disjointness condition, which requires the base addresses $\mathcal{A}(\Pi)$ of a pre-proof Π to be pairwise disjoint (see Proviso 1.2). Thus, when unfolding a cut rule, we have to attribute to the cut occurrences fresh addresses. To perform this, we will use during the unfolding an infinite supply of disjoint addresses S, splitting it on branching rules, and consuming a new address for cut rules. We define precisely in the following this operation of unfolding.

Definition 2.32. Let $\Pi = (G, \lambda, \rho, \sigma, c)$ be a circular pre-proof. Let $g \in G$, b be a renaming and S be an infinite set of disjoint addresses.

We set $\Gamma = \lambda(g)$. We define the pre-proof $\mathcal{U}(g, b, S)$ coinductively as follows:

- If $\rho(g) = r$ where r is a $\mu\mathsf{S}^\infty$ inference rule of arity n which is not a cut rule, and if $g_i = \sigma_c(i)$, we decompose S into n infinite sets $S_1, \dots, S_n$ such that $S = \uplus_i S_i$ and we set:
$$\mathcal{U}(g, b, S) = \dfrac{\mathcal{U}(g_1, b, S_1) \quad \dots \quad \mathcal{U}(g_n, b, S_n)}{\vdash \Gamma \bullet b}\ (r)$$
- If $\rho(g) = (\mathsf{Cut})$, $g_i = \sigma_g(i)$ where $i = 1, 2$, then $\lambda(g_1) = \Delta_1, \varphi_\alpha$ and $\lambda(g_2) = \Delta_2, \varphi^\perp_{\alpha^\perp}$. Let $f \in S$ and let S_1, S_2 be two infinite sets such that $S \setminus \{f\} = S_1 \uplus S_2$. We set:
$$\mathcal{U}(g, b, S) = \dfrac{\mathcal{U}(g_1, b :: \{\alpha \mapsto f\}, S_1) \quad \mathcal{U}(g_2, b :: \{\alpha^\perp \mapsto f^\perp\}, S_2)}{\vdash \Gamma \bullet b}\ (\mathsf{Cut})$$
- If $\rho(g) = (b')$ is a renaming rule and $g' = \sigma_g(1)$, then:
$$\mathcal{U}(g, b, S) = \mathcal{U}(g', b \bullet (b')^{-1}, S)$$

Let $\{\alpha_i\}_{1 \le i \le n}$ be the set of addresses of the conclusion occurrences of Π. Let b_0 be the renaming $\{\alpha_i \mapsto rl^i\}_{1 \le i \le n}$ and let $S_0 = \{rl^i \mid i > n\}$. We define the ***unfolding of*** Π to be $\mathcal{U}(c, b_0, S_0)$, we denote it by $\mathcal{U}(\Pi)$.

Thanks to the second condition on well-typedness of circular proofs, the coinductive definition of $\mathcal{U}(g, b, S)$ is productive, since we cannot loop in a sequence of renaming rules, which is the case where the definition does not produce anything. The produced object is indeed a $\mu\mathsf{S}^\omega$ pre-proof thanks to the first condition on well-typedness.

Morally, $\mathcal{U}(g, b, S)$ is the unfolding of Π starting from the vertex g, to which we apply the renaming b, and when the addresses of the cut occurrences are chosen among those of the supply S. In particular, $\mathcal{U}(\Pi)$ is the unfolding of Π starting from the conclusion c, in which the addresses of the conclusion occurrences were renamed by the addresses rl^i and the cut occurrences are chosen among the supply S_0, in order to guarantee the disjointness condition on sequents.

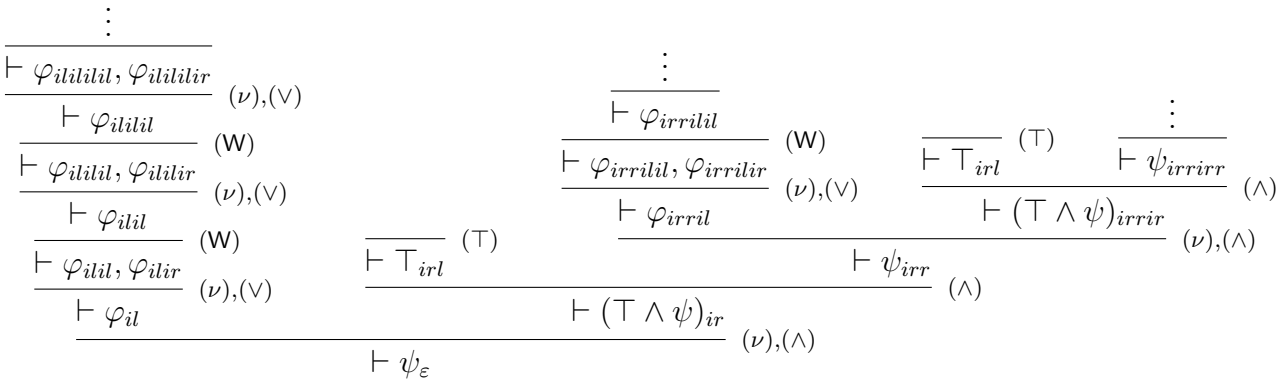


Figure 2.8: Unfolding of the circular pre-proof of Figure 2.7

Example 2.15. The unfolding of the circular proof of Figure 2.7 is the $\mu\mathsf{LK}^\infty$ pre-proof that starts as shown in Figure 2.8.

Note the importance of the renaming rule, which allowed us to unfold unambiguously the circular pre-proof of Example 2.14: with this precise renaming rule of the left loop, we obtained a leftmost branch that weakens always the right occurrence produced by the $\vee$. If the renaming rule of the left loop where instead $\{ililil \mapsto ilir, ililir \mapsto ilil\}$, the unfolding of the leftmost branch would weaken alternatively the right and the left occurrences produced by the $\vee$.

Remark 2.15. We will usually not write the bijection b in the renaming rule when it is inferable from the context. This situation holds when the conclusion c (hence the premise) of the renaming rule contains occurrences with distinct underlying formulas, that is, for every $F, G \in c$ we have $F \not\equiv G$.

Just as $\mu\mathsf{S}^\infty$ pre-proofs, $\mu\mathsf{S}^\omega$ pre-proofs are unsound. To fix that defect, we introduce a validity condition on circular pre-proofs, similar to the one used for infinitary pre-proofs. We relate these two validity conditions by showing that a circular proof is valid if and only if its unfolding is valid.

Definition 2.33. Let Π be a circular pre-proof of conclusion c and let $\mathcal{G}$ be its graph. Let $p = (g_i)_{i\in\omega}$ be an infinite path in $\mathcal{G}$. A ***trace*** on p is a sequence of occurrences $t = (F_i)_{i\geq k}$ where $k \in \omega$, such that for every $i \geq k$, we have that $F_i \in \lambda(g_i)$ and either:

- $\rho(g_i)$ is not a renaming rule, then $F_i \to F_{i+1}$ or $F_i = F_{i+1}$;
- or $\rho(g_i)$ is a renaming rule (b) and $F_i = \varphi_\alpha$, then $F_{i+1} = \varphi_{b(\alpha)}$.

A ***trace*** t on an infinite path is said to be ***valid*** when $\mathsf{min}(\mathsf{Inf}(\bar{t}))$ is a ν-formula. An ***infinite path is valid*** if it has a valid trace. A ***circular pre-proof is valid*** when all its infinite paths are valid. A $\mu\mathsf{S}^\omega$ proof is a valid $\mu\mathsf{S}^\omega$ pre-proof.

Proposition 2.10. *A $\mu\mathsf{S}^\omega$ pre-proof Π is valid if and only if $\mathcal{U}(\Pi)$ is valid.*

Proof. Note that the infinite paths of Π are in a one-to-one correspondence with the infinite branches of $\mathcal{U}(\Pi)$. Moreover, for every trace $tr = (F_i)_{i\in\omega}$ of an infinite path of Π, there is an infinite thread $th = (G_i)_{i\in\omega}$ in its corresponding branch such that $\overline{F_i} = \overline{G_i}$, and conversely. Thus $\mathsf{min}(\mathsf{Inf}(\overline{tr})) = \mathsf{min}(\mathsf{Inf}(\overline{th}))$ and we can conclude that Π is valid if and only if $\mathcal{U}(\Pi)$ is. □

We have seen that every circular pre-proof can be unfolded into an infinitary one. It is however not true that every infinitary pre-proof is the unfolding of a circular pre-proof. This holds only for a subset of infinitary pre-proof called **regular**, we define them below.

Definition 2.34. Let Π be a $\mu\mathsf{S}^\infty$ pre-proof and let $\mathcal{S}$ be the set of its sub-trees. We say that Π is ***regular*** if $S|_\equiv$ (the quotient set of $\mathcal{S}$ by $\equiv$) is finite.

In other words, Π is regular if it contains only finitely many sub-trees up-to renaming.

Proposition 2.11. *If Π is a regular $\mu\mathsf{S}^\infty$ pre-proof, then there is a $\mu\mathsf{S}^\omega$ pre-proof Θ such that $\mathcal{U}(\Theta) \equiv \Pi$. We say that Θ is a* **circular representation** *of Π.*

Proof. Let Π be a $\mu\mathsf{S}^\infty$ pre-proof of conclusion c. Let us construct the subtree T of Π as follows: at the beginning T contains only the conclusion c of Π. Suppose that we have just added a sequent to T, and let s be one of its successors. If there is a sequent $s' \in T$ such that the sub-preproofs of Π rooted respectively in s and s' are equal up to renaming, then we add s to T and stop processing s. If this is not the case, we add s to T and continue with its successors.

Since Π is regular, it cannot contain a branch such that the sub-preproofs rooted in the sequents of this branch are all pairwise not equal up-to renaming. Thus the process described above that constructs T will always halt, and yields a finite sub-derivation of Π rooted in c, such that for every leaf $\vdash \Delta$ of T, either $\vdash \Delta$ is the conclusion of 0-ary rule, or there is an inner node $\vdash \Delta'$ (that is, a node which is not a leaf) of T, such that the sub-preproofs of Π rooted respectively in $\vdash \Delta$ and $\vdash \Delta'$ are equal up to the renaming, in particular there is a renaming b such that $\Delta' = \Delta \bullet b$.

We construct a circular proof Θ from T, by applying to each leaf of this second kind the renaming rule (b), and setting $\vdash \Delta'$ to be the successor of $\vdash \Delta$.

Using a bisimulation, we can show that Π and $\mathcal{U}(\Theta)$ are equal up-to renaming. □

Remark 2.16. Note that the representation of a regular pre-proof by a circular one is not unique.

Remark 2.17. Notice that in general, we cannot strengthen Proposition 2.11 into a syntactic equality between $\mathcal{U}(\Theta)$ and Π, since the base addresses of $\mathcal{U}(\Theta)$ are chosen from the particular supply $\{rl^n \mid n \in \omega\}$, and may not be equal to those of Π.

Decidability of the validity condition. Even if circular proofs are finitely representable, their validity condition concerns their infinite paths, which can be generated by any combination of elementary cycles of their graphs. In particular, checking the validity of elementary cycles is not enough to check the validity of the whole proof as illustrated by the following example.

Example 2.16. Let $\varphi = \nu X.\mu Y.(Y \vee X)$, $\psi = \mu Y.(Y \vee \varphi)$ its unfolding and $\bot = \mu X.X \wedge X$. Consider the following circular pre-proof.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(\star)}{\vdash \psi_{\alpha il}, \psi_{\alpha iri}, \bot_{\gamma il}}\ (\{\alpha il \mapsto \beta, \alpha iri \mapsto \alpha\})}{\vdash \psi_{\alpha il}, \varphi_{\alpha ir}, \bot_{\gamma il}}\ (\nu)}{\vdash \psi_{\alpha}, \bot_{\gamma il}}\ (\mu), (\vee)}{\vdash \psi_{\alpha}, \psi_{\beta}, \bot_{\gamma il}}\ (\mathsf{W}) \qquad \dfrac{\dfrac{\dfrac{\dfrac{(\star)}{\vdash \psi_{\beta il}, \psi_{\beta iri}, \bot_{\gamma ir}}\ (\{\beta il \mapsto \alpha, \beta iri \mapsto \beta\})}{\vdash \psi_{\beta il}, \varphi_{\alpha ir}, \bot_{\gamma ir}}\ (\nu)}{\vdash \psi_{\beta}, \bot_{\gamma.ir}}\ (\mu), (\vee)}{\vdash \psi_{\alpha}, \psi_{\beta}, \bot_{\gamma ir}}\ (\mathsf{W})}{(\star) \vdash \psi_{\alpha}, \psi_{\beta}, \bot_{\gamma}}\ (\mu), (\wedge)$$

If γ_r and γ_l denote the right and left cycles respectively, the paths $(\gamma_r)^\omega$ and $(\gamma_l)^\omega$ are valid, but the path $(\gamma_r.\gamma_l)^\omega$ that alternates the two cycles is not valid.

Despite this, we show that the validity of a circular proof is decidable, more precisely that it is in PSPACE. For that, we adapt the proof used by Dax *et al.* [DHL06] to show that their validity condition is decidable. We fix in the rest of this section a circular pre-proof $\Pi = (P, c)$, where $P = (G, \lambda, \rho, \sigma)$. Let $\mathcal{G}$ be the graph of Π.

The idea is to design two word automata, the first one is a Büchi automaton that accepts all the paths of $\mathcal{G}$ and the other is a parity automaton that accepts only the valid ones. Then checking the validity of Π is reduced to verifying the language inclusion of these two automata. For an introduction to automata over infinite words, see Chapter 8.

Let us first define the parity automaton $\mathcal{A}$ whose language is the set of valid paths of $\mathcal{G}$.

Definition 2.35. We define the parity automaton $\mathcal{A} = (\Sigma, Q, c, \delta, Q_I)$ as follows.

- The alphabet Σ is the set of vertices of $\mathcal{G}$.
- The set of states Q is the set of occurrences appearing in the sequents labeling the vertices of $\mathcal{G}$, together with a new state w. That is, $Q = \bigcup\limits_{g \in G} \lambda(g) \uplus \{w\}$.
- The priority function $c : Q \to \omega$ is a function such that:
  - If F is a μ-occurrence (resp. ν-occurrence), then $c(F)$ is odd (resp. even).
  - If $F \leq G$ then $c(F) \leq c(G)$.
  - $c(w)$ is odd.

  The function c is not uniquely defined, but we fix one arbitrarily.
- The initial states Q_I are the conclusion occurrences and the state w, that is $Q_I = \lambda(c) \cup \{w\}$.
- The transition relation δ is defined as follows. Let $g \in \Sigma$ and $q \in Q$.
  - If $q \in \lambda(g)$ and if $\rho(g)$ is a $\mu\mathsf{S}^\omega$ rule of arity n, then for every $i \leq n$ and every occurrence $q' \in \lambda(\sigma_g(i))$ which is a sub-occurrence of q, we set $(q, g, q') \in \Delta$.
  - If $q := \varphi_\alpha \in \lambda(g)$ and if $\rho(g)$ is a renaming rule (b), then we set $(q, g, \varphi_{b(\alpha)}) \in \Delta$.

- If $q = w$, then we set $(w, g, w) \in \Delta$. If moreover, $\lambda(g) = (\mathsf{Cut})$, we have that $\lambda(\sigma_g(1)) = \Gamma, F$ and $\lambda(\sigma_g(2)) = \Gamma', F^\perp$, and we set $(w, g, F), (w, g, F^\perp) \in \Delta$.

In this automaton, the state w means "wait". It is a state waiting for a trace to start. We use it here because our traces are not required to start from the conclusion, but may start from cut occurrences.

Lemma 2.1. *For every infinite path p of $\mathcal{G}$, $p \in \mathcal{L}(\mathcal{A})$ if and only if p is valid.*

Proof. We show that if p is valid then it belongs to the language of $\mathcal{L}(\mathcal{A})$. The other direction can be shown in the same way.

Let $t = (F_i)_{i \geq k}$ be a valid trace on p. Let $\rho = (G_i)_{i \in \omega}$ where $G_i = w$ for every $i < k$ and $G_i = F_i$ otherwise. Notice that ρ is a run of $\mathcal{A}$ over p. By assumption, the minimal formula of $\mathsf{Inf}(\bar{t})$ is a ν-formula. Since the priority function of $\mathcal{A}$ is compatible with the subformula ordering, and since the priority of a ν-formula is even, the minimal priority seen infinitely often is even. Thus, $p \in \mathcal{L}(\mathcal{A})$. □

Let us now define the Büchi automaton $\mathcal{B}$ that accepts all the infinite paths of $\mathcal{G}$.

Definition 2.36. Let $\mathcal{B} = \{\Sigma, \Sigma, \delta, q_I, \Sigma\}$ be the Büchi automaton whose set of states is the alphabet Σ itself. Its transition relation δ is defined as follows: for every $g \in \Sigma$, and for every g' such that $g \to g'$, we set $(g, g, g') \in \delta$. The initial state q_I is the conclusion vertex c, and the set of final states is Σ.

The following proposition is a direct consequence of Lemma 2.1.

Proposition 2.12. *The circular pre-proof Π is valid if and only if $\mathcal{L}(\mathcal{B}) \subseteq \mathcal{L}(\mathcal{A})$.*

This shows that validity can be decided in PSPACE.

Theorem 2.4. *Deciding whether a circular proof is valid is in* PSPACE.

Proof. By Proposition 2.12 it suffices to check the non-emptiness of the language $\mathcal{L}(\mathcal{B}) \cap \overline{\mathcal{L}(\mathcal{A})}$. Using well-known automata-theoretic constructions and Savitch's theorem, this problem can be reduced to checking the emptiness of an automaton of the form $\mathcal{B}' \times \mathcal{A}'$, which can be done in PSPACE. □

Cut-elimination. As for the finitary system $\mu\mathsf{S}$, one can extend the rewriting rules of S to handle the interaction between the least and greatest fixed points with the cut rule, by adding the auxiliary reduction rule $(\sigma)/(\mathsf{Cut})$ (Figure 2.9) and the principal reduction rule $(\mu)/(\nu)$ (Figure 2.10). The question now is to know whether the proof system $\mu\mathsf{S}^\omega$, equipped with this rewriting rules admits the cut elimination property. But first we need to be more precise about what we call the cut-elimination property. Indeed, as we are dealing with infinitary proofs, it is clear that we cannot eliminate cuts in a finite number of steps. Thus we have to consider infinite reduction sequences, and the notion of termination of the cut elimination procedure in the finitary case should be replaced by the notion of **productivity**, which ensures that infinite reduction sequences eliminate cuts at the limit. Furthermore, as $\mu\mathsf{S}^\omega$ is equipped with a validity condition, one should ensure that the pre-proof produced by the cut-elimination process is indeed a proof. Hence, the cut elimination property in an

$$\dfrac{\vdash \Gamma, C \qquad \dfrac{\vdash C^\perp, \Delta, F[\sigma X.F/X]}{\vdash C^\perp, \Delta, \sigma X.F}\,(\sigma)}{\vdash \Gamma, \Delta, \mu X.F}\,(\mathsf{Cut}) \quad \rightarrow \quad \dfrac{\dfrac{\vdash \Gamma, C \qquad \vdash C^\perp, \Delta, F[\sigma X.F/X]}{\vdash \Gamma, \Delta, F[\sigma X.F/X]}\,(\mathsf{Cut})}{\vdash \Gamma, \Delta, \sigma X.F}\,(\sigma)$$

Figure 2.9: Auxiliary reduction rule $(\sigma)/(\mathsf{Cut})$ in $\mu\mathsf{S}^\omega$.

$$\dfrac{\dfrac{\dfrac{\Pi}{\vdash \Gamma, B[(\mu X.B)/X]}\,(\mu)}{\vdash \Gamma, \mu X.B} \qquad \dfrac{\dfrac{\Theta}{\vdash \Delta, B^\perp[(\mu X.B)^\perp/X]}\,(\nu)}{\vdash \Delta, (\mu X.B)^\perp}}{\vdash \Gamma, \Delta}\,(\mathsf{Cut})$$

$$\downarrow$$

$$\dfrac{\dfrac{\Pi}{\vdash \Gamma, B[(\mu X.B)/X]} \qquad \dfrac{\Theta}{\vdash \Delta, B^\perp[(\mu X.B)^\perp/X]}}{\vdash \Gamma, \Delta}\,(\mathsf{Cut})$$

Figure 2.10: Principal cut-elimination rule $(\mu)/(\nu)$ in $\mu\mathsf{S}^\omega$.

infinitary setting means that the infinite cut elimination procedure is productive, and that the produced pre-proof is valid. Little work has been done in this direction, and the only known result of cut-elimination in an infinitary setting is for $\mu\mathsf{ALL}^\infty$ (the additive linear logic with least and greatest fixed points) due to Fortier and Santocanale [FS13]. We dedicate Chapter 3 to the cut-elimination result for $\mu\mathsf{MALL}$ and discuss its extension to $\mu\mathsf{LL}$, $\mu\mathsf{LK}$ and $\mu\mathsf{LK}\odot$.

Derivable rules. In finitary proof theory, when we derive a rule, we can use it as a proper rule of the system. For infinitary proofs, one should be more careful when using a derived rule, since the derivation may hide the minimum of the thread linking an occurrence of the conclusion to an occurrence of some premise of the rule. When considering the validity of an infinitary proof using such a derived rule, one has to look inside the derivation of this rule to compute the minimum of the threads of a branch using it infinitely often. Thus, we use derived rules in this thesis as a shortcut/ abbreviation for a derivation, not as a proper rule of the infinitary system under consideration.

2.4.4 From finitary to circular proofs

We show in this section how to transform effectively every $\mu\mathsf{S}$ proof into a $\mu\mathsf{S}^\omega$ proof of the same conclusion. For that, we need to introduce the functoriality construction in $\mu\mathsf{S}^\omega$. This construction is the infinitary counterpart of functoriality in $\mu\mathsf{S}$, in the sense that it transforms every proof π of $\vdash F, G$, into a proof of $\vdash B[F/X], B^\perp[G/X]$, by applying coinductively η-expansions on the connectives of B, and plugging the derivation π when the variable X is reached. To define properly the functoriality in $\mu\mathsf{S}^\omega$, we need to generalize it to handle multiple variables.

Definition 2.37. Let B be an occurrence, $\vec{X} = (X_i)_{1\leq i\leq n}$ be a family of variables and let $\vec{\Pi} = (\Pi_i)_{1\leq i\leq n}$ be a collection of $\mu\mathsf{S}^\omega$ pre-proofs of respective conclusions $\vdash F_i, G_i$. We define coinductively the ***pre-proof*** $\mathsf{F}^\infty_B(\vec{\Pi})$ of conclusion $\vdash B[\vec{F}/\vec{X}], B^\perp[\vec{G}/\vec{X}]$, where $\vec{F} = \{F_i\}_i$ and $\vec{G} = \{G\}_i$, as follows:

- If for every $1 \leq i \leq n$, we have $X_i \notin \mathsf{fv}(B)$ then we set $\mathsf{F}^\infty_B(\pi)$ to be an instance of the axiom rule on B.
- If $B \equiv X_i$ for some $1 \leq i \leq n$, then $\mathsf{F}^\infty_B(\pi)$ is π_i up to relocation of F_i and G_i in the addresses of B and $B^\perp$ respectively.
- If $B = c(\vec{C})$, where $c \notin \{\mu, \nu\}$ then $\mathsf{F}^\infty_B(\vec{\Pi})$ is:

$$\dfrac{\left\{\quad\dfrac{\vdash \mathsf{F}^\infty_C(\vec{\Pi})}{\vdash C[\vec{F}/\vec{X}], C^\perp[\vec{G}/\vec{X}]}\quad\right\}_{C\in\vec{C}}}{\vdash c(\vec{C})[\vec{F}/\vec{X}], c^\perp(\vec{C}^\perp)[\vec{G}/\vec{X}]}\ (\eta_c)$$

- If $B = \mu X.C$ then $\mathsf{F}^\infty_B(\vec{\Pi})$ is obtained from applying functoriality on C with $B(\vec{\Pi})$ as the derivation for the new free variable $X_{n+1} := X$:

$$\mathsf{F}^\infty_B(\vec{\Pi}) \quad = \quad \dfrac{\dfrac{\dfrac{\mathsf{F}^\infty_C(\vec{\Pi}, B(\vec{\Pi}))}{\vdash C[(\mu X.C)/X][\vec{F}/\vec{X}], C^\perp[(\nu X.C^\perp)/X][\vec{G}/\vec{X}]}\ (\nu)}{\vdash C[(\mu X.C)/X][\vec{F}/\vec{X}], (\nu X.C^\perp)[\vec{G}/\vec{X}]}\ (\mu)}{\vdash (\mu X.C)[\vec{F}/\vec{X}], (\nu X.C^\perp)[\vec{G}/\vec{X}]}$$

- The case where $B = \nu X.C$ is treated symmetrically.

The derivation $\mathsf{F}^\infty_B(\Pi)$ is regular, we denote by $\mathsf{F}^\omega_B(\Pi)$ the least circular representation of it.

Example 2.17. Let $B = (\mu Y.X \otimes Y)_\varepsilon$ and let Π be a $\mu\mathsf{MALL}^\omega$ proof of conclusion $\vdash F, G$. The derivation $\mathsf{F}^\omega_B(\Pi)$ is the following:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Pi}{\vdash F, G} \qquad \dfrac{(\star)}{\vdash B[F/X], B^\perp[G/X]}}{\vdash F \parr B[F/X], G \otimes B^\perp[G/]}\ (\parr),\ (\otimes)}{\vdash F \parr B[F/X], \nu Y.G \otimes Y}\ (\nu)}{(\star) \vdash \mu Y.F \parr Y, \nu Y.G \otimes Y}\ (\mu)$$

Remark 2.18. As for the finitary functoriality, the infinitary functoriality on B can be written as the following rule named $(\mathsf{F}^\omega_\mathsf{B})$ since it is entirely guided by B.

$$\dfrac{\vdash F, G}{\vdash B[F/X], B^\perp[G/X]}\ (\mathsf{F}^\omega_\mathsf{B})$$

Proposition 2.13. *If $\vec{\Pi}$ is a collection of $\mu\mathsf{S}^\omega$ proofs, then $\mathsf{F}^\omega_B(\vec{\Pi})$ is also a $\mu\mathsf{S}^\omega$ proof.*

Proof. An infinite path of $\mathsf{F}^\omega_B(\vec{\Pi})$ either has an infinite path of some Π_i as a suffix, or is only visiting sequents of $\mathsf{F}^\omega_B(\vec{\Pi})$ that are not sequents of the input derivations $\vec{\Pi}$. In the former case, the path is valid provided that the input derivations are valid. In the latter case, the path contains exactly two dual traces, one of which must be valid. Thus, $\mathsf{F}^\omega_B(\vec{\Pi})$ is valid provided that the input derivations are. □

We now make use of functoriality to translate finitary $\mu\mathsf{S}$ proofs into circular derivations.

Definition 2.38 (Translation from $\mu\mathsf{S}$ to $\mu\mathsf{S}^\omega$)**.** Given a $\mu\mathsf{S}$ proof π of $\vdash \Gamma$, we define inductively the $\mu\mathsf{S}^\omega$ pre-proof Π of $\vdash \Gamma$, as follows:

- If π starts with an inference that is present in $\mu\mathsf{S}^\omega$, that is an inference in S or the rule (μ), then we use the same inference for Π and proceed recursively. For instance,

$$\pi \;=\; \dfrac{\dfrac{\theta}{\vdash \Gamma, F[\mu X.F/X]}}{\vdash \Gamma, \mu X.F}\,(\mu) \qquad \text{yields} \quad \Pi \;=\; \dfrac{\dfrac{\Theta}{\vdash \Gamma, F[\mu X.F/X]}}{\vdash \Gamma, \mu X.F}\,(\mu)$$

  where Θ is the translation of θ.

- Otherwise, π starts with an instance of the ν rule of $\mu\mathsf{S}$:

$$\pi \;=\; \dfrac{\dfrac{\pi_1}{\vdash \Gamma, S} \qquad \dfrac{\pi_2}{\vdash S^\perp, F[S/X]}}{\vdash \Gamma, \nu X.F}\,(\nu)$$

  We transform it as follows, where Π_1 and Π_2 are the translations of π_1 and π_2 respectively:

$$\Pi \;=\; \dfrac{\dfrac{\Pi_1}{\vdash \Gamma, S} \qquad \dfrac{\dfrac{\dfrac{\Pi_2}{\vdash S^\perp, F[S/X]} \qquad \dfrac{\dfrac{(\star)}{\vdash S^\perp, \nu X.F}}{\vdash F^\perp[S^\perp/X], F[(\nu X.F)/X]}\,(\mathsf{F}^\omega_\mathsf{F})}{\vdash S^\perp, F[\nu X.F/X]}\,(\mathsf{Cut})}{(\star)\ \vdash S^\perp, \nu X.F}\,(\nu)}{\vdash \Gamma, \nu X.F}\,(\mathsf{Cut})$$

Remark 2.19. The infinite paths of the $\mu\mathsf{S}^\omega$ pre-proof Π constructed above are either infinite paths of the functoriality construct or come from the loop on the sequent $\vdash S^\perp, \nu X.F$.

Proposition 2.14. *For any $\mu\mathsf{S}$ derivation π, its translation Π is a $\mu\mathsf{S}^\omega$ proof.*

Proof. We have to check that all infinite paths of Π are valid. Consider one such infinite path. After a finite prefix, the path must be contained in the pre-proof obtained from the translation of a coinduction rule (second case in the above definition). If the path is eventually contained in a functoriality construct, then it is valid by Proposition 2.14. Otherwise, the path visits infinitely often the sequent $\vdash S^\perp, \nu X.F$ corresponding to our translated coinduction rule. This path is validated by the trace that contains the successive sub-occurrences of $\nu X.F$ in those sequents. Indeed, in this trace, $\nu X.F$ is principal infinitely often, moreover it is minimal among formulas that appear infinitely often: this simply follows from the fact that all formulas encountered along the trace inside the functoriality construct $(\mathsf{F}^\omega_\mathsf{F})$ contain $\nu X.F$ as a subformula. □

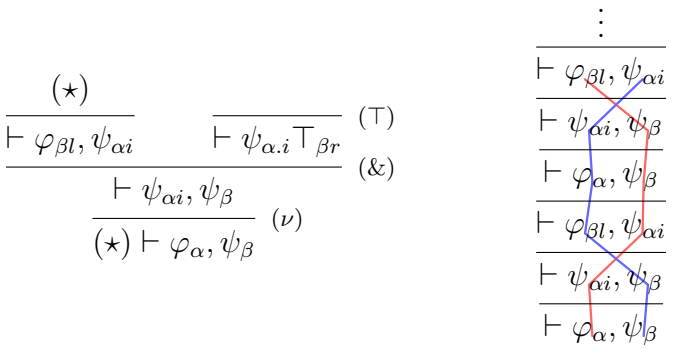


Figure 2.11: A circular proof which is not translatable and its infinite path.

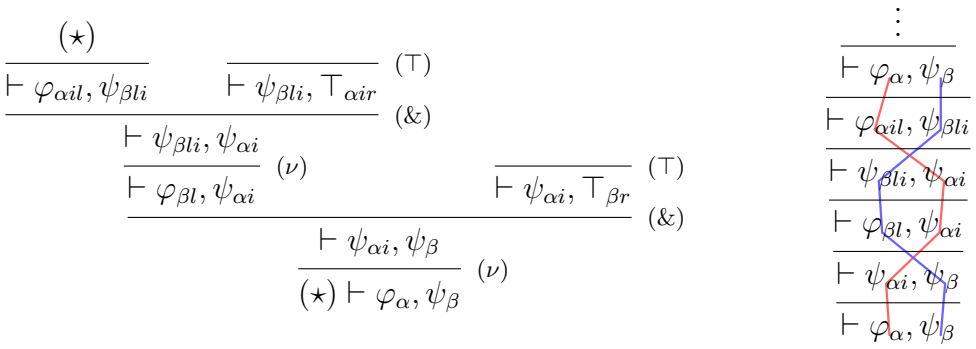


Figure 2.12: A translatable circular proof and its infinite path.

2.4.5 From circular to finitary proofs

Going from circular to finitary proofs is much more involved, since one has to extract from circular proofs the right invariants for Park's rule (ν). Unfortunately, we do not have a general algorithm to translate circular proofs into finitary ones. Instead, we give a sufficient condition on $\mu\mathsf{S}^\omega$ proofs which guarantees that they can be translated into $\mu\mathsf{S}$ ones, we call it the **translatability criterion**.

The translatability criterion

Definition 2.39. Let Π be a circular pre-proof and $p = (g_i)_{i\in\omega}$ be an infinite path of $\mathcal{G}(\Pi)$. A trace $t = (F_i)_{i\in\omega}$ of p is said to be ***strongly valid*** if t is valid and if there is $k \in \omega$ such that $\forall i, j \geq k$, if $g_i = g_j$ then $F_i = F_j$.

A circular proof Π is said to be ***translatable*** if every infinite path of $\mathcal{G}(\Pi)$ has a strongly valid trace.

The strong validity condition for the trace t means that, at some point in the path p, for every vertex g, whenever the trace t meets g, it meets it at the level of the same occurrence.

Example 2.18. Let $\varphi = \nu X.X \,\&\, \top$ and $\psi = \varphi \,\&\, \top$ be its unfolding. Consider the circular proof of Figure 2.11. This proof is not translatable, since its infinite path (shown in Figure 2.11) is not strongly valid. Indeed, the red trace meets the sequent $\vdash \varphi_\alpha, \psi_\beta$ alternatively at the level of the occurrences φ_α and ψ_β. The same holds for the blue trace.

Let us consider the circular proof of Figure 2.12, which is an other circular representation of the unfolding of the circular proof of Figure 2.11. This time, this circular proof is translatable. Its infinite path (shown in Figure 2.12) is strongly valid, thanks to the red trace for instance.

Remark 2.20. As shown by Example 2.18, even if two circular proofs represent the same infinitary proof, one of them may be translatable and the other not. The property of being translatable is not inherent to $\mu\mathsf{S}^\infty$ proofs, but depends strongly on their representation as circular proofs.

The invariant property

Our translation result relies on a geometric condition on circular proofs, which we called the translatability criterion. But it relies also on a property on the source and the target proof systems for the translation. We call this property, the **invariant property**. It consists in showing that a number of rules can be derived in $\mu\mathsf{S}^\omega$ and $\mu\mathsf{S}$, which will be helpful during the translation.

Definition 2.40. The proof systems $\mu\mathsf{S}$ and $\mu\mathsf{S}^\omega$ have the ***invariant property*** if for every occurrence $\nu X.F$ and for every set of occurrences Δ there is an occurrence $\mathcal{I}$, called the ***invariant of*** $(\nu X.F, \Delta)$, such that the following rules are derivable in $\mu\mathsf{S}$:

$$\frac{\vdash \Sigma[\nu X.F/X]}{\vdash \Sigma[\mathcal{I}/X]}\ (\mathsf{Subst}) \qquad \frac{\vdash \psi[\mathcal{I}/X], \Sigma}{\vdash \mathcal{I}, \Sigma}\ (\mathsf{Unfold}) \qquad \frac{}{\vdash \mathcal{I}, \Delta}\ (\mathsf{Close}) \qquad \frac{\vdash \Delta, F[\mathcal{I}/X]}{\vdash \mathcal{I}^\perp, F[\mathcal{I}/X]}\ (\mathsf{Replace})$$

And the following rules are derivable in $\mu\mathsf{S}^\omega$:

$$\frac{\vdash \Sigma[\nu X.F/X]}{\vdash \Sigma[\mathcal{I}/X]}\ (\mathsf{Subst}^\omega) \qquad \frac{\vdash F[\mathcal{I}/X], \Sigma}{\vdash \mathcal{I}, \Sigma}\ (\mathsf{Unfold}^\omega)$$

In the rules (Subst), (Unfold), (Subst$^\omega$) and (Unfold$^\omega$), $\mathcal{I}$ behaves as if it was $\nu X.F$. It behaves as if it was $(\vee\Delta)^\perp$ in the rules (Close) and (Replace).

Proposition 2.15. *For every* $\mathsf{S} \in \{\mathsf{MALL}, \mathsf{LL}, \mathsf{LK}, \mathsf{LK}\odot\}$*, the proof systems* $\mu\mathsf{S}$ *and* $\mu\mathsf{S}^\omega$ *have the invariant property.*

Proof. When S is LK or LK$\odot$, we set $\mathcal{I} = \nu X.F \vee \kappa$ where $\kappa = (\vee\Delta)^\perp$. In μLK and $\mu\mathsf{LK}\odot$, we derive the rule (Subst) as follows.

$$\dfrac{\dfrac{\vdash \Sigma[\nu X.F/X]}{\vdash (\vee\Sigma)[\nu X.F/X]}\ (\vee) \qquad \dfrac{\dfrac{\dfrac{}{\vdash (\nu X.F)^\perp, \nu X.F}\ (\mathsf{Ax}) \qquad \dfrac{\dfrac{\dfrac{}{\vdash F^\perp[(\nu X.F)^\perp/X], F[\nu X.F/X], \kappa}\ (\mathsf{Ax})}{\vdash F^\perp[(\nu X.F)^\perp/X], F[\nu X.F/X] \vee \kappa}\ (\vee)}{\vdash (\nu X.F)^\perp, F[\nu X.F/X] \vee \kappa}\ (\mu)}{\vdash (\nu X.F)^\perp, \mathcal{I}}\ (\nu)}{\vdash (\vee\Sigma)^\perp[(\nu X.F)^\perp/X], \Sigma[\mathcal{I}/X]}\ (\wedge), \{(\mathsf{F}_F),\ F \in \Sigma\}}{\vdash \Sigma[\mathcal{I}/X]}\ (\mathsf{Cut})$$

In $\mu\mathsf{LK}^\omega$ and $\mu\mathsf{LK}\odot^\omega$, we derive the rule ($\mathsf{Subst}^\omega$) in the same way as ($\mathsf{Subst}$), except that we use the infinitary version of the functoriality ($\mathsf{F}^\omega_\mathsf{F}$) instead of ($\mathsf{F}_\mathsf{F}$).
In $\mu\mathsf{LK}$ and $\mu\mathsf{LK}\odot$, the rule (Unfold) can be derived as follows:

$$\dfrac{\dfrac{\dfrac{\vdash F[\mathcal{I}/X], \Sigma}{\vdash F[\mathcal{I}/X], \kappa, \Sigma}\,(\mathsf{W})}{\vdash F[\mathcal{I}/X] \vee \kappa, \Sigma}\,(\vee)}{\vdash \mathcal{I}, \Sigma}\,(\nu)$$

In $\mu\mathsf{LK}^\omega$ and $\mu\mathsf{LK}\odot^\omega$, the rule ($\mathsf{Unfold}^\omega$) can be derived in the same way as (Unfold).
For the rule (Close) we proceed as follows:

$$\dfrac{\dfrac{}{\vdash F \vee (\vee\Delta)^\perp, \Delta}\,(\vee),\,(\wedge),\,(\mathsf{Ax})}{\vdash \mathcal{I}, \Delta}\,(\nu_\mathsf{u})$$

Finally, for the rule ($\mathsf{Replace}$), we use the following derivation:

$$\dfrac{\dfrac{\dfrac{}{\vdash F^\perp[\mathcal{I}^\perp/X], F[\mathcal{I}/X]}\,(\mathsf{Ax}) \qquad \dfrac{\vdash \Delta, F[\mathcal{I}/X]}{\vdash \kappa^\perp, F[\mathcal{I}/X]}\,(\vee)}{\vdash F^\perp[\mathcal{I}^\perp/X] \wedge \kappa^\perp, F[\mathcal{I}/X]}\,(\wedge)}{\vdash \mathcal{I}^\perp, F[\mathcal{I}/X]}\,(\mu)$$

If S is MALL or LL we set $\mathcal{I} = \nu X.F \oplus \kappa$ where $\kappa = (\parr\Delta)^\perp$. In $\mu\mathsf{MALL}$ and $\mu\mathsf{LL}$, we derive the rule (Subst) as follows.

$$\dfrac{\dfrac{\vdash \Sigma[\nu X.F/X]}{\vdash (\parr\Sigma)[\nu X.F/X]}\,(\parr) \qquad \dfrac{\dfrac{}{\vdash (\nu X.F)^\perp, \nu X.F}\,(\mathsf{Ax}) \qquad \dfrac{\dfrac{\dfrac{}{\vdash F^\perp[(\nu X.F)^\perp/X], F[\nu X.F/X]}\,(\mathsf{Ax})}{\vdash F^\perp[(\nu X.F)^\perp/X], F[\nu X.F/X] \oplus \kappa}\,(\oplus_1)}{\vdash (\nu X.F)^\perp, F[\nu X.F/X] \oplus \kappa}\,(\mu)}{\vdash (\nu X.F)^\perp, \mathcal{I}}\,(\nu)}{\vdash (\parr\Sigma)^\perp[(\nu X.F)^\perp/X], \Sigma[\mathcal{I}/X]}\,(\otimes),\,\{(\mathsf{F}_\mathsf{F}),\, F \in \Sigma\}}{\vdash \Sigma[\mathcal{I}/X]}\,(\mathsf{Cut})$$

In $\mu\mathsf{MALL}^\omega$ and $\mu\mathsf{LL}^\omega$, we derive the rule (Subst^ω) in the same way as (Subst), except that we use the infinitary version of the functoriality ($\mathsf{F}^\omega_\mathsf{F}$) instead of ($\mathsf{F}_\mathsf{F}$).
In $\mu\mathsf{MALL}$ and $\mu\mathsf{LL}$, the rule (Unfold) can be derived as follows:

$$\dfrac{\dfrac{\dfrac{\vdash F[\mathcal{I}/X], \Sigma}{\vdash F[\mathcal{I}/X] \oplus \kappa, \Sigma}\,(\oplus_1)}{\vdash \mathcal{I}, \Sigma}\,(\nu_\mathsf{u})$$

In $\mu\mathsf{MALL}^\omega$ and $\mu\mathsf{LL}^\omega$, the rule (Unfold^ω) can be derived in the same way as (Unfold).
For the rule (Close) we proceed as follows:

$$\dfrac{\dfrac{}{\vdash F \oplus (\parr\Delta)^\perp, \Delta}\,(\oplus_2),\,(\otimes),\,(\mathsf{Ax})}{\vdash \mathcal{I}, \Delta}\,(\nu_\mathsf{u})$$

Finally, for the rule (Replace), we use the following derivation:

$$\dfrac{\dfrac{\dfrac{}{\vdash F^\perp[\mathcal{I}^\perp/X], F[\mathcal{I}/X]}\,(\mathsf{Ax}) \qquad \dfrac{\vdash \Delta, F[\mathcal{I}/X]}{\vdash \kappa^\perp, F[\mathcal{I}/X]}\,(\parr)}{\vdash F^\perp[\mathcal{I}^\perp/X] \,\&\, \kappa^\perp, F[\mathcal{I}/X]}\,(\&)}{\vdash \mathcal{I}^\perp, F[\mathcal{I}/X]}\,(\mu)$$

We neglected the addresses in these derivations since they are easily inferable from the context. □

We will use (Subst), (Unfold), (Close) and (Replace) as proper rules of $\mu\mathsf{S}$, and (Subst^ω) and (Unfold^ω) as proper rules of $\mu\mathsf{S}^\omega$.

Some surgery on circular proofs

To translate a circular proof, we will proceed inductively, cutting it into smaller pieces, translating each piece into a finitary proof and gathering the translated proofs into one finitary proof which will be the translation of the initial circular proof.

We introduce in this section these different operations. The first one consists in cutting a circular proof into smaller pieces. At the level of the vertex on which we cut the proof, the well-typedness condition is not satisfied anymore since the successors of this vertex are gone with the other parts of the proof. To fix that, we replace the rule of the vertex at the level of which we cuted the proof by a new 0-ary rule which we call the **assumption rule** and which we denote by (A). Therefore, during our transformation steps, we will step out of the proof system $\mu\mathsf{S}^\omega$, and fall in the extension of $\mu\mathsf{S}^\omega$ that contains the rule (A), which we denote by $\mu\mathsf{S}_\mathsf{A}{}^\omega$. The translations of $\mu\mathsf{S}_\mathsf{A}{}^\omega$ proofs will also contain the rule (A), we denote by $\mu\mathsf{S}_\mathsf{A}$ the finitary proof system that extends $\mu\mathsf{S}$ with the rule (A). We introduce the assumption rule (A) and the proof systems $\mu\mathsf{S}_\mathsf{A}{}^\omega$ and $\mu\mathsf{S}_\mathsf{A}$ in the following.

Definition 2.41. S_A is the proof system obtained from S by adding the following rule of arity 0, called ***assumption rule***:

$$\dfrac{}{\vdash \Gamma}\,(\mathsf{A})$$

The finitary and the circular extensions of S_A with least and greatest fixed points are respectively $\mu\mathsf{S}_\mathsf{A}$ and $\mu\mathsf{S}_\mathsf{A}{}^\omega$. If Π is a $\mu\mathsf{S}_\mathsf{A}$ or a $\mu\mathsf{S}_\mathsf{A}{}^\omega$ proof, we denote by $\mathcal{A}_\Pi$ the sequents conclusion of an assumption rule in Π.

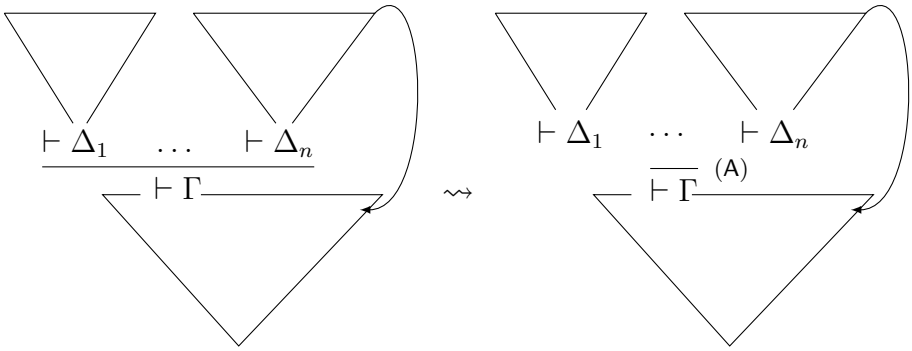
We introduce now three operations on proofs. The first two are operations on circular proofs, they consist respectively in adding a new assumption and taking the reachable part. Put together, they will allow us in the translation procedure to cut a circular proof into smaller pieces. The third operation glues finitary proofs at the level of an assumption. This operation will be used to gather the translations of the smaller pieces of a circular proof.

Addition of new assumptions.

Definition 2.42. Let $\Pi = (P, c)$ be a circular proof, where $P = (G, \lambda, \rho, \sigma)$, and let $v \in G$. We define Π^v to be the circular proof $((G, \lambda, \rho', \sigma), c)$, where

$$\begin{array}{rcll} \rho'(g) & = & \rho(g) & \text{if } g \in G \setminus \{v\} \\ & = & (\mathsf{A}) & \text{if } v \in A \end{array}$$

We draw this operation as follows:

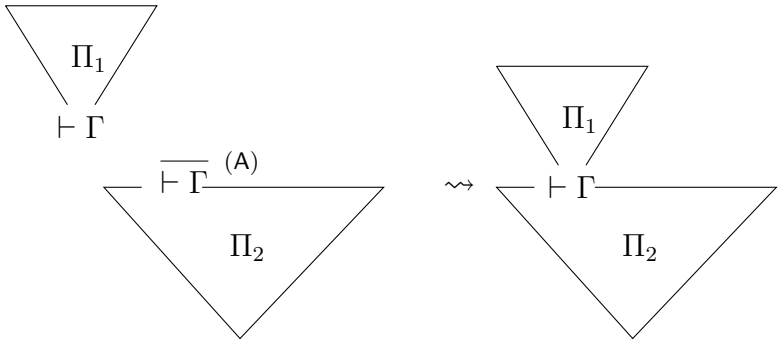


Taking the reachable part.

Definition 2.43. Let $\Pi = (P, c)$ be a circular proof, where $P = (G, \lambda, \rho, \sigma)$. Let $g \in G$. We define $\overline{\Pi, g}$ to be the circular proof (P', g) where $P' = (H, \lambda', \rho', \sigma')$ and $h \in H$ if and only if $g \to^\star h$ in $\mathcal{G}(\Pi)$; λ', ρ', σ' are the restrictions of λ, ρ, σ respectively to H.

Gluing two proofs

Definition 2.44. Let Π_1 be a $\mu\mathsf{S}_\mathsf{A}$ proof of conclusion s and let Π_2 be a $\mu\mathsf{S}_\mathsf{A}$ proof where the sequent s appears as the conclusion of an assumption rule. We define $\mathsf{Glue}(\Pi_1, \Pi_2, s)$ to be the $\mu\mathsf{S}_\mathsf{A}$ proof obtained from Π_1 by replacing the assumption on s by the proof Π_2. We draw this operation as follows:



The procedure

The main result of this section is the following.

Theorem 2.5. *Let Π be a $\mu\mathsf{S}^\omega$ proof of a sequent s. If Π is translatable and if $\mu\mathsf{S}$ and $\mu\mathsf{S}^\omega$ satisfy the invaiant property, then we can construct effectively a proof of s in $\mu\mathsf{S}$.*

The proof of this result is inspired by a proof technique used by Santocanale in [San02]. In this paper, Santocanale defines a categorical semantics for his calculus (which is the circular proof system for additive linear logic: $\mu\mathsf{ALL}^\omega$). For that, he associates to every circular proof of his calculus a system of equations, then he shows that every such system admits a unique solution. To prove this last result, he proceeds by induction on what he

calls the "complexity" of his circular proof: he decomposes it into proofs having smaller complexity, the system of equations associated to each of these proofs admits a unique solution by induction hypothesis, he gathers these to solutions together and shows that the obtained object is a solution of the initial system of equations. We can see his proof as a translation procedure from circular proofs into his semantics. Our idea was to follow the same proof schema, but instead of translating towards semantics, we translate our circular proofs towards the finitary proof system. His calculus being purely additive, his technique does not scale as it is in a more general setting where multiplicative rules may occur. For instance his measure of complexity does not work in general.

Let us define now our measure of complexity. As said earlier, we will step out the system $\mu\mathsf{S}^\omega$ during this induction, thus we define below the complexity for $\mu\mathsf{S}_\mathsf{A}{}^\omega$ proofs.

Definition 2.45. Let Π be a $\mu\mathsf{S}_\mathsf{A}{}^\omega$ proof. We define the set $\mathcal{V}_\Pi$ to be the set of vertices of Π not labeled by an assumption rule. We define $\mathcal{C}_\Pi$ to be the set of elementary cycles of $\mathcal{G}(\Pi)$. The ***complexity of*** Π, denoted $\#\Pi$ is the pair $(\mathsf{card}\ \mathcal{C}_\Pi, \mathsf{card}\ \mathcal{V}_\Pi)$, of the cardinal of $\mathcal{C}_\Pi$ and the cardinal of $\mathcal{V}_\Pi$.

Definition 2.46. The set $\omega \times \omega$ is naturally equipped with the ***lexicographic order*** $\preceq$ defined by: $(n, m) \preceq (n', m')$ if and only if $n \leq n'$ and if $n = n'$ then $m \leq m'$. We denote by $\prec$ the strict order arising from $\preceq$.

Since $\prec$ is a well-founded relation, we will show our result by induction on $\#\Pi$, providing a base case if $\mathsf{card}\ \mathcal{C}_\Pi = 0$ and an induction step when $\mathsf{card}\ \mathcal{C}_\Pi > 0$.
We introduce the last definition before showing our main result. In the inductive case, we will need to consider two situations: when the proof is **strongly connected** and when it is not.

Definition 2.47. A $\mu\mathsf{S}_\mathsf{A}{}^\omega$ proof Π is said to be ***strongly connected***, if, for each pair $g_1, g_2 \in \mathcal{V}_\Pi$, we can find paths from g_1 to g_2, and from g_2 to g_1 in $\mathcal{G}(\Pi)$.

To show Theorem 2.5, We establish the following strengthened result:

Proposition 2.16. *If $\mu\mathsf{S}$ and $\mu\mathsf{S}^\omega$ satisfy the invariant property, and if Π is a translatable $\mu\mathsf{S}_\mathsf{A}{}^\omega$ proof of $\vdash \Gamma$, then there is a $\mu\mathsf{S}_\mathsf{A}$ proof π of $\vdash \Gamma$ such that $\mathcal{A}_\pi \subseteq \mathcal{A}_\Pi$.*

Proof. The proof is by induction on the complexity of Π. When

$$\mathsf{card}\ \mathcal{C}_\Pi = 0$$

This means that the graph of Π has the shape of a tree. The rules used in Π are of three kinds: either 1) $\mu\mathsf{S}^\infty$ rules, 2) Assumption rule 3) One of the rules (Subst^ω) or (Unfold^ω). The translation of Π is the $\mu\mathsf{S}_\mathsf{A}$ proof π obtained from Π by keeping the rules of kind 1) and 2) unchanged and by replacing the rules (Subst^ω) or (Unfold^ω) by their finitary versions (Subst) or (Unfold). We have obviously that $\mathcal{A}_\pi \subseteq \mathcal{A}_\Pi$.
Otherwise:

$$\mathsf{card}\ \mathcal{C}_\Pi > 0$$

and Π is either strongly connected or not. Therefore, it is possible to argue as follows.

If Π is not strongly connected. Let $g_1, g_2 \in \mathcal{V}_\Pi$ and suppose that there is no path from g_1 to g_2 in $\mathcal{G}(\Pi)$. Let Π_1 be the reachable part of Π from g_1:

$$\Pi_1 = \overline{\Pi, g_1}$$

and let Π_2 be the proof obtained from Π by adding an assumption on g_1 and taking the reachable part from the conclusion c of Π:

$$\Pi_2 = \overline{\Pi^{g_1}, c}$$

Let $\vdash \Delta$ be the label of g_1, Π_1 is thus a proof of $\vdash \Delta$ and Π_2 a proof of $\vdash \Gamma$. Note that $\mathcal{A}_{\Pi_1} \subseteq \mathcal{A}_\Pi$ and $\mathcal{A}_{\Pi_2} \subseteq \mathcal{A}_\Pi \cup \{\vdash \Delta\}$.

Since g_2 is not reachable from g_1, Π_1 does not contain g_2 as a vertex, thus $\mathsf{card}\ \mathcal{V}_{\Pi_1} < \mathsf{card}\ \mathcal{V}_\Pi$. Since g_1 is an assumption in Π_2, $\mathsf{card}\ \mathcal{V}_{\Pi_2} < \mathsf{card}\ \mathcal{V}_\Pi$. Thus by induction hypothesis, there are two $\mu\mathsf{S}_\mathsf{A}$ proofs π_1 and π_2, respectively of conclusion $\vdash \Delta$ and $\vdash \Gamma$, such that $\mathcal{A}_{\pi_1} \subseteq \mathcal{A}_{\Pi_1}$ and $\mathcal{A}_{\pi_2} \subseteq \mathcal{A}_{\Pi_2}$.

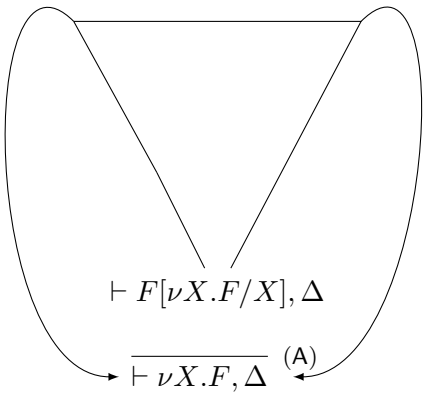
Let $\pi = \mathsf{Glue}(\Pi_1, \Pi_2, \vdash \Delta)$. We have that $\mathcal{A}_\pi = (\mathcal{A}_{\pi_1} \cup \mathcal{A}_{\pi_1} \setminus \{\vdash \Delta\}) \subseteq \mathcal{A}_\Pi$, which concludes this case.

If Π is strongly connected. There is an infinite path p that visits all the vertices of $\mathcal{G}(\Pi)$. Since Π is translatable, this path is validated by a strongly valid trace t. Let g be a vertex where the minimal formula of t has been unfolded, that is $\rho(g) = (\nu)$, $\lambda(g) = (\vdash \nu X.F, \Delta)$, where $\nu X.F \equiv \mathsf{min}(\mathsf{Inf}(\bar{t}))$ and the successor of g, that we denote by g', is labeled by the sequent $\vdash F[\nu X.F/X], \Delta$.

We suppose without loss of generality that g is the conclusion of Π. Let Π' be the proof of conclusion g' obtained from Π by adding an assumption on g:

$$\Pi' = \overline{\Pi^g, g'}$$

Graphically, Π' is of the following shape:



Let $\mathcal{I}$ be the invariant of $(\nu X.F, \Delta)$ (Definition 2.40). We construct the circular proof Θ from Π' by replacing the label $\vdash F[\nu X.F/X], \Delta$ of its conclusion g' by $\vdash F[\mathcal{I}/X], \Delta$, and simply propagating that substitution along the trace t, unfolding $\mathcal{I}$ when the corresponding φ is unfolded by using the rule (Unfold). Let us show how to propagate this substitution only along the trace t whilst preventing it from affecting other occurrences.

By strong validity, if the trace t meets a vertex v, it meets it always at the level of the same occurrence that we denote by $t(v)$. Since $\overline{\nu X.F}$ is the minimal formula of the trace t, every occurrence $t(v)$ is of the form $G[\nu X.F/X]$. We apply the following transformation on the labels of Π' vertices starting from the conclusion:

Let v be a vertex of Π' and let $\{v_i\}_{i\in I}$ be the set of its successors. Each v_i is either touched by the trace, or it is not (it is an assumption then). Let $\{v_j\}_{j\in J}$ be the successors of v of the first category and $\{v_k\}_{k\in K}$ be its successors of the second one.

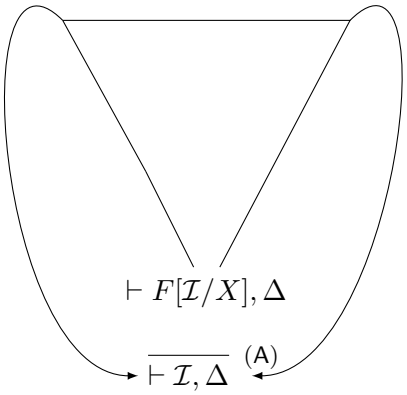
The label of v is of the form $G[\nu X.F/X], \Sigma$ where $G[\nu X.F/X] = t(v)$. For every $j \in J$, the label of v_j is of the form $G_j[\nu X.F/X], \vec{H_j}[\nu X.F/X], \Sigma_j$ where $G_j[\nu X.F/X] = t(v_j)$ and $\vec{H_j}[\nu X.F/X]$ are sub-occurrences of $G[\nu X.F/X]$. For every $k \in K$, the label of v_k is of the form $\vec{H_k}[\nu X.F/X], \Sigma_k$ where $\vec{H_k}[\nu X.F/X]$ are sub-occurrences of $G[\nu X.F/X]$. We show below the inference rule corresponding to the vertex v:

$$\frac{\{\vdash G_j[\nu X.F/X], \vec{H_j}[\nu X.F/X], \Sigma_j\}_{j\in J} \cup \{\vdash \vec{H_k}[\nu X.F/X], \Sigma_k\}_{k\in K}}{\vdash G[\nu X.F/X], \Sigma}\ \rho(v)$$

When we substitute $\nu X.F$ by $\mathcal{I}$ in $t(v)$, we want this substitution to reach only $t(v_j)$ and not $\vec{H_j}[\nu X.F/X]$, nor $\vec{H_j}[\nu X.F/X]$. We use for that the rule (Subst) as follows:

$$\frac{\left\{\ \dfrac{\vdash G_j[\mathcal{I}/X], \vec{H_j}[\nu X.F/X], \Sigma_j}{\vdash G_j[\mathcal{I}/X], \vec{H_j}[\mathcal{I}/X], \Sigma_j}\ (\mathsf{Subst})\ \right\}_{j\in J} \bigcup \left\{\ \dfrac{\vdash \vec{H_k}[\nu X.F/X], \Sigma_k}{\vdash \vec{H_k}[\mathcal{I}/X], \Sigma_k}\ (\mathsf{Subst})\ \right\}_{k\in K}}{\vdash G[\mathcal{I}/X], \Sigma}\ \rho(v)$$

Note that the occurrence $\nu X.F$ in the vertex g will become $\mathcal{I}$ in Θ, we depict the proof Θ as follows:



The complexity of the proof Θ is strictly less than that of Π, since it contains strictly less elementary cycles. Note also that Θ is still translatable, thus we can apply the induction hypothesis to Θ, and get a $\mu\mathsf{S_A}$ proof θ of the sequent $\vdash F[\mathcal{I}/X], \Delta$ such that $\mathcal{A}_\theta \subseteq \mathcal{A}_\Theta$.

The proof Θ has one more assumption than Π which is the assumption on $\vdash \mathcal{I}, \Delta$, thus θ may also contain this assumption. We adapt θ to get rid of this assumption in order to obtain a $\mu\mathsf{S_A}$ derivation β such that $\mathcal{A}_\beta \subseteq \mathcal{A}_\Pi$. For that we replace the assumption on $\vdash \mathcal{I}, \Delta$ by the rule (Close). The obtained $\mu\mathsf{S_A}$ proof β satisfies $\mathcal{A}_\beta \subseteq \mathcal{A}_\Pi$ and its conclusion is $\vdash F[\mathcal{I}/X], \Delta$. We are now ready to conclude by constructing a $\mu\mathsf{S_A}$ proof π of conclusion

$\vdash \nu X.F, \Delta$ such that $\mathcal{A}_\pi \subseteq \mathcal{A}_\Pi$. The derivation starts with a (ν) rule on $\nu X.F$, using $\mathcal{I}$ as invariant.

$$\pi \quad = \quad \dfrac{\dfrac{}{\vdash \Delta, \mathcal{I}}\ (\mathsf{Close}) \qquad \dfrac{\dfrac{\beta}{\vdash F[\mathcal{I}/X], \Delta}}{\vdash F[\mathcal{I}/X], \mathcal{I}^\perp}\ (\mathsf{Replace})}{\vdash \nu X.F, \Delta}\ (\nu)$$

□

Remark 2.21. From the proof of Proposition 2.16, we can extract an algorithm that transforms a circular proof into a finitary one, provided that we have an algorithm that returns for every regular path a strongly valid trace for it. Indeed, the "strongly connected case" of the procedure requires to find a strongly valid trace for the path that visits all the sequents. This should be done in an effective way to get a translation algorithm.

We state in the following a sufficient condition on circular proofs, which allows to transform the procedure of the proof of Proposition 2.16 into an effective algorithm.

Proposition 2.17. *Let* Π *be a* $\mu\mathsf{S}^\omega$ *proof of* s*. If every valid trace of* Π *is strongly valid, then we can transform effectively* Π *into a* $\mu\mathsf{S}$ *proof of* s*.*

Proof. By Remark 2.21, to turn the translation procedure described in the proof of Proposition 2.16 into an effective algorithm, one should provide an algorithm that returns for every regular path a strongly valid trace. Since all the traces of Π are strongly valid, it is enough to provide an algorithm that returns a valid trace for every regular path. For that, we design for every regular path p a parity word automaton $\mathcal{A}_p$ whose language is the set of all valid traces of p. Then we check $\mathcal{A}_p$ for emptiness, using for example Ramsey based techniques [FL12]. These emptiness tests either answer that the automaton is empty, or returns a regular word in its language if this is not the case. In our situation, the emptiness test will always return a word which is a valid tarce for p. □

Example 2.19. Let us consider the following $\mu\mathsf{MALL}^\omega$ proof, where we have:

$$\begin{array}{lcllcl} \varphi & = & \nu X.\psi & \qquad \psi & = & \kappa \parr \xi \\ \kappa & = & \mu Y.X \otimes Y & \qquad \xi & = & \sigma \oplus \top \\ \sigma & = & \nu Z.Z \oplus \bot & & & \end{array}$$

$$\mathcal{G}(\Pi, a) \quad = \quad \dfrac{\dfrac{\dfrac{(\star)}{\vdash \varphi, \xi} \qquad \dfrac{\dfrac{\dfrac{(\dagger)}{\vdash \kappa[\varphi/X], \sigma}}{\vdash \kappa[\varphi/X], \sigma \oplus \bot}\ (\oplus_1)}{(\dagger) \vdash \kappa[\varphi/X], \sigma}\ (\nu)}{\vdash \varphi \otimes \kappa[\varphi/X], \xi, \sigma}\ (\otimes)}{\vdash \kappa[\varphi/X], \xi, \sigma}\ (\mu)}{\vdash \psi[\varphi/X], \sigma}\ (\parr)}{\vdash \psi[\varphi/X], \xi}\ (\oplus_1)}{(\star) \vdash \varphi, \xi}\ (\nu)$$

In this example, we will neglect the use of addresses since they are unambiguously inferable from the context.

We show the different steps of the algorithm described above when applied to this proof. The proof Π is not strongly connected, we cut it into the following two proofs:

$$\Pi_1 \quad = \quad \dfrac{\dfrac{\dfrac{}{\vdash \kappa[\varphi/X], \sigma}(\dagger)}{\vdash \kappa[\varphi/X], \sigma \oplus \bot}(\oplus_1)}{(\dagger) \vdash \kappa[\varphi/X], \sigma}(\nu)$$

$$\Pi_2 \quad = \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash \varphi, \xi}(\star) \qquad \dfrac{}{\vdash \kappa[\varphi/X], \sigma}(\mathsf{A})}{\vdash \varphi \otimes \kappa[\varphi/X], \xi, \sigma}(\otimes)}{\vdash \kappa[\varphi/X], \xi, \sigma}(\mu)}{\vdash \psi[\varphi/X], \sigma}(\mathfrak{P})}{\vdash \psi[\varphi/X], \xi}(\oplus_1)}{(\star) \vdash \varphi, \xi}(\nu)$$

Now we will translate each of these proofs.

Translation of Π_1. The proof Π_1 is strongly connected and it has only one infinite path with exactly one strongly valid trace (drawn in blue), whose minimal formula is σ. The derivation Π_1' is the immediate unfolding of Π_1, where we added an assumption on the vertex labeled $\vdash \kappa[\varphi/X], \sigma$.

$$\Pi_1' \quad = \quad \dfrac{\dfrac{}{\vdash \kappa[\varphi/X], \sigma}(\mathsf{A})}{\vdash \kappa[\varphi/X], \sigma \oplus \bot}(\oplus_1)$$

Let $\mathcal{I}_1$ be the invariant of $(\sigma, \{\kappa[\varphi/X]\})$. The proof Θ_1 is obtained by replacing σ by $\mathcal{I}_1$ in the conclusions of Π_1' and propagating this substitution along the strongly valid trace:

$$\Theta_1 \quad = \quad \dfrac{\dfrac{}{\vdash \kappa[\varphi/X], \mathcal{I}_1}(\mathsf{A})}{\vdash \kappa[\varphi/X], \mathcal{I}_1 \oplus \bot}(\oplus)$$

Since $\mathcal{G}(\Theta_1)$ is a tree, we have that $\mathcal{C}_{\Theta_1} = 0$, thus its translation θ_1 is Θ_1 itself. Now, we eliminate the assumption from θ_1 using the rule (Close), doing so we get the following μMALL proof β_1:

$$\beta_1 \quad = \quad \dfrac{\dfrac{}{\vdash \kappa[\varphi/X], \mathcal{I}_1}(\mathsf{Close})}{\vdash \kappa[\varphi/X], \mathcal{I}_1 \oplus \bot}(\oplus)$$

Now we can construct π_1, the translation of Π_1, by applying a rule (ν) on σ using $\mathcal{I}_1$ as invariant.

$$\pi_1 \quad = \quad \dfrac{\dfrac{}{\vdash \kappa[\varphi/X], \mathcal{I}_1}(\mathsf{Close}) \qquad \dfrac{\dfrac{\beta_1}{\vdash \kappa[\varphi/X], \mathcal{I}_1 \oplus \bot}}{\vdash \mathcal{I}_1^{\bot}, \mathcal{I}_1 \oplus \bot}(\mathsf{Replace})}{\vdash \kappa[\varphi/X], \sigma}(\nu)$$

Translation of Π_2. The proof Π_2 is strongly connected and it has only one infinite path, with exactly one strongly valid trace (drawn in red), whose minimal formula is φ.

The derivation Π_2' is the immediate unfolding of Π_2, where we added an assumption on the vertex labeled $\vdash \varphi, \xi$.

$$\Pi_2' \quad = \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash \varphi, \xi}\,(\mathsf{A}) \qquad \dfrac{}{\vdash \kappa[\varphi/X], \sigma}\,(\mathsf{A})}{\vdash \varphi \otimes \kappa[\varphi/X], \xi, \sigma}\,(\otimes)}{\vdash \kappa[\varphi/X], \xi, \sigma}\,(\mu)}{\vdash \psi[\varphi/X], \sigma}\,(\parr)}{\vdash \psi[\varphi/X], \xi}\,(\oplus_1)$$

Let $\mathcal{I}_2$ be the invariant of $(\varphi, \{\xi\})$. The proof Θ_2 is obtained by replacing φ by $\mathcal{I}_2$ in the conclusions of Π_2' and propagating this substitution all along the strongly valid trace:

$$\Theta_2 \quad = \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash \mathcal{I}_2, \xi}\,(\mathsf{A}) \qquad \dfrac{\dfrac{}{\vdash \kappa[\varphi/X], \sigma}\,(\mathsf{A})}{\vdash \kappa[\mathcal{I}_2/X], \sigma}\,(\mathsf{Subst})}{\vdash \mathcal{I}_2 \otimes \kappa[\mathcal{I}_2/X], \xi, \sigma}\,(\otimes)}{\vdash \kappa[\mathcal{I}_2/X], \xi, \sigma}\,(\mu)}{\vdash \psi[\mathcal{I}_2/X], \sigma}\,(\parr)}{\vdash \psi[\mathcal{I}_2/X], \xi}\,(\oplus_1)$$

Since Θ_2 is a tree, its translation θ_2 is Θ_2 itself. The proof θ_2 has one more assumption than Π_2, which is $\vdash \mathcal{I}_2, \xi$. We eliminate this additional assumption using the rule (Close). Doing so, we obtain the derivation β_2:

$$\beta_2 \quad = \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash \mathcal{I}_2, \xi}\,(\mathsf{Close}) \qquad \dfrac{\dfrac{}{\vdash \kappa[\varphi/X], \sigma}\,(\mathsf{A})}{\vdash \kappa[\mathcal{I}_2/X], \sigma}\,(\mathsf{Subst})}{\vdash \mathcal{I}_2 \otimes \kappa[\mathcal{I}_2/X], \xi, \sigma}\,(\otimes)}{\vdash \kappa[\mathcal{I}_2/X], \xi, \sigma}\,(\mu)}{\vdash \psi[\mathcal{I}_2/X], \sigma}\,(\parr)}{\vdash \psi[\mathcal{I}_2/X], \xi}\,(\oplus_1)$$

Now we can construct π_2, the translation of Π_2, by applying a rule (ν) on φ using $\mathcal{I}_2$ as invariant.

$$\pi_2 \quad = \quad \dfrac{\dfrac{}{\vdash \xi, \mathcal{I}_2}\,(\mathsf{Close}) \qquad \dfrac{\dfrac{\beta_2}{\vdash \psi[\mathcal{I}_2/X], \xi}}{\vdash \psi[\mathcal{I}_2/X], \mathcal{I}_2^{\perp}}\,(\mathsf{Replace})}{\vdash \varphi, \xi}\,(\nu)$$

Translation of Π. Now that we have the translation π_1 of Π_1 and π_2 of Π_2, we can glue them at the level of the assumption $\vdash \kappa[\varphi/X], \sigma$, to get the following proof, which is the

translation of Π.

$$\dfrac{\dfrac{}{\vdash \xi, \mathcal{I}_2}\,(\mathsf{Close}) \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash \mathcal{I}_2, \xi}\,(\mathsf{Close}) \qquad \dfrac{\dfrac{\dfrac{}{\vdash \kappa[\varphi/X], \mathcal{I}_1}\,(\mathsf{Close})}{\vdash \kappa[\varphi/X], \mathcal{I}_1 \oplus \bot}\,(\oplus)}{\vdash \mathcal{I}_1^{\bot}, \mathcal{I}_1 \oplus \bot}\,(\mathsf{Replace})}{\vdash \kappa[\varphi/X], \sigma}\,(\nu)}{\vdash \kappa[\mathcal{I}_2/X], \sigma}\,(\mathsf{Subst})}{\vdash \mathcal{I}_2 \otimes \kappa[\mathcal{I}_2/X], \xi, \sigma}\,(\otimes)}{\vdash \kappa[\mathcal{I}_2/X], \xi, \sigma}\,(\mu)}{\vdash \psi[\mathcal{I}_2/X], \sigma}\,(\parr)}{\vdash \psi[\mathcal{I}_2/X], \xi}\,(\oplus_1)}{\vdash \psi[\mathcal{I}_2/X], \mathcal{I}_2^{\bot}}\,(\mathsf{Replace})}{\vdash \varphi, \xi}\,(\nu)$$

Part I

Linear logic with least and greatest fixed points

Chapter 3

Cut-elimination for $\mu\mathsf{MALL}^\infty$

In this Chapter we establish the cut-elimination result for the infinitary proof system $\mu\mathsf{MALL}^\infty$.

Why cut-elimination for $\mu\mathsf{MALL}^\infty$ instead of $\mu\mathsf{MALL}^\omega$? Of course, from a Curry-Howard perspective, we are interested in the cut-elimination result for the circular proof system $\mu\mathsf{MALL}^\omega$, but we cannot expect to achieve such a result. Indeed, in Section 3.6 we will show that we can encode in $\mu\mathsf{MALL}^\omega$ the function that takes a natural number n as an argument and returns the stream $S_n = n :: (n+1) :: n+2 :: \dots$. When we cut the encoding of this function against the $\mu\mathsf{MALL}^\omega$ proof encoding a natural number n, the cut-elimination procedure yields the infinitary proof encoding the stream S_n. Thus, the cut-elimination for circular proofs does not preserve circularity. This is the reason why we do not restrict ourselves to $\mu\mathsf{MALL}^\omega$, and show the more general result of cut-elimination for $\mu\mathsf{MALL}^\infty$.

This remark on the loss of circularity should not be viewed as a weakness, quite the contrary: if circularity was preserved we would not be able to encode in $\mu\mathsf{MALL}^\omega$ the function $n \mapsto S_n$.

Peculiarity of cut-elimination in an infinitary setting. In a finitary setting, the cut-elimination result is usually stated as follows: "Every proof can be reduced after a finite number of steps to a cut-free proof". In an infinitary setting such as $\mu\mathsf{MALL}^\omega$, this result obviously does not hold. Returning to the example of the $\mu\mathsf{MALL}^\omega$ encoding of $n \mapsto S_n$, we cannot hope to reduce this proof into a the cut-free proof encoding S_n in a finite number of steps, simply because the result is not regular. Actually, what we need to establish it not the termination of the cut-elimination procedure, but its **productivity**. This property means that every finite prefix of the result can be computed in a finite number of steps. This allows us for example to inspect the stream S_n at any depth we need.

The other particularity of cut-elimination in an infinitary setting comes from the validity condition. If productivity guarantees that we obtain a pre-proof at the limit of the cut-elimination procedure, nothing says that it is indeed a proof. We will show in this chapter that the pre-proof obtained by cut elimination satisfies the validity condition, we call the result the **validity preservation**. The productivity being strongly based on the validity of the proof, the validity preservation result is of great importance when we consider higher-order functions, since the result of cut-elimination may be itself used as a function.

Finally, as usual with infinitary reductions it is not the case that all reduction sequences converge: for instance, one could reduce only deep cuts in a proof, leaving a cut untouched at the root. We avoid this problem by reducing bottom-most cuts only, that is we consider a form of head reduction.

About the proof. It seems difficult to lift the standard arguments used to show cut elimination in the finitary setting (for example induction on formulas and proofs or reducibility candidates) to the infinitary setting, since they rely strongly on the finiteness of the proofs. The only known cut-elimination result in an infinitary setting is the one concerning $\mu\mathsf{ALL}^\infty$ by Fortier and Santocanale [FS13]. To show this result, the authors use a topological argument on $\mu\mathsf{ALL}^\infty$ proofs which works only for the additive framework. No simple adaptation of this argument seems to be possible when multiplicative rules are also considered.

Let us sketch coarsely our argument. We proceed by contradiction, and suppose that there is a non productive sequence of cut-elimination reductions, starting from a $\mu\mathsf{MALL}^\infty$ proof Π. The sequents of Π that participated to this reduction are called its **trace**, and form a sub-derivation of Π. Pruned correctly, this trace yields a proof of the empty sequent, in a proof system very close to $\mu\mathsf{MALL}^\infty$ which we call the truncated proof system $\mu\mathsf{MALL}^\infty_\tau$. We equip $\mu\mathsf{MALL}^\infty$ formulas with a natural thruth semantics and show that $\mu\mathsf{MALL}^\infty_\tau$ is sound with its respect. This yields a contradiction with the fact that the empty sequent is derivable in $\mu\mathsf{MALL}^\infty_\tau$.

This argument is unusual when we know that the soundness of a proof system is usually obtained as a corollary of the cut-elimination result (the admissibility of the cut rule to be more precise), this is perfectly the opposite of our approach.

The validity preservation result is obtained using a similar argument.

Organization of the chapter. In Section 3.1, we introduce the cut-elimination reduction rules for $\mu\mathsf{MALL}^\infty$ and specify our strategy to eliminate cuts. We define in Section 3.2 the trace of a reduction sequence, and show that we can see it as a proper proof of a proof system that we call $\mu\mathsf{MALL}^\infty_\tau$. A coarse thruth semantics for $\mu\mathsf{MALL}$ formulas proofs is introduced in Section 3.3, and shown to be sound for $\mu\mathsf{MALL}^\infty_\tau$. The productivity of the cut-elimination procedure and the preservation of validity are shown respectively in Sections 3.4 and 3.5, using the technical tools introduced in earlier sections and following the proof sketch above. In Section 3.6, we show some examples of $\mu\mathsf{MALL}^\infty$ proofs and their cut-elimination. Finally, we discuss in Section 3.7 the extension to the cut-elimination result to $\mu\mathsf{LL}$, $\mu\mathsf{LK}$ and $\mu\mathsf{LK}\odot$.

3.1 Reduction rules

In this section we define the cut-elimination reduction rules for $\mu\mathsf{MALL}^\infty$ (Sections 3.1.1 3.1.2). These rules extend the well-known reduction rules for MALL with the auxiliary rule $(\mu)/(\nu)$ and the principal rules $(\nu)/(\mathsf{Cut})$ and $(\mu)/(\mathsf{Cut})$. In order to get our cut-elimination result, we cannot apply these rules any how. As already mentioned, the first constraint is a head-reduction strategy, imposed by the infinitary setting. To make our argument work, we also need another constraint called *fairness*, we introduce it formally in Section 3.1.3. In Section 3.1.4, we state our cut-elimination result and give a blueprint of the proof.

3.1.1 The multicut rule

Before giving the reduction rules for $\mu\mathsf{MALL}^\infty$, we need to solve a technicality relative to the commutation of cut rules. Indeed, what happens in the situation where the cut we are trying to reduce (the bottom-most) encounters another cut rule? If we just try to permute the two cuts (see reduction below), then the $(\mathsf{Cut})_2$ which is now the bottom most, is under the cut rule $(\mathsf{Cut})_1$ we should then permute them, and we will loop in this $(\mathsf{Cut})/(\mathsf{Cut})$ reduction sequence which is obviously not productive.

To overcome this problem, instead of permuting the two cuts, we *merge* them into one rule called ***multicut***.

Definition 3.1. Given two sequents s and s', we say that they are ***cut-connected on an occurrence*** F when $F \in s$ and $F^\perp \in s'$. We say that they are ***cut-connected*** when they are cut-connected for some F. We call ***cut net*** any *non-empty* set of sequents $\{s_i\}_i$ such that:

- Any sequents s_i and s_j are cut-connected on at most one occurrence.
- If s_i and s_j are cut-connected on F, and if s_j and s_k are cut-connected on G, then F and G are disjoint.
- The set $\{s_i\}_i$ is connected and acyclic with respect to the cut-connection relation.

The ***conclusion of a cut net*** $\{s_i\}_i$ is the sequent s made of all the occurrences F appearing in some s_i but such that no s_j is cut-connected to s_i on F.
We define the ***multicut*** rule (mcut) as shown below with conclusion s and premisses $\{s_i\}_i$, where the set $\{s_i\}_i$ is a cut net and s its conclusion.

$$\frac{s_1 \quad \dots \quad s_n}{s}\ (\mathsf{mcut})$$

Example 3.1. Let s be any sequent, then the following rule is a valid multicut rule:

$$\frac{s}{s}\ (\mathsf{mcut})$$

Let F, G, H, K and L be pairwise disjoint occurrences. We define $\mathcal{M}$ to be the follwoing cut net: $\{\vdash F, G,\ \vdash F^\perp, H, K,\ \vdash H^\perp,\ \vdash K^\perp,\ \vdash G^\perp, L\}$. The conclusion of $\mathcal{M}$ is L, thus its corresponding rules is:

$$\frac{\vdash F, G \quad \vdash F^\perp, H, K \quad \vdash H^\perp \quad \vdash K^\perp \quad \vdash G^\perp, L}{\vdash L}\ (\mathsf{mcut})$$

We now give examples of sets which are *not* cut nets. The empty set is not a multicut net. If F and G are two disjoint occurrences then $\{\vdash F, G,\ \vdash F^\perp, G^\perp\}$ is not a cut net, since its two sequents are cut-connected on both F and G, thus it does not satisfy the first condition. If H is another occurrence disjoint from F, the set $\{\vdash F^\perp,\ \vdash F, G,\ \vdash F, H\}$ not a cut net either since it does not satisfy the second condition. Finally $\{\vdash F,\ \vdash G\}$ is not a cut net because it is not connected, nor is $\{\vdash F, H,\ \vdash F^\perp, G,\ \vdash G^\perp, H^\perp\}$ because it is cyclic.

From now on we shall work with a new proof system $\mu\mathsf{MALL}^\infty_\mathsf{m}$, that extends $\mu\mathsf{MALL}^\infty$ with the multicut rule:

Definition 3.2. *A $\mu\mathsf{MALL}^\infty_\mathsf{m}$ derivation*, is a $\mu\mathsf{MALL}^\infty$ derivation in which the multicut rule may occur, though only at most once per branch. The notions of thread and validity are unchanged for $\mu\mathsf{MALL}^\infty_\mathsf{m}$.

In $\mu\mathsf{MALL}^\infty_\mathsf{m}$, we will only reduce multicuts, using the rules that we introduce in the next section.

3.1.2 Reduction rules

In Chapter 1, we defined our sequents as sets of disjoint occurrences. To guarentee this disjointness condition on sequents, we stated a simple condition saying that the addresses of the conculusion occurrences together with the cut occurrences should be pairwise disjoint. Althougt easy to state, this condition is too rigid. In particular, it is not stable by the cut-elimination reductions that we will show in this section. We need to relax it into the following condition, which guarantees that the proofs produced by the cut-elimination reductions (and also by the proof transformationss that we will introduce in Chapter 4) satisfy the disjointness condition on sequents as well:

Proviso 3.1. All along this Part, we suppose that our proofs satisfy the following conditions:

- Any two occurrences appearing in different branches must be disjoint except if the branches first differ right after a (&) inference.
- If φ_α and $\psi_{\alpha^\perp}$ occur in a proof, they must be the respective sub-occurrences of the formula occurrences F and $F^\perp$ introduced by a (Cut) rule.

Before introducing the reduction rules for $\mu\mathsf{MALL}^\infty_\mathsf{m}$, we need to introduce a few preliminary definitions, in order to treat well the principal reduction rule $(\otimes)/(\mathsf{Cut})$. Given a sequent $\vdash \Gamma, \Delta, F\otimes G$ that is a premise of a multicut, we need to define which part of the multicut is connected to Γ and which part is connected to Δ. These two sub-nets, respectively called $\mathcal{C}_\Gamma$ and $\mathcal{C}_\Delta$, will be split apart in the $(\otimes)/(\mathsf{Cut})$ reduction.

Definition 3.3. Let $\mathcal{M}$ be a cut net, and F be an occurrence appearing in some $s \in \mathcal{M}$. We define $\mathcal{C}_F \subseteq \mathcal{M}$ as follows. If $F^\perp \in s'$ for some $s' \in \mathcal{M}$, then $\mathcal{C}_F$ is the connected component of $\mathcal{M} \setminus \{s\}$ containing s'. Otherwise, $\mathcal{C}_F = \emptyset$. If Δ is a set of occurrences, we define $\mathcal{C}_\Delta := \biguplus_{F\in\Delta} \mathcal{C}_F$.

Example 3.2. Let $\mathcal{M} = \{\vdash F, G, H, I, \vdash F^\perp, K, \vdash K^\perp, \vdash G^\perp, L, M \ \vdash L^\perp, \vdash H^\perp, O, P, \vdash O^\perp, \ \vdash P^\perp\}$ be a cut net. We have $\mathcal{C}_F = \{\vdash F^\perp, K, \ \vdash K^\perp\}$, $\mathcal{C}_G = \{\vdash G^\perp, L, M \ \vdash L^\perp\}$ and $\mathcal{C}_{\{F,G\}} = \mathcal{C}_F \cup \mathcal{C}_G$.

Proposition 3.1. *Let $s = \vdash F, \Delta, \Gamma$ be a sequent, and $\mathcal{M} = \{s\} \uplus \mathcal{C}$ be a cut net of conclusion $\vdash F, \Sigma$. One has $\mathcal{C} = \mathcal{C}_\Delta \uplus \mathcal{C}_\Gamma$. Moreover, $\{\vdash \Gamma\} \cup \mathcal{C}_\Gamma$ and $\{\vdash \Delta\} \cup \mathcal{C}_\Delta$ are cut nets and, if Σ_Γ and Σ_Δ are their respective conclusions, we have $\Sigma = \Sigma_\Delta \uplus \Sigma_\Gamma$.*

Proof. Since $\mathcal{M}$ is a tree, we have that $\mathcal{C} = \biguplus_{G\in s} \mathcal{C}_G$. We have also that $\mathcal{C}_F = \emptyset$ as F is in the conclusion of $\mathcal{M}$. Thus, $\mathcal{C} = \mathcal{C}_\Delta \uplus \mathcal{C}_\Gamma$. Moreover, every $\mathcal{C}_G$ where $G \in \Gamma$ is a tree, whose root is cut-connected to Γ, thus $\{\vdash \Gamma\} \cup \mathcal{C}_\Gamma$ is a cut net. The same holds for $\{\vdash \Delta\} \cup \mathcal{C}_\Delta$. It is easy to see that $\Sigma = \Sigma_\Delta \uplus \Sigma_\Gamma$. □

$$\dfrac{\mathcal{C} \quad \dfrac{\vdash \Delta, F \quad \vdash \Gamma, G}{\vdash \Delta, \Gamma, F \otimes G}\,(\otimes)}{\vdash \Sigma_\Delta, \Sigma_\Gamma, F \otimes G}\,(\mathsf{mcut}) \quad \underset{r}{\longrightarrow} \quad \dfrac{\dfrac{\mathcal{C}_\Delta \quad \vdash \Delta, F}{\vdash \Sigma_\Delta, F}\,(\mathsf{mcut}) \qquad \dfrac{\mathcal{C}_\Gamma \quad \vdash \Gamma, G}{\vdash \Sigma_\Gamma, G}\,(\mathsf{mcut})}{\vdash \Sigma_\Delta, \Sigma_\Gamma, F \otimes G}\,(\otimes)$$

$$\dfrac{\mathcal{C} \quad \dfrac{\vdash \Delta, F, G}{\vdash \Delta, F \parr G}\,(\parr)}{\vdash \Sigma, F \parr G}\,(\mathsf{mcut}) \quad \underset{r}{\longrightarrow} \quad \dfrac{\dfrac{\mathcal{C} \quad \vdash \Delta, F, G}{\vdash \Sigma, F, G}\,(\mathsf{mcut})}{\vdash \Sigma, F \parr G}\,(\parr)$$

$$\dfrac{\mathcal{C} \quad \dfrac{\vdash \Delta, F \quad \vdash \Delta, G}{\vdash \Delta, F \,\&\, G}\,(\&)}{\vdash \Sigma, F \,\&\, G}\,(\mathsf{mcut}) \quad \underset{r}{\longrightarrow} \quad \dfrac{\dfrac{\mathcal{C} \quad \vdash \Delta, F}{\vdash \Sigma, F}\,(\mathsf{mcut}) \qquad \dfrac{\mathcal{C} \quad \vdash \Delta, G}{\vdash \Sigma, G}\,(\mathsf{mcut})}{\vdash \Sigma, F \,\&\, G}\,(\&)$$

$$\dfrac{\mathcal{C} \quad \dfrac{\vdash \Delta, F_i}{\vdash \Delta, F_1 \oplus F_2}\,(\oplus_\mathsf{i})}{\vdash \Sigma, F_1 \oplus F_2}\,(\mathsf{mcut}) \quad \underset{r}{\longrightarrow} \quad \dfrac{\dfrac{\mathcal{C} \quad \vdash \Delta, F_i}{\vdash \Sigma, F_i}\,(\mathsf{mcut})}{\vdash \Sigma, F_1 \oplus F_2}\,(\oplus_\mathsf{i})$$

$$\dfrac{\mathcal{C} \quad \dfrac{}{\vdash \Delta, \top_\alpha}\,(\top)}{\vdash \Sigma, \top_\alpha}\,(\mathsf{mcut}) \quad \underset{r}{\longrightarrow} \quad \dfrac{}{\vdash \Sigma, \top_\alpha}\,(\top)$$

$$\dfrac{\mathcal{C} \quad \dfrac{\vdash \Delta}{\vdash \Delta, \bot_\alpha}\,(\bot)}{\vdash \Sigma, \bot_\alpha}\,(\mathsf{mcut}) \quad \underset{r}{\longrightarrow} \quad \dfrac{\mathcal{C} \quad \vdash \Delta}{\vdash \Sigma}\,(\mathsf{mcut})}{\vdash \Sigma, \bot_\alpha}\,(\bot) \qquad \dfrac{\dfrac{}{\vdash 1_\alpha}\,(1)}{\vdash 1_\alpha}\,(\mathsf{mcut}) \quad \underset{r}{\longrightarrow} \quad \dfrac{}{\vdash 1_\alpha}\,(1)$$

Figure 3.1: External reduction rules, where $r = (\mathsf{ext}, F)$ and F is the occurrence that is principal after the rule application.

We are now ready to introduce our reduction rules. Cut reduction rules are of two kinds, *principal* reductions and *auxiliary* ones. In the infinitary setting, principal cut reductions do not immediately contribute to producing a cut-free pre-proof. On the contrary, auxiliary cut reductions are productive in that sense. In other words, principal rules are seen as internal computations of the cut elimination process, while auxiliary rules are seen as a partial output of that process. Accordingly, the former will be called *internal rules* and the latter *external rules*.

Definition 3.4. ***External reductions*** are defined in Figure 3.1. The sets $\Sigma_\Delta, \Sigma_\Gamma, \mathcal{C}_\Delta$ and $\mathcal{C}_\Gamma$ used for the $(\otimes)/(\mathsf{mcut})$ external reduction rule are those defined in Proposition 3.1.

Remark 3.1. Notice that the $(\otimes)/(\mathsf{mcut})$ and $(\&)/(\mathsf{mcut})$ external reduction cases yield multiple multicuts, though always on disjoint sub-trees. Thus $\mu\mathsf{MALL}^\infty_\mathsf{m}$ is stable by external

reductions.

Remark 3.2. In external reductions, we pushed away the multicut rule above a logical rule. If we start from a multicut at the root, and perform only external rules, we will produce at the limit a cut-free proof. This is the reason why we say that external reductions are ***productive***. This is not the case for the internal reduction rules that we shall see next.

Definition 3.5. ***Merge reduction*** is defined as follows, with $r = (\mathsf{merge}, \{F, F^\perp\})$:

$$\dfrac{\mathcal{C} \qquad \dfrac{\vdash \Delta, F \qquad \vdash \Gamma, F^\perp}{\vdash \Delta, \Gamma}\,(\mathsf{Cut})}{\vdash \Sigma}\,(\mathsf{mcut}) \quad \underset{r}{\longrightarrow} \quad \dfrac{\mathcal{C} \qquad \vdash \Delta, F \qquad \vdash \Gamma, F^\perp}{\vdash \Sigma}\,(\mathsf{mcut})$$

Principal reductions are defined in Figure 3.2.
Internal reductions are the union of merge and principal reductions.

Remark 3.3. Notice that $\mu\mathsf{MALL}^\infty_\mathsf{m}$ is also stable by internal reductions.

Remark 3.4. In internal reductions, the multicut stays at the same level. Thus, if after some point of the cut-elimination procedure we perform only internal reduction rules, we will not produce a cut-free proof. Actually, we will show later that, at every moment of the reduction, some external redex will be available eventually. This result will ensure the productivity of cut-elimination, provided that we are sufficiently ***fair*** when reducing cuts.

3.1.3 Reduction sequences

We can now provide more explicit notions of reduction sequences and fairness.

Definition 3.6. A ***multicut reduction sequence*** is a finite or infinite sequence $\sigma = (\pi_i, r_i)_{i\in\lambda}$, with $\lambda \in \omega + 1$, where the π_i are $\mu\mathsf{MALL}^\infty_\mathsf{m}$ proofs, the r_i are labels identifying a multicut reduction rule and, whenever $i + 1 \in \lambda$, $\pi_i \underset{r_i}{\longrightarrow} \pi_{i+1}$.

We have seen that external rules may generate several multicuts. In a multicut reduction sequence, we are allowed to reduce all these multicuts, in any order. Later on, we will be interested in tracing the evolution of one single multicut, selecting only one sibling in the case of $(\otimes)/(\mathsf{mcut})$ and $(\&)/(\mathsf{mcut})$ external reductions. For that, we define the notion of ***reduction paths***:

Definition 3.7. A ***reduction path*** is a finite or infinite sequence $\sigma = (\theta_i, r_i)_{i\in\lambda}$, with $\lambda \in \omega + 1$, where the θ_i are $\mu\mathsf{MALL}^\infty_\mathsf{m}$ proofs ending with a multicut rule, r_i are labels identifying a multicut reduction rule and, whenever $i + 1 \in \lambda$, either r_i is an internal reduction label and then $\theta_i \underset{r_i}{\longrightarrow} \theta_{i+1}$, or r_i is an external reduction label, in this case, if π_i is the proof such that $\theta_i \underset{r_i}{\longrightarrow} \pi_i$ then θ_{i+1} should be a premise of π_i.

Remark 3.5. If a muticut reduction sequence is infinite, then by König's lemma, we can extract from it an infinite reduction path.

$$\dfrac{\mathcal{C} \quad \dfrac{\vdash \Delta, F \quad \vdash \Gamma, G}{\vdash \Delta, \Gamma, F \otimes G}\,(\otimes) \quad \dfrac{\vdash \Theta, G^\perp, F^\perp}{\vdash \Theta, G^\perp \parr F^\perp}\,(\parr)}{\vdash \Sigma}\,(\mathsf{mcut}) \quad \underset{r}{\longrightarrow} \quad \dfrac{\mathcal{C} \quad \vdash \Delta, F \quad \vdash \Gamma, G \quad \vdash \Theta, G^\perp, F^\perp}{\vdash \Sigma}\,(\mathsf{mcut})$$

$$\dfrac{\mathcal{C} \quad \dfrac{\vdash \Delta, F_2 \quad \vdash \Delta, F_1}{\vdash \Delta, F_2 \,\&\, F_1}\,(\&) \quad \dfrac{\vdash \Gamma, F_i^\perp}{\vdash \Gamma, F_1^\perp \oplus F_2^\perp}\,(\oplus_i)}{\vdash \Sigma}\,(\mathsf{mcut}) \quad \underset{r}{\longrightarrow} \quad \dfrac{\mathcal{C} \quad \vdash \Delta, F_i \quad \vdash \Gamma, F_i^\perp}{\vdash \Sigma}\,(\mathsf{mcut})$$

$$\dfrac{\mathcal{C} \quad \dfrac{\vdash \Delta, F[\mu X.F/X]}{\vdash \Delta, \mu X.F}\,(\mu) \quad \dfrac{\vdash \Gamma, F^\perp[\nu X.F^\perp/X]}{\vdash \Gamma, \nu X.F^\perp}\,(\nu)}{\vdash \Sigma}\,(\mathsf{mcut}) \quad \underset{r}{\longrightarrow} \quad \dfrac{\mathcal{C} \quad \vdash \Delta, F[\mu X.F/X] \quad \vdash \Gamma, F^\perp[\nu X.F^\perp/X]}{\vdash \Sigma}\,(\mathsf{mcut})$$

$$\dfrac{\mathcal{C} \quad \dfrac{\vdash \Gamma}{\vdash \Gamma, \perp}\,(\perp) \quad \dfrac{}{\vdash 1}\,(1)}{\Sigma}\,(\mathsf{mcut}) \quad \underset{r}{\longrightarrow} \quad \dfrac{\mathcal{C} \quad \vdash \Gamma}{\Sigma}\,(\mathsf{mcut})$$

Figure 3.2: Principal reductions, where $r = (\mathsf{principal}, \{F, F^\perp\})$ with $\{F, F^\perp\}$ the principal occurrences that have been reduced.

We will be interested in a particular kind of multicut reduction sequences, the ***fair*** ones, which are sequences such that any redex which is available at some point of the sequence will eventually have disappeared from the sequence (being reduced or erased). The following definition of fair reduction is standard from rewriting theory (see for instance chapter 9 of [Ter03]):

Definition 3.8. A ***multicut reduction sequence*** $(\pi_i, r_i)_{i \in \lambda}$ is ***fair*** if for every $i \in \lambda$ and r such that $\pi_i \underset{r}{\longrightarrow} \pi'$, there is some $j \geq i$, $j \in \lambda$, such that π_j cannot be reduced using r, in other words there is no proof θ such that $\pi_j \underset{r}{\longrightarrow} \theta$.

Fairness is defined in the same way for a reduction path rather than a reduction sequence. In that case, fairness can be rephrased in a simpler way: A ***multicut reduction path***

$(\pi_i, r_i)_{i\in\lambda}$ is ***fair*** if for every $i \in \lambda$ and r such that $\pi_i \underset{r}{\longrightarrow} \pi'$, there is some $j \geq i$, $j \in \lambda$, such that r has disappeared from π_{j+1}, that is, either r_j is r or r_j erases r.

Remark 3.6. Note that reduction paths extracted from a fair reduction sequence are always fair.

3.1.4 Statement of the cut-elimination theorem and proof sketch

The main result of this chapter is that fair reductions eliminate multicuts:

Theorem 3.1. *Fair multicut reduction sequences on $\mu\mathsf{MALL}^\infty_\mathsf{m}$ proofs produce $\mu\mathsf{MALL}^\infty$ proofs.*

Additionally, if all cuts in the initial derivation are above multicuts, the resulting $\mu\mathsf{MALL}^\infty$ derivation must actually be cut-free: indeed, multicut reductions never produce a cut. Thus Theorem 3.1 gives a way to eliminate cuts from any $\mu\mathsf{MALL}^\infty$ proof π of conclusion $\vdash \Gamma$: we start by forming a multicut with conclusion $\vdash \Gamma$ and π as unique sub-derivation (by applying the first multicut rule of Example 3.1), then we eliminate multicuts (and cuts) from that $\mu\mathsf{MALL}^\infty_\mathsf{m}$ proof.

The proof of Theorem 3.1 is in two parts. We first prove that fair *internal* multicut reductions cannot diverge (Proposition 3.5), hence fair multicut reductions are productive, *i.e.*, reductions of $\mu\mathsf{MALL}^\infty_\mathsf{m}$ proofs converge to $\mu\mathsf{MALL}^\infty$ pre-proofs. We then establish that the obtained pre-proof is a valid proof (Proposition 3.8).

Regarding productivity, assuming that there exists an infinite fair sequence σ of internal cut-reductions from a given proof π of conclusion Γ, we obtain a contradiction by extracting from π a proof of the empty sequent in a suitably defined proof-system. More specifically, we start by defining the subtree π_σ of π, which is the subtree visited by σ (Section 3.2). Since σ is fair, this means that no principal redex was available during the reduction, thus no occurrence from Γ is principal in π_σ. Hence, by erasing every occurrence in Γ from π_σ, local correctness of the proof is preserved, resulting in a tree deriving the empty sequent. This tree can be viewed as a proof in a new proof-system $\mu\mathsf{MALL}^\infty_\tau$ which is shown to be sound (Proposition 3.3) with respect to the traditional Boolean semantics of the μ-calculus, thus the contradiction.

The proof of validity of the produced pre-proof is done in a similar way.

3.2 Extracting proofs from reduction paths

We define now a key notion to analyze the behaviour of multicut-elimination: given a reduction path starting from π, we extract a (slightly modified) subderivation of π which corresponds to the part of the derivation that has been explored by the reduction. More precisely, given a reduction path σ starting with π, we consider the subtree of π whose sequents occur in the reduction path as premises of some multicut, and we call it the ***trace*** of σ. This subtree is obviously not always a $\mu\mathsf{MALL}^\infty$ derivation since some of its nodes may have missing premises, we call such sequent ***useless sequents*** We will provide an extension of $\mu\mathsf{MALL}^\infty$ where traces can be viewed as proper derivations, but first let us characterize the situations where useless sequents arise.

Definition 3.9. Let $\mathcal{R} = (\theta_i, r_i)_i$ be a fair reduction path starting with π. A sequent $s = (\vdash \Gamma, F)$ of π is said to be ***useless*** with ***distinguished occurrence*** F in one of the following cases:

1. The sequent s eventually occurs as a premise of all multicuts of $\mathcal{R}$ and F is the principal occurrence of s in π. Note that the distinguished occurrence F of a useless sequent s of sort (1) must be a sub-occurrence of a cut occurrence in π. Otherwise, the fair reduction path $\mathcal{R}$ would eventually have applied an external rule on s. Moreover, $F^\perp$ never becomes principal in the reduction path, otherwise by fairness the internal rule reducing F and $F^\perp$ would have been applied.

2. The sequent s is a premise of a rule (&) whose principal occurrence is of the form $F \,\&\, G$ or $G \,\&\, F$, and such that, at some point in the reduction, s is erased in an internal (&)/($\oplus$) multicut reduction. In other words, there is an index i such that either:

$$\theta_i \quad = \quad \dfrac{\mathcal{C} \qquad \dfrac{\vdash \Gamma, F \qquad \vdash \Gamma, G}{\vdash \Gamma, F \,\&\, G}\ (\&) \qquad \dfrac{\vdash \Delta, G^\perp}{\vdash \Delta, F^\perp \oplus G^\perp}\ (\oplus_i)}{\vdash \Sigma}\ (\mathsf{mcut})$$

or

$$\theta_i \quad = \quad \dfrac{\mathcal{C} \qquad \dfrac{\vdash \Gamma, G \qquad \vdash \Gamma, F}{\vdash \Gamma, G \,\&\, F}\ (\&) \qquad \dfrac{\vdash \Delta, G^\perp}{\vdash \Delta, G^\perp \oplus F^\perp}\ (\oplus_i)}{\vdash \Sigma}\ (\mathsf{mcut})$$

and

$$\theta_{i+1} \quad = \quad \dfrac{\mathcal{C} \qquad \vdash \Gamma, G \qquad \vdash \Delta, G^\perp}{\vdash \Sigma}\ (\mathsf{mcut})$$

3. The sequent s is a premise of a rule $\star \in \{(\otimes), (\&)\}$ whose principal occurrence is of the form $F \star G$ or $G \star F$, and such that s is ignored at some point in the reduction path because it is not present in the selected multicut after a branching external reduction $\star/(\mathsf{mcut})$. In other words, there exists an index i such that either :

$$\theta_i \quad = \quad \dfrac{\mathcal{C} \qquad \dfrac{\vdash \Gamma, F \qquad \vdash \Gamma, G}{\vdash \Gamma, F \,\&\, G}\ (\&)}{\vdash \Sigma, F \,\&\, G}\ (\mathsf{mcut}) \qquad \text{and} \qquad \theta_{i+1} \quad = \quad \dfrac{\mathcal{C} \qquad \vdash \Gamma, G}{\vdash \Sigma, G}\ (\mathsf{mcut})$$

or

$$\theta_i \quad = \quad \dfrac{\mathcal{C} \qquad \dfrac{\vdash \Gamma, F \qquad \vdash \Delta, G}{\vdash \Gamma, \Delta, F \otimes G}\ (\otimes)}{\vdash \Sigma_\Gamma, \Sigma_\Delta, F \otimes G}\ (\mathsf{mcut}) \qquad \text{and} \qquad \theta_{i+1} \quad = \quad \dfrac{\mathcal{C}_\Delta \qquad \vdash \Delta, G}{\vdash \Sigma_\Delta, G}\ (\mathsf{mcut})$$

4. The sequent s is ignored at some point in the reduction path because a ($\otimes$)/(mcut) external reduction distributes it to the multicut that is not selected in the path. In other words, there exists an index i such that:

$$\theta_i \quad = \quad \dfrac{\mathcal{C} \qquad \dfrac{\vdash \Theta, G \qquad \vdash \Delta, H}{\vdash \Theta, \Delta, G \otimes H}\ (\otimes)}{\vdash \Sigma_\Theta, \Sigma_\Delta, G \otimes H}\ (\mathsf{mcut}) \qquad \text{and} \qquad \theta_{i+1} \quad = \quad \dfrac{\mathcal{C}_\Theta \qquad \vdash \Theta, G}{\vdash \Sigma_\Theta, G}\ (\mathsf{mcut})$$

and such that $s \in \mathcal{C}_\Delta$. Note that all the sequents of $\mathcal{C}_\Delta$ are useless of sort (4). To define the distinguished occurrences of an element t of $\mathcal{C}_\Delta$, in particular that of s, we proceed by induction on the distance of t to the sequent $\vdash \Delta, H$ with respect to the cut-connexion relation, as follows:

- If a sequent t is cut-connected to $\vdash \Delta, H$ on the occurrence $I \in t$, we choose I to be the distinguished occurrence of t.
- If the distinguished occurrence of a sequent t has been defined, and if t' is cut-connected to t on $G \in t'$, we choose G as the distinguished occurrence of t'.

Note that, although the external reduction for $\top$ erases sequents, we do not need to consider such sequents as useless: indeed, we will only need to work with useless sequents in infinite reduction paths, and the external reduction associated to $\top$ terminates a path.

Remark 3.7. Note that two dual cut occurrences F and $F^\perp$ can never both be distinguished. Moreover, if F is distinguished, then every occurrence G extending F, *i.e.*, $F \sqsubseteq G$, is not distinguished.

Example 3.3. Let $\varphi = \nu X.X$ and F Let α, β and γ be three disjoint addresses and $\varphi = \nu X.X$. Consider the following $\mu\mathsf{MALL}^\infty_\mathsf{m}$ proof, where $F = \varphi_\alpha$, $F' = \varphi_{\alpha.i}$, $G = \varphi_\beta$, $G' = \varphi_{\beta.i}$ and $H = \psi_\gamma$. There is only one possible reduction sequence starting from this proof, which performs external reductions on G and its sub-occurrences.

$$\dfrac{\dfrac{\dfrac{(\dagger)}{\vdash F', H}\,(\nu)}{(\dagger) \vdash F, H}\,(\nu) \qquad \dfrac{\dfrac{(\star)}{\vdash F^\perp, G'}\,(\nu)}{(\star) \vdash F^\perp, G}\,(\nu)}{\vdash H, G}\,(\mathsf{mcut})$$

The sequent $\vdash F, H$ is useless of sort (1) for the reduction sequence starting from this proof. Its distinguished occurrence is F.

Definition 3.10. Let π be a $\mu\mathsf{MALL}^\infty_\mathsf{m}$ proof of $\vdash \Gamma$ and $\mathcal{R} = (\theta_i, r_i)_i$ be a reduction path starting with π. The ***trace*** of $\mathcal{R}$, denoted $\mathsf{TR}(\mathcal{R})$, is the subtree of π containing, for every i, the premises of θ_i.

Example 3.4. The trace of the reduction sequence starting from the proof of Example 3.3 is the following:

$$\dfrac{\vdash F, H \qquad \dfrac{\dfrac{(\star)}{\vdash F^\perp, G'}\,(\nu)}{(\star) \vdash F^\perp, G}\,(\nu)}{\vdash H, G}\,(\mathsf{mcut})$$

Notice that if s and t are two sequents of $\mathsf{TR}(\mathcal{R})$, appearing in the same branch of π, then all the sequents between s and t in π belong also to $\mathsf{TR}(\mathcal{R})$. Thus, $\mathsf{TR}(\mathcal{R})$ is indeed a tree, more precisely an open $\mu\mathsf{MALL}^\infty_\mathsf{m}$ derivation of conclusion $\vdash \Gamma$. Moreover, the infinite branches of $\mathsf{TR}(\mathcal{R})$ are also infinite branches of π, hence they satisfy the validity condition. Note also that the open sequents of $\mathsf{TR}(\mathcal{R})$ are exactly the useless sequents of $\mathcal{R}$. The

derivation $\mathsf{TR}(\mathcal{R})$ is then almost a $\mu\mathsf{MALL}^\infty_{\mathsf{m}}$ derivation, one needs only to close it on useless sequents. This is done by replacing distinguished occurrences by $\top$, through what we call ***truncations***.

Definition 3.11. A ***truncation*** τ is a partial function from Σ^* to $\{\top, 0\}$ such that:

- For any $\alpha \in \Sigma^*$, if $\alpha \in \mathtt{Dom}(\tau)$, then $\alpha^\perp \in \mathtt{Dom}(\tau)$ and $\tau(\alpha) = \tau(\alpha^\perp)^\perp$.
- If $\alpha \in \mathtt{Dom}(\tau)$ then for any $\beta \in \Sigma^+$, $\alpha.\beta \notin \mathtt{Dom}(\tau)$.

Definition 3.12. Let $\mathcal{R}$ be a reduction path. The ***truncation τ associated to $\mathcal{R}$*** is defined by setting $\tau(\alpha) = \top$ and $\tau(\alpha^\perp) = 0$ for every occurrence φ_α that is distinguished in some useless sequent of $\mathcal{R}$.

The above definition is justified thanks to Remark 3.7.

Example 3.5. The truncation associated to the reduction path of the proof of Example 3.3 is defined by $\tau(\alpha) = \top$ and $\tau(\alpha^\perp) = 0$, where α is the address of the useless occurrence F.

We now introduce a proof system where we can see the trace of a reduction path as a proper proof. This proof system is parametrized by a truncation τ, we denote it $\mu\mathsf{MALL}^\infty_\tau$. In $\mu\mathsf{MALL}^\infty_\tau$, we are allowed to replace an occurrence by its image by τ.

Definition 3.13. Given a truncation τ, the infinitary ***proof system*** $\mu\mathsf{MALL}^\infty_\tau$ is obtained by taking all the rules of $\mu\mathsf{MALL}^\infty$, with the proviso that they only apply when the address of their principal occurrence is not in the domain of τ, otherwise the following rule has to be applied:

$$\frac{\vdash \tau(\alpha)_{\alpha i}, \Delta}{\vdash \varphi_\alpha, \Delta}\ (\tau) \qquad \text{if } \alpha \in \mathtt{Dom}(\tau)$$

The notions of thread and validity is the same as in $\mu\mathsf{MALL}^\infty$.

The address $\alpha.i$ associated with $\tau(\alpha)$ in the rule (τ) forbids loops on a (τ) rule. Indeed if $\alpha \in \mathtt{Dom}(\tau)$ then $\alpha.i \notin \mathtt{Dom}(\tau)$.

Definition 3.14. Let $\mathcal{R}$ be a fair infinite reduction path starting with π and τ be the truncation associated to it. We define $\pi_\mathcal{R}$ to be the $\mu\mathsf{MALL}^\infty_\tau$ proof obtained from the trace $\mathsf{TR}(\mathcal{R})$, by closing every useless sequent $\vdash \varphi_\alpha, \Gamma$, with distinguished occurrence φ_α, using the following derivation:

$$\dfrac{\dfrac{}{\vdash \top_{\alpha i}, \Gamma}\ (\top)}{\vdash \varphi_\alpha, \Gamma}\ (\tau)$$

Example 3.6. The trace of the reduction sequence starting from the proof of Example 3.3 is the following:

$$\dfrac{\dfrac{\dfrac{}{\vdash \top_{\alpha.i}, H}\ (\top)}{\vdash F, H}\ (\tau) \qquad \dfrac{\dfrac{(\star)}{\vdash F^\perp, G'}\ (\nu)}{(\star) \vdash F^\perp, G}\ (\nu)}{\vdash H, G}\ (\mathsf{mcut})$$

Actually, we have to justify that Definition 3.14 is always correct. Indeed, one can imagine that the dual $\varphi^{\perp}_{\alpha^\perp}$ of a distinguished occurrence φ_α, is principal in $\mathsf{TR}(\mathcal{R})$. Since we have that $\alpha^\perp \in \mathtt{Dom}(\tau)$, and since in $\mu\mathsf{MALL}^\infty_\tau$ we apply rule (τ) as soon as possible, we will have to replace the rule applied to $\varphi^{\perp}_{\alpha^\perp}$ by the rule (τ) which will yield an occurrence of the formula 0. Thus, embeding $\mathsf{TR}(\mathcal{R})$ in $\mu\mathsf{MALL}^\infty_\tau$ would break its shape, while our goal is just to close the open useless sequents. Actually, we show that this situation never happens: if an occurrence is distinguished, then its dual is never principal in $\mathsf{TR}(\mathcal{R})$.

Proposition 3.2. *Let $\mathcal{R}$ be a fair reduction path. If F is distinguished then $F^\perp$ is not principal in $\mathsf{TR}(\mathcal{R})$.*

Proof. Notice that the dual $F^\perp$ of a distinguished occurrence F may only occur in $\mathcal{R}$ for distinguished occurrences of type (1) and (4). We show that in both cases, $F^\perp$ cannot be principal in $\mathsf{TR}(\mathcal{R})$. Let $\mathcal{R} = (\theta_i, r_i)_i$ be a fair reduction path and F be a distinguished occurrence appearing in a useless sequent s.

Suppose that F is distinguished of type (1), *i.e.*, there exists j such that for all $k \geq j$, the sequent s is the premise of θ_k, and F is the principal occurrence of s in π. Suppose by contradiction that $F^\perp$ is the principal occurrence of a sequent t in $\mathsf{TR}(\mathcal{R})$. Since t appears in $\mathsf{TR}(\mathcal{R})$, thus there exists $k \geq j$ such that t is the premise of θ_k. Hence in θ_k, F and $F^\perp$ form a redex, which by fairness should be eventually reduced. But if we reduce this redex, this means that s is not a useless sequent of type (1) anymore, contradiction.

Now suppose that F is distinguished of type (4). Thus, s is cut-connected to a useless sequent t (of type (3) or (4)) on F, hence $F^\perp \in t$. Since $F^\perp$ belongs to a useless sequent, it cannot be principal in $\mathsf{TR}(\mathcal{R})$. □

3.3 Truncated truth semantics

3.3.1 Truncated semantics

We fix a truncation τ and define a truth semantics with respect to which $\mu\mathsf{MALL}^\infty_\tau$ will be sound. The semantics is classical, assigning a boolean value to each occurrence. For convenience, we take $(\mathcal{B} := \{0, \top\}, \leq)$ as our Boolean lattice, with $\wedge$ and $\vee$ being the usual meet and join operations on it. The following interpretation assigns to every $\mu\mathsf{MALL}$ occurrence φ_α the boolean value $\tau(\alpha)$, if its address α is in the domain of τ. Otherwise, it is interpreted as a μ-calculus formula, by forgetting the linearity of the connectives *i.e.*, by interpreting the connectives $\parr$ and $\oplus$ as the classical disjunction $\vee$, and the connectives $\otimes$ and $\&$ as the classical conjunction $\wedge$.

Definition 3.15. We define the set E to be $\Sigma^* \to \mathcal{B}$, that is the set of total functions mapping addresses to booleans. Let $\leq$ be the pointwise order on E. The set E equipped with $\leq$ forms a complete lattice. We denote by $\bigvee E$ the supremum of E and by $\bigwedge$ its infimum.

Definition 3.16. Let φ_α be an occurrence. We call ***environment*** any function $\mathcal{E}$ mapping free variables of φ to elements of E. We define $[\![\varphi_\alpha]\!]^{\mathcal{E}} \in \mathcal{B}$, the ***interpretation*** of φ_α in the environment $\mathcal{E}$, by:

- If $\alpha \in \mathtt{Dom}(\tau)$ then $[\![\varphi_\alpha]\!]^{\mathcal{E}} = \tau(\alpha)$.

- Otherwise:
  - $[\![X_\alpha]\!]^{\mathcal{E}} = \mathcal{E}(X)(\alpha)$.
  - $[\![\top_\alpha]\!]^{\mathcal{E}} = [\![1_\alpha]\!]^{\mathcal{E}} = \top$.
  - $[\![0_\alpha]\!]^{\mathcal{E}} = [\![\bot_\alpha]\!]^{\mathcal{E}} = 0$.
  - $[\![(\varphi \mathbin{⅋} \psi)_\alpha]\!]^{\mathcal{E}} = [\![\varphi_{\alpha.l}]\!]^{\mathcal{E}} \wedge [\![\psi_{\alpha.r}]\!]^{\mathcal{E}}$, for $⅋ \in \{\&, \otimes\}$.
  - $[\![(\varphi \mathbin{⊗} \psi)_\alpha]\!]^{\mathcal{E}} = [\![\varphi_{\alpha.l}]\!]^{\mathcal{E}} \vee [\![\psi_{\alpha.r}]\!]^{\mathcal{E}}$, for $⊗ \in \{\oplus, ⅋\}$.
  - $[\![(\mu X.\varphi)_\alpha]\!]^{\mathcal{E}} = \mathsf{lfp}(f)(\alpha)$ and $[\![(\nu X.\varphi)_\alpha]\!]^{\mathcal{E}} = \mathsf{gfp}(f)(\alpha)$, where $f : E \to E$ is defined by:

$$f : h \mapsto \beta \mapsto \begin{cases} \tau(\beta) & \text{if } \beta \in \mathtt{Dom}(\tau) \\ [\![\varphi_{\beta.i}]\!]^{\mathcal{E}, X \mapsto h} & \text{otherwise.} \end{cases}$$

When F is closed, we simply write $[\![F]\!]$ for $[\![F]\!]^{\emptyset}$.

The least and greatest fixed point of f are guaranteed to exist in the above definition because $[\![\varphi]\!]^{\mathcal{E}}$ is a monotonic operator in the complete lattice E. The monotonicity of $[\![\varphi]\!]^{\mathcal{E}}$ is inherited from that of operators and can easily be proved by induction on φ.

We show now that $\mu\mathsf{MALL}^{\infty}_{\tau}$ is sound with respect to the truncated truth semantics, that is:

Proposition 3.3. *If* $\vdash \Gamma$ *is provable in* $\mu\mathsf{MALL}^{\infty}_{\tau}$*, then* $[\![F]\!] = \top$ *for some* $F \in \Gamma$.

We first give a sketch of the proof of soundness, which proceeds by contradiction. Assuming we are given a proof π of an occurrence F such that $[\![F]\!] = 0$, we exhibit a branch β of π containing only occurrences interpreted by 0. A validating thread of β unfolds infinitely often some formula $\nu X.\varphi$. Since the interpretation of $\nu X.\varphi$ is defined as the greatest fixed point of a monotonic operator f we have, for each occurrence $(\nu X.\varphi)_\alpha$ in β, an ordinal λ such that $[\![(\nu X.\varphi)_\alpha]\!] = f^{\downarrow\lambda}(\bigvee E)(\alpha)$, where $f^{\downarrow\lambda}(\bigvee E)$ is the descending iteration of f starting from $\bigvee E$ (Definition 2.6). We show that this ordinal can be forced to decrease along β at each fixed point unfolding, contradicting the well-foundedness of the class of ordinals.

3.3.2 Soundness *w.r.t.* truncated semantics

Marked formulas

In order to develop a formal proof of soundness, we need to work with a slightly enriched notion of formula. We thus introduce below a generalization of formulas, occurrences and of their interpretations (generalising Definition 3.16).

Definition 3.17. ***Marked formulas*** are built over the following syntax, where θ is an ordinal:

$$\varphi, \psi ::= 0 \mid \top \mid \varphi \oplus \psi \mid \varphi \,\&\, \psi \mid \bot \mid 1 \mid \varphi \mathbin{⅋} \psi \mid \varphi \otimes \psi \mid \mu X.\varphi \mid \nu^{\theta} X.\varphi \mid X \text{ with } X \in \mathcal{V}.$$

A marked occurrence is given by a marked formula φ and an address α and is written φ_α.

We define the interpretation of a marked occurrence as follows, generalizing Definition 3.16:

Definition 3.18. Let φ_α be a marked occurrence and $\mathcal{E}$ be an environment, *i.e.*, a function mapping every free variable of φ to an element of E. We define $[\varphi_\alpha]^{\mathcal{E}} \in \mathcal{B}$, the ***interpretation*** of φ_α in the environment $\mathcal{E}$ as follows: if $\alpha \in \mathtt{Dom}(\tau)$ then $[\varphi_\alpha]^{\mathcal{E}} = \tau(\alpha)$; otherwise:

- $[X_\alpha]^{\mathcal{E}} = \mathcal{E}(X)(\alpha)$.
- $[\top_\alpha]^{\mathcal{E}} = [1_\alpha]^{\mathcal{E}} = \top$.
- $[0_\alpha]^{\mathcal{E}} = [\bot_\alpha]^{\mathcal{E}} = 0$.
- $[(\varphi \mathbin{⅋} \psi)_\alpha]^{\mathcal{E}} = [\varphi_{\alpha.l}]^{\mathcal{E}} \wedge [\psi_{\alpha.r}]^{\mathcal{E}}$, for $⅋ \in \{\&, \otimes\}$.
- $[(\varphi \mathbin{⊕} \psi)_\alpha]^{\mathcal{E}} = [\varphi_{\alpha.l}]^{\mathcal{E}} \vee [\psi_{\alpha.r}]^{\mathcal{E}}$, for $⊕ \in \{\oplus, ⅋\}$.
- $[(\mu X.\varphi)_\alpha]^{\mathcal{E}} = \mathsf{lfp}(f)(\alpha)$ and $[(\nu^\theta X.\varphi)_\alpha]^{\mathcal{E}} = f^{\downarrow\theta}(\bigvee E)(\alpha)$,
  where $f : E \to E$ is defined by:

$$f : h \mapsto \beta \mapsto \begin{cases} \tau(\beta) & \text{if } \beta \in \mathtt{Dom}(\tau) \\ [\varphi_{\beta.i}]^{\mathcal{E}, X \mapsto h} & \text{otherwise.} \end{cases}$$

and where $f_\theta(\bigvee E)$ is the descending iteration of f starting from $\bigvee E$ (see Definition 2.6). We denote by $\mathcal{O}(\varphi, X, \mathcal{E})$ the operator f and we set $[\varphi]^{\mathcal{E}} := (\alpha \mapsto [\varphi_\alpha]^{\mathcal{E}})$.

As for Definition 3.16, the existence of the least fixed point of f relies on the monotonicity of $[\varphi]^{\mathcal{E}}$, which can be shown easily by induction on φ.

The interpretation $[\_]$ of a marked occurrence proceeds exactly as the interpretation $[\![\_]\!]$ of μMALL occurrences, but instead of defining the semantics of a ν-formula as the greatest fixed point of the associated operator, it iterates this operator up to θ, starting from the supremum $\bigvee E$ of E, where θ is the ordinal marking this ν-formula.

To relate the interpretations $[\![\_]\!]$ and $[\_]$, recall that Theorem 2.3 states that the greatest fixed point of a monotonic operator f over E is obtained by iterating down f starting from $\bigvee E$, λ_E many times where λ_E is the ordinal associated to E (Definition 2.5). Hence, if an occurrence is marked only with the ordinal λ_E, then its interpretation by $[\_]$ is exactly its interpretation by $[\![\_]\!]$ if we forget its marking.

Definition 3.19. Let λ_E be the ordinal associated to E (Definition 2.5) and let F be a μMALL occurrence. We define $\overline{\overline{F}}$ to be the marked occurrence, obtained from F by marking every ν binder of F by the ordinal λ_E.

By earlier remark, we have the following:

Proposition 3.4. *Let F be a μMALL occurrence. We have:*

$$[\![F]\!] = [\overline{\overline{F}}]$$

Technical lemmas

Let us now show some lemmas that will help us to prove soundness.

Lemma 3.1. *Let φ, ψ be marked formulas such that $X \notin \mathsf{fv}(\psi)$. One has:*

$$[\varphi_\alpha]^{\mathcal{E}, X \mapsto [\psi]^{\mathcal{E}}} = [(\varphi[\psi/X])_\alpha]^{\mathcal{E}}.$$

Proof. The proof is by induction on φ. We treat only the cases where φ is a fixed point formula, the other cases are immediate.

Suppose that $\varphi = \nu^\theta Y.\xi$ and let $f = \mathcal{O}(\xi, Y, \mathcal{E} :: X \mapsto [\psi]^{\mathcal{E}})$ and $g = \mathcal{O}(\xi[\psi/X], Y, \mathcal{E})$. By induction hypothesis one has $f^{\downarrow\theta}(\bigvee E) = g^{\downarrow\theta}(\bigvee E)$, which concludes this case.

Suppose now that $\varphi = \mu Y.\xi$, then we have:

$$\begin{array}{rcl} [(\mu Y.\xi)_\alpha]^{\mathcal{E}, X \mapsto [\psi]^{\mathcal{E}}} & = & \mathsf{lfp}(\mathcal{O}(\xi, Y, \mathcal{E} :: X \mapsto [\psi]^{\mathcal{E}}))(\alpha) \\ & \overset{*}{=} & \mathsf{lfp}(\mathcal{O}(\xi, Y, \mathcal{E} :: X \mapsto [\psi]^{\mathcal{E}, Y \mapsto h}))(\alpha) \\ & \overset{IH}{=} & \mathsf{lfp}(\mathcal{O}(\xi[\psi/X], Y, \mathcal{E}))(\alpha) \\ & = & [(\mu Y.\xi[\psi/X])_\alpha]^{\mathcal{E}} \end{array}$$

(*) We are considering capture-free substitutions, hence $Y \notin fv(\psi)$ and $[\psi]^{\mathcal{E}, Y \mapsto f} = [\psi]^{\mathcal{E}}$. □

An immediate consequence of this proposition is that the interpretation of a least fixed point occurrence is equal to the interpretation of its unfolding:

Lemma 3.2. *If $\alpha \notin \mathtt{Dom}(\tau)$, $[(\mu X.\varphi)_\alpha]^{\mathcal{E}} = [(\varphi[\mu X.\varphi/X])_{\alpha.i}]^{\mathcal{E}}$*

Proof. We set $f = \mathcal{O}(\varphi, X, \mathcal{E})$. Let us first notice that for all $\alpha \in \Sigma^*$, one has $[(\mu X.\varphi)_\alpha]^{\mathcal{E}} = \mathsf{lfp}(f)(\alpha)$. Indeed, one has the equality by definition when $\alpha \notin \mathtt{Dom}(\tau)$ and it is easy to prove it when $\alpha \in \mathtt{Dom}(\tau)$ since both sides are equal to $\tau(\alpha)$.

$$\begin{array}{rcl} [(\mu X.\varphi)_\alpha]^{\mathcal{E}} & = & \mathsf{lfp}(f)(\alpha) \\ & \overset{\star}{=} & [\varphi_{\alpha.i}]^{\mathcal{E}, X \mapsto \mathsf{lfp}(f)} \\ & = & [\varphi_{\alpha.i}]^{\mathcal{E}, X \mapsto [\mu X.\varphi]^{\mathcal{E}}} \\ & = & [(\varphi[\mu X.\varphi/X])_{\alpha.i}]^{\mathcal{E}} \end{array}$$

(*) Since $\alpha \notin \mathtt{Dom}(\tau)$. □

Lemma 3.3. *If $[(\nu^\theta X.\varphi)_\alpha]^{\mathcal{E}} = 0$ and $\alpha \notin \mathtt{Dom}(\tau)$ then there is an ordinal $\gamma < \theta$ such that $[(\varphi[\nu^\gamma X.\varphi/X])_{\alpha.i}]^{\mathcal{E}} = 0$.*

Proof. We set $f = \mathcal{O}(\varphi, X, \mathcal{E})$. If θ is a successor ordinal $\delta + 1$, then:

$$\begin{array}{rcl} [(\nu^\theta X.\varphi)_\alpha]^{\mathcal{E}} & = & f^{\downarrow\delta+1}(\bigvee E)(\alpha) \\ & = & [\varphi_{\alpha.i}]^{\mathcal{E}, X \mapsto f^{\downarrow\delta}(\bigvee E)} \\ & = & [\varphi_{\alpha.i}]^{\mathcal{E}, X \mapsto [\nu^\delta X.\varphi]^{\mathcal{E}}} \\ & = & [(\varphi[\nu^\delta X.\varphi/X])_{\alpha.i}]^{\mathcal{E}} \end{array}$$

We take γ to be the ordinal δ and we have obviously that $[(\varphi[\nu^\gamma X.\varphi/X])_{\alpha.i}]^{\mathcal{E}} = 0$.

If θ is a limit ordinal, then:

$$\begin{aligned}[(\nu^\theta X.\varphi)_\alpha]^{\mathcal{E}} &= f^{\downarrow\theta}(\bigvee E)(\alpha)\\ &= \bigcap_{\beta<\theta} f^{\downarrow\beta}(\bigvee E)\\ &= \bigcap_{\delta+1<\theta} f^{\downarrow\delta+1}(\bigvee E)\end{aligned}$$

Hence there is a successor ordinal $\delta+1$ such that $[(\nu^\theta X.\varphi)_\alpha]^{\mathcal{E}} = f^{\downarrow\delta+1}(\bigvee E)(\alpha)$ and we continue as before. □

We prove easily the following lemma by induction on F:

Lemma 3.4. *Let F be an (unmarked) occurrence. One has $[\![F^\perp]\!] = [\![F]\!]^\perp$.*

Proof of soundness

We can finally establish our soundness result:

Proof of Proposition 3.3. If F is a marked occurrence, we denote by F^* the occurrence obtained by forgetting the marking information.

Suppose that $\vdash \Gamma$ has a $\mu\mathsf{MALL}^\infty_\tau$ proof π and that $[\![F]\!] = 0$ for all $F \in \Gamma$. We will construct a branch $\gamma = (s_i)_{i\in\omega}$ of π and a sequence of functions $(f_i)_{i\in\omega}$ where f_i maps every occurrence G of s_i to a marked occurrence $f_i(G)$ such that $[f_i(G)] = 0$, and such that $(f_i(G))^* = G$ unless $G = \varphi_{\alpha.i}$ with $\alpha \in \mathtt{Dom}(\tau)$.

We set $s_0 = \Gamma$ and for every $F \in \Gamma$, $f_0(F) = \overline{\overline{F}}$. By proposition 3.4, we have $[\overline{\overline{F}}] = [\![F]\!] = 0$. Moreover, we have that $(\overline{\overline{F}})^\star = F$.

Suppose that for $i \in \omega$, we have constructed s_i and f_i. We construct now s_{i+1} depending on the rule applied to s_i in π:

- If the rule is a logical rule, G being principal in s_i, we set $G_m := f_i(G)$, we have the following cases:
  - If $G = H \parr K$, then G_m is of the form $G_m = H_m \parr K_m$. We set s_{i+1} to be the unique premise of s_i, $f_{i+1}(H) = H_m$ and $f_{i+1}(K) = K_m$. Since $[G_m] = 0$ and $[G_m] = [H_m] \vee [K_m]$, one has $[G_m] = 0$ and $[K_m] = 0$. For every other occurrence L of s_{i+1} we set $f_{i+1}(L) = f_i(L)$.
  - If $G = H \oplus K$, we proceed exactly in the same way as above.
  - If $G = H \otimes K$, then G_m is of the form $G_m = H_m \otimes K_m$. Since $[G_m] = 0$ and $[G_m] = [H_m] \wedge [K_m]$, one has $[H_m] = 0$ or $[K_m] = 0$. Suppose w.l.o.g. that $[H_m] = 0$. We set s_{i+1} to be the premise of s_i that contains H and $f_{i+1}(H) = H_m$. For every other occurrence L of s_{i+1} we set $f_{i+1}(L) = f_i(L)$.
  - If $G = H \,\&\, K$, we proceed exactly in the same way as above.
  - If $G = \mu X.K$, then G_m is of the form $G_m = \mu X.K_m$. We set s_{i+1} to be the premise of s_i and $f_{i+1}(K[G/X]) = K_m[G_m/X]$. By Corollary 3.2 and since $[G_m] = 0$, one has $[K_m[G_m/X]] = 0$. For every other occurrence L of s_{i+1}, we set $f_{i+1}(L) = f_i(L)$.

– If $G = \nu X.H$, then G_m is of the form $G_m = \nu^\theta X.K_m$. Let s_{i+1} be the unique premise of s_i. By corollary 3.3 and since $[G_m] = 0$, there is an ordinal $\delta < \theta$ such that $[K_m[\nu^\delta X.K_m/X]] = 0$. We set $f_{i+1}(H[G/X]) = K_m[\nu X^\delta.K_m/X]$ and for every other occurrence L of s_{i+1}, we set $f_{i+1}(L) = f_i(L)$.

- Suppose that the rule applied to s_i is a cut on the occurrence G. By Lemma 3.4, either $[\![G]\!] = 0$ or $[\![G^\perp]\!] = 0$, suppose w.l.o.g. that $[G] = 0$. We set s_{i+1} to be the premise of s_i containing G, $f_{i+1}(G) = \overline{\overline{G}}$ and for every other occurrence L of s_{i+1}, $f_{i+1}(L) = f_i(L)$.
- If the rule applied to s_i is the rule (τ) with a principal occurrence $G = \varphi_\alpha$, then $\alpha \in \mathtt{Dom}(\tau)$ and the address of $f_i(G)$ is also α. By definition of the interpretation $[\_]$, we have $[f_i(G)] = \tau(\alpha)$ and by construction $[f_i(G)] = 0$, thus $\tau(\alpha) = 0$. We set s_{i+1} to be the unique premise of s_i, $f_{i+1}(\tau(\alpha)_{\alpha.i}) = \tau(\alpha)_{\alpha.i}$ and for every other occurrence L of s_{i+1}, $f_{i+1}(L) = f_i(L)$.

Since π is a valid pre-proof, its branch γ must contain a valid thread $t = (F_i)_{i\in\omega}$. Let $\nu X.\varphi$ be the minimal formula of t and $(i_k)_{k\in\omega}$ be the sequence of indices where $\nu X.\varphi$ gets unfolded. By construction, for all $k > 0$ one has $f_{i_k}(F_{i_k}) = \nu^{\theta_k} X.G_k$ and the sequence of ordinals $(\theta_k)_k$ is strictly decreasing, which contradicts the well-foundedness of ordinals. □

3.4 Productivity of the cut-elimination process

We show that fair reduction sequences are productive.

Lemma 3.5. *Let π be a $\mu\mathsf{MALL}^\infty$ proof of $\vdash \Gamma$. If $\mathcal{R}$ is an infinite sequence of internal reductions starting from π, then the occurrences of Γ are never principal in $\pi_\mathcal{R}$.*

Proof. Suppose by contradiction that an occurrence F of Γ is principal in $\pi_\mathcal{R}$ and let s be the sequent of $\pi_\mathcal{R}$ where F is principal. Since $s \in \pi_\mathcal{R}$, then there is a proof θ of $\mathcal{R}$ such that s is a premise of θ. Since F is a conclusion occurrence of π, the proof θ has $r = (\mathsf{ext}, F)$ as redex. Since the premises of s also belong to $\pi_\mathcal{R}$, this means that r has been reduced in $\mathcal{R}$, which contradicts the fact that $\mathcal{R}$ is a sequence of internal reductions only. □

Proposition 3.5. *Any fair reduction sequence produces a $\mu\mathsf{MALL}^\infty$ pre-proof.*

Proof. By contradiction, consider a fair infinite sequence of internal reductions starting from π. By Remark 3.5, we can extract from it an infinite reduction path $\mathcal{R}$, which is also fair by Remark 3.6. Let τ and $\pi_\mathcal{R}$ be the truncation and the $\mu\mathsf{MALL}^\infty_\tau$ proof associated to $\mathcal{R}$. By Lemma 3.5, the conclusion occurrences Γ of π, which are also the conclusion occurrences of $\pi_\mathcal{R}$, are never principal in $\pi_\mathcal{R}$. We can then erase the occurrences of Γ from $\pi_\mathcal{R}$ without breaking its local and global validity. This yields a $\mu\mathsf{MALL}^\infty_\tau$ proof of the empty sequent, which contradicts soundness of $\mu\mathsf{MALL}^\infty_\tau$. □

3.5 Preservation of validity by cut-elimination

We show that the cut-free pre-proof resulting from a fair reduction sequence is actually a valid proof. We proceed by contradiction and suppose that the pre-proof obtained by

Proposition 3.5 is not valid. Thus, it must have an invalid branch, and this branch was produced by a reduction path $\mathcal{R}$. To get a contradiction, we consider as before, the truncation τ and the μMALL$^\infty_\tau$ proof $\pi_\mathcal{R}$ associated to $\mathcal{R}$, but instead of extracting from it a μMALL$^\infty_\tau$ proof of the empty sequent as we did for the proof of Proposition 3.5, we extract from it a proof of a sequent whose occurrences are all interpreted by 0, which, again, contradicts the soundness of μMALL$^\infty_\tau$.

Definition 3.20. A truncation τ is ***compatible*** with an occurrence φ_α if $\alpha \notin dom(\tau)$ and, for any $\alpha \sqsubseteq \beta.d \in \mathtt{Dom}(\tau)$ where $d \in \{l, r, i\}$, we have that φ_α admits a sub-occurrence ψ_β with $\otimes$ or & as the top-level connective of ψ, $d \in \{l, r\}$, and $\alpha.d' \notin \mathtt{Dom}(\tau)$ for any $d \neq d'$.

In other words, a truncation τ is compatible with an occurrence F if it truncates only sons of $\otimes$ or & nodes in the tree of F and at most one son of each such node.

Proposition 3.6. *If F is an occurrence compatible with τ and containing no ν binders, no $\top$ and no 1, then $[\![F]\!] = 0$.*

To show this proposition we need to generalize it as follows:

Proposition 3.7. *Let φ_α be an occurrence compatible with τ and containing no ν binders, no $\top$ and no 1 sub-formulas. Let $\mathcal{E}$ be an environment such that for all $\beta \notin \mathtt{Dom}(\tau)$, $\mathcal{E}(X)(\beta) = 0$. We have $[\![\varphi_\alpha]\!]^\mathcal{E} = 0$.*

Proof. The proof is by induction on φ.

- The cases when $\varphi = 0$ or $\bot$ are trivial.
- If $\varphi = X$, then $[\![X_\alpha]\!]^\mathcal{E} = \mathcal{E}(X)(\alpha) = 0$ by hypothesis on $\mathcal{E}$ and since $\alpha \notin \mathtt{Dom}(\tau)$ by compatibility with τ.
- If $\varphi = \xi \mathbin{⅋} \psi$, where $⅋ \in \{\oplus, ⅋\}$, then $[\![(\xi \mathbin{⅋} \psi)_\alpha]\!]^\mathcal{E} = [\![\xi_{\alpha.l}]\!]^\mathcal{E} \vee [\![\psi_{\alpha.r}]\!]^\mathcal{E}$. Since $(\xi \mathbin{⅋} \psi)_\alpha$ is compatible with τ, one has $\alpha.l \notin \mathtt{Dom}(\tau)$ and $\alpha.r \notin \mathtt{Dom}(\tau)$. Indeed, if a formula is compatible with a truncation τ, then τ cannot truncate a son of $\oplus$ or a ⅋ node. We can thus apply our induction hypothesis, obtaining $[\![\xi_{\alpha.l}]\!]^\mathcal{E} = [\![\psi_{\alpha.r}]\!]^\mathcal{E} = 0$, hence $[\![(\xi \mathbin{⅋} \psi)_\alpha]\!]^\mathcal{E} = 0$.
- If $\varphi = \xi \mathbin{⊗} \psi$, where $⊗ \in \{\&, ⊗\}$, then $[\![(\xi \mathbin{⊗} \psi)_\alpha]\!]^\mathcal{E} = [\![\xi_{\alpha.l}]\!]^\mathcal{E} \wedge [\![\psi_{\alpha.r}]\!]^\mathcal{E}$. Since $(\xi \mathbin{⊗} \psi)_\alpha$ is compatible with τ, one has $\alpha.l \notin \mathtt{Dom}(\tau)$ or $\alpha.r \notin \mathtt{Dom}(\tau)$. Indeed, if a formula is compatible with a truncation τ, then τ cannot truncate both sons of a & or a $\otimes$ node. We conclude by induction as before on the sub-formula that is not truncated, and which is thus still compatible with τ.
- If $\varphi = \mu X.\psi$, then $[\![\mu X.B]\!]^\mathcal{E} = \mathsf{lfp}(f)(\tau)$ where f is as in Definition 3.16. By Theorem 2.3, $[\![(\mu X.B)_\alpha]\!]^\mathcal{E} = \bigvee_{\delta<\lambda} \varphi^\delta(\bigwedge E)(\alpha)$. We show by an easy transfinite induction that for all $\delta < \lambda$ and $\beta \notin \mathtt{Dom}(\tau)$, we have $\varphi^\delta(\bigwedge E)(\beta) = 0$. This concludes the proof.

□

Proposition 3.8. *Any fair reduction produces a μMALL$^\infty$ proof.*

Proof. Let π be a $\mu\mathsf{MALL}^\infty_{\mathsf{m}}$ proof of conclusion $\vdash \Gamma$, and π' the cut-free pre-proof obtained by Proposition 3.5. Any branch of π' corresponds to a multicut reduction path. For the sake of contradiction, assume that π' is invalid. It must thus have an invalid infinite branch, corresponding to an infinite reduction path $\mathcal{R}$. Let τ and $\theta := \pi_{\mathcal{R}}$ be the truncation and the $\mu\mathsf{MALL}^\infty_\tau$ truncated proof associated to $\mathcal{R}$.

We first observe that formulas of Γ cannot have sub-occurrences of the form 1_α or $\top_\alpha$ that are principal in π'. Indeed, this could only be produced by an external rule $(\top)/(\mathsf{mcut})$ in the reduction path $\mathcal{R}$, but that would terminate the path, contradicting its infiniteness.

Next, we claim that all threads of θ starting from occurrences in Γ are invalid. Indeed, all rules applied to those formulas are transferred (by means of external rules) to the branch produced by the reduction path. The existence of a valid thread starting from the conclusion sequent in θ would thus imply the existence of a valid thread in our branch of π'.

By the first observation, we can replace all 1 and $\top$ subformulas of Γ by 0 without changing the derivation, and obviously without breaking its validity. By the second observation, we can further modify Γ by changing all ν combinators into μ combinators. We call Γ' the obtained sequent. The derivation θ is easily adapted (using rule (μ) instead of (ν)) and it remains valid, since the validity of θ is not due to a valid thread starting from the root. We thus obtain a valid pre-proof θ' of $\vdash \Gamma'$ in $\mu\mathsf{MALL}^\infty_\tau$, where Γ' contains no ν, 1 and $\top$.

We finally show that τ is compatible with any occurrence from Γ. Indeed, if $\tau(\beta)$ is defined for some sub-occurrence ψ_β of an occurrence $\varphi_\alpha \in \Gamma$, then it can only be because of a useless sequent of sort (3), *i.e.*, a truncation due to the fact that the reduction path has selected only one sibling after a branching external rule. We thus conclude, by Proposition 3.6, that all formulas of Γ are interpreted as 0 in the truncated semantics associated to τ, which contradicts the validity of θ' and Proposition 3.3. □

3.6 Examples

The type of natural numbers can be expressed by the formula $\mathsf{nat} := \mu X.1 \oplus X$. We give below a few examples of proofs/computations on natural numbers, shown using two sided sequents for clarity: any sequent of the form $F_1, \dots, F_n \vdash \Gamma$ should be read as $\vdash \Gamma, F_1^\perp, \dots, F_n^\perp$ as usual.

We define $\pi(n, \alpha)$, the representation of the natural number n in $\mu\mathsf{MALL}^\infty$, localized at the address α, by induction on n:

$$\pi(0,\alpha) \quad = \quad \dfrac{\dfrac{}{\vdash 1_{\alpha il}}\,{\scriptstyle (1)}}{\vdash \mathsf{nat}_\alpha}\,{\scriptstyle (\mu),(\oplus_1)}$$

$$\pi(n+1,\alpha) \quad = \quad \dfrac{\dfrac{\pi(n,\alpha ir)}{\vdash \mathsf{nat}_{\alpha ir}}}{\vdash \mathsf{nat}_\alpha}\,{\scriptstyle (\mu),(\oplus_2)}$$

The proof $\pi_{\mathsf{succ}}(\alpha,\beta)$, shown below, computes the successor on natural numbers:

$$\pi_{\mathsf{succ}}(\alpha,\beta) \quad = \quad \dfrac{\dfrac{\dfrac{}{\mathsf{nat}_\alpha \vdash \mathsf{nat}_{\beta ir}}\,(\mathsf{Ax})}{\mathsf{nat}_\alpha \vdash (1\oplus\mathsf{nat})_{\beta i}}\,(\oplus)}{\mathsf{nat}_\alpha \vdash \mathsf{nat}_\beta}\,(\mu)$$

If we cut $\pi_{\mathsf{succ}}(\alpha,\beta)$ against a $\pi(n,\alpha)$, we obtain after a finite number of cut-elimination steps the proof $\pi(n+1,\beta)$.

Consider now the following pre-proof, called $\pi_{\mathsf{dup}}(\alpha,\beta)$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\pi(0,\beta l)}{\vdash \mathsf{nat}_{\beta l}} \quad \dfrac{\pi(0,\beta r)}{\vdash \mathsf{nat}_{\beta r}}}{1_{\alpha il} \vdash (\mathsf{nat}\otimes\mathsf{nat})_\beta}\,(\bot),(\otimes) \quad \dfrac{(\star)}{\mathsf{nat}_{\alpha ir} \vdash (\mathsf{nat}\otimes\mathsf{nat})_\gamma} \quad \dfrac{\dfrac{\pi_{\mathsf{succ}}(\gamma l,\beta l)}{\mathsf{nat}_{\gamma l} \vdash \mathsf{nat}_{\beta l}} \quad \dfrac{\pi_{\mathsf{succ}}(\gamma r,\beta r)}{\mathsf{nat}_{\gamma r} \vdash \mathsf{nat}_{\beta r}}\,(\parr),(\otimes)}{(\mathsf{nat}\otimes\mathsf{nat})_\gamma \vdash (\mathsf{nat}\otimes\mathsf{nat})_\beta}\,(\mathsf{Cut})}{\mathsf{nat}_{\alpha ir} \vdash (\mathsf{nat}\otimes\mathsf{nat})_\beta}\,(\nu),(\&) \quad \mathsf{nat}_\alpha \vdash (\mathsf{nat}\otimes\mathsf{nat})_\beta}{(\star)}$$

The pre-proof $\pi_{\mathsf{dup}}(\alpha,\beta)$ is indeed a proof: it has exactly one infinite branch, validated by the thread starting with nat_α. If we cut that proof against the proof $\pi(n,\alpha)$, and perform cut-elimination steps, we obtain in finite time the following cut-free proof of $(\mathsf{nat}\otimes\mathsf{nat})_\beta$ which consists of two copies (up-to addresses) of the original proof $\pi(n,\alpha)$.

$$\dfrac{\dfrac{\pi(n,\beta l)}{\vdash \mathsf{nat}_{\beta l}} \quad \dfrac{\pi(n,\beta r)}{\vdash \mathsf{nat}_{\beta r}}}{\vdash (\mathsf{nat}\otimes\mathsf{nat})_\beta}\,(\otimes)$$

Now let $\mathsf{stream} = \nu X.\mathsf{nat}\otimes X$ be the formula representing infinite streams of natural numbers. Let us consider the derivation $\pi_{\mathsf{InfStr}}(\alpha,\beta)$ shown below:

$$\dfrac{\dfrac{\pi_{\mathsf{dup}}(\alpha,\gamma)}{\mathsf{nat}_\alpha \vdash (\mathsf{nat}\otimes\mathsf{nat})_\gamma} \quad \dfrac{\dfrac{\dfrac{}{\mathsf{nat}_{\gamma l} \vdash \mathsf{nat}_{\beta il}}\,(\mathsf{Ax}) \quad \dfrac{\dfrac{\pi_{\mathsf{succ}}(\gamma r,\sigma)}{\mathsf{nat}_{\gamma r} \vdash \mathsf{nat}_\sigma} \quad \dfrac{(\star)}{\mathsf{nat}_\sigma \vdash \mathsf{stream}_{\beta ir}}\,(\mathsf{Cut})}{\mathsf{nat}_{\gamma r} \vdash \mathsf{stream}_{\beta ir}}\,(\otimes)}{\mathsf{nat}_{\gamma l},\mathsf{nat}_{\gamma r} \vdash (\mathsf{nat}\otimes\mathsf{stream})_{\beta i}}\,(\parr)}{(\mathsf{nat}\otimes\mathsf{nat})_\gamma \vdash (\mathsf{nat}\otimes\mathsf{stream})_{\beta i}}\,(\mathsf{Cut})}{\mathsf{nat}_\alpha \vdash (\mathsf{nat}\otimes\mathsf{stream})_{\beta i}}\,(\nu)}{(\star) \quad \mathsf{nat}_\alpha \vdash \mathsf{stream}_\beta}$$

It is a valid proof thanks to the validity of $\pi_{\mathsf{dup}}(\alpha,\gamma)$ and that of the thread starting on stream_β. When we cut $\pi_{\mathsf{InfStr}}(\alpha,\beta)$ against the proof $\pi(n,\alpha)$, we obtain the proof that starts as follows:

$$\dfrac{\dfrac{\dfrac{\pi(n,\beta il)}{\vdash \mathsf{nat}_{\beta il}} \quad \dfrac{\dfrac{\dfrac{\pi(n+1,\beta iril)}{\vdash \mathsf{nat}_{\beta iril}} \quad \dfrac{\vdots}{\vdash \mathsf{stream}_{\beta irir}}}{\vdash (\mathsf{nat}\otimes\mathsf{stream})_{\beta iri}}\,(\otimes)}{\vdash \mathsf{stream}_{\beta ir}}\,(\nu)}{\vdash (\mathsf{nat}\otimes\mathsf{stream})_{\beta i}}\,(\otimes)}{\vdash \mathsf{stream}_\beta}\,(\nu)$$

This proof represents the stream $n :: (n+1) :: (n+2) :: \ldots$. This proof although obtained by the cut-elimination from a circular proof is not circular itself. As discussed earlier, this is not problematic, on the contrary it shows the strength of our proof system.

3.7 Extensions of the cut-elimination result

3.7.1 Treatment of atoms

We neglected atoms in our proof of cut-elimination, not because their treatment is fundamentally difficult, but because they would simply have burdened the presentation. Indeed, our axioms introduce a relocation, and we would be obliged to track these relocations along the reduction. To circumvent this problem, we can work additive slice by additive slice, relocate the axioms in such a way that they apply to dual occurrences and use the same argument as before.

3.7.2 Extension to $\mu\mathsf{LL}^\omega$, $\mu\mathsf{LK}^\omega$ and $\mu\mathsf{LK}\odot^\omega$

In this section we hint at how to obtain the extension of the infinitary cut-elimination result to other proof systems. We proceed essentially by encoding the proof system for which we desire to get the cut-elimination into the proof system that already enjoys it, then show that the cut-elimination steps can be simulated using this encoding.

To extend the cut-elimination result from $\mu\mathsf{MALL}$ to $\mu\mathsf{LL}$, we encode the exponential connectives using fixed points as follows:

$$\begin{array}{rcl} ?^\bullet\varphi & = & \mu X.\varphi \oplus ((X \parr X) \oplus \bot) \\ !^\bullet\varphi & = & \nu X.\varphi \,\&\, ((X \otimes X) \,\&\, 1) \end{array}$$

The exponential rules can be simulated in $\mu\mathsf{MALL}$ as follows:

$$\mathsf{Dereliction}: \quad \frac{\vdash F, \Delta}{\vdash ?^\bullet F, \Delta}\ (\mu), (\oplus_1)$$

$$\mathsf{Contraction}: \quad \frac{\vdash ?^\bullet F, ?^\bullet F \Delta}{\vdash ?^\bullet F, \Delta}\ (\mu), (\oplus_2), (\oplus_1), (\parr)$$

$$\mathsf{Weakening}: \quad \frac{\vdash \Delta}{\vdash ?^\bullet F, \Delta}\ (\mu), (\oplus_2), (\oplus_2), (\bot)$$

We denote these derivable rules by $?^\bullet$, $\mathsf{C}^\bullet$ and $\mathsf{W}^\bullet$ respectively. The promotion rule is derivable using the following circular proof:

$$\dfrac{\vdash F, ?^\bullet\Delta \qquad \dfrac{\dfrac{\dfrac{(\star)}{\vdash !^\bullet F, ?^\bullet\Delta} \qquad \dfrac{(\star)}{\vdash !^\bullet F, ?^\bullet\Delta}\ (\otimes)}{\vdash !^\bullet F \otimes !^\bullet F, ?^\bullet\Delta, ?^\bullet\Delta}\ (\mathsf{C}^\bullet)}{\vdash !^\bullet F \otimes !^\bullet F, ?^\bullet\Delta} \qquad \dfrac{\dfrac{}{\vdash 1}\ (1)}{\vdash 1, ?^\bullet\Delta}\ (\mathsf{W}^\bullet)}{(\star) \vdash !^\bullet F, ?^\bullet\Delta}\ (\nu), (\&)$$

We denote this derivable rule by $!^\bullet$. Now we have to show that the cut-elimination rules of $\mu\mathsf{LL}$ can be simulated by this encoding. For instance, the following reduction can be simulated by applying the extrenal reduction rule $(\mu)/(\mathsf{Cut})$ followed by the external reduction rule $(\oplus)/(\mathsf{Cut})$.

$$\dfrac{\dfrac{\vdash F, G, \Gamma}{\vdash ?^\bullet F, G, \Gamma}\,(?^\bullet) \qquad \vdash G^\perp, \Delta}{\vdash ?^\bullet F, \Gamma, \Delta}\,(\mathsf{Cut}) \longrightarrow \dfrac{\dfrac{\vdash F, G, \Gamma \qquad \vdash G^\perp, \Delta}{\vdash F, \Gamma, \Delta}\,(\mathsf{Cut})}{\vdash ?^\bullet F, \Gamma, \Delta}\,(?^\bullet)$$

The challenge is to show that this also holds for the reductions implying $(!^\bullet)$ since this rule hides an infinite derivation.

We hope that the extention from $\mu\mathsf{LL}$ to $\mu\mathsf{LK}$ can be made using the usual encoding of LK into LL. Finally, if the result of cut-elimination is obtained for $\mu\mathsf{LK}$, the extention to $\mu\mathsf{LK}\odot$ does not seem to pose a significant problem, since the reduction rules for $\odot$ do not seem to introduce more complexity than the other connectives.

Chapter 4

Focalization of μMALL$^\infty$ proofs

Focalization for MALL. Focalization is a result coming from Andreoli's work in the early 90s [And92], whose aim was to provide an algorithm for proof search in linear logic. While exploiting reversibility of inferences in proof search is an old idea, Andreoli noticed that duals of reversible connectives also had a good property.

The idea of focalization is to classify the connectives in two categories: the **negative** and the **positive** ones.

The negative connectives are those having *reversible* inferences, meaning that if the conclusion of the inference is provable, so are its premises. The negative connectives of MALL are $⅋, \&, \top, \bot$. In a proof search strategy, one can then apply safely negative inferences without losing provability.

The positive connectives are those having non-reversible inferences, that is, applying such inference may lead to a loss of provability. For instance, the formula $\top \oplus \bot$ is provable in MALL, but applying the rule $\oplus_2$ leads to $\bot$ which is not provable. In general, non-reversible inferences require to make some choices, for example splitting the context in $(\otimes)$ or choosing between $(\oplus_1)$ and $(\oplus_2)$ rules, and a bad choice may lead to a loss of provability.

Even if it does not give a concrete way to make the right choices when dealing with positive formulas, Andreoli's result was to show that we can reduce very much the non-determinism induced by the positive formulas, showing that they satisfy the ***focalization*** property [And92]: in any provable sequent containing only positive formulas, some formula can be chosen as a ***focus***, hereditarily selecting its positive subformulas as principal formulas until a negative subformula is reached. This induces the following complete proof search strategy, called the ***focalization discipline***:

- **Negative phase:** If the sequent contains a negative formula N, then decompose N.
- **Positive phase:** If the sequent contains only positive formulas, then select one and decompose it, and its subformulas, hereditarily until getting to atoms or negative subformulas.

Focalization is now recognized as one of the deep outcomes of linear logic, putting to the foreground the role of *polarity* in logic. Besides its deep impact on proof search and logic programming [AP91, Mil96], focalization resulted in important advances in all aspects of

computational proof theory: in the game-semantical analysis of logic [Lau04a, MT10], the understanding of evaluation order of programming languages [DJS97], CPS translations, or semantics of pattern matching [CM10, Zei09] and the space compression in computational complexity [BST11]. More fundamentally, focalization is at the heart of polarized linear logic [Lau02], and ludics [Gir01, Ter11] which we shall treat in Chapter 5.

What is a focalization result in general? There are many ways to answer this question. We choose to say that a proof system admits the focalization result if we can classify its connectives in two categories, positive and negative ones such that the focalization discipline described earlier is a complete proof-search algorithm. In MALL, the classification of connectives into positive and negative ones coincides with their classification into reversible and non-reversible inferences. We shall see that this identification is not relevant in the setting of infinitary proofs.

Proof of focalization. Focalization for MALL can be proved by means of proof transformations [Lau04b, MS07, BST11]: one starts from a linear proof and shows that it can be transformed into a ***focused*** proof, that is a proof respecting the focalization discipline described earlier. As these transformations preserve the denotation of the proof, we get as a byproduct of this proof technique, that focused proofs are complete for proofs, not only provability: any linear proof is equivalent to a focused proof, up to cut-elimination. We will be interested more specifically on a very elegant and modular proof by Miller and Saurin [MS07] which relies on *inference permutations* and *focalization graphs*. It is carried out in two steps:

I. **Reversibility of negatives.** This step is based on the fact that negative inferences permute down with any inference. Permuting down all the negative inferences, one can transform a linear proof into a proof having a layer of negative inference rules, the frontier of this layer being positive sequents, that is sequent containing only positive formulas.

II. **Focalization of positives.** This step relies on the fact that the positive inferences, while not reversible, all permute with each other. As a consequence, if the positive layer of some positive formula is completely decomposed within the lowest part of the proof, below any negative inference, then it can be taken as a focus. Focalization graphs ensure that it is always possible: their acyclicity provides a source which can be taken as a focus.

Focusing infinitary proofs. Our goal in this chapter is to adapt the proof technique of [MS07] to get a focalization result for μMALL$^\infty$. This means that we have to classify our connectives into negative and positive ones, and show that the proof search algorithm described earlier is complete for μMALL$^\infty$. Of course, this algorithm should be interpreted coinductively since we are dealing with infinitary proofs. The infinitary nature of our proofs interferes with focalization in several ways:

- For an infinitary proof system, being negative for a connective is not equivalent to its rule being reversible. For instance, the rule of the connective μ is reversible, but

we will see that if μ is classified as negative, the focalization discipline will not be complete. The classification of $\mu\mathsf{MALL}^\infty$ connectives is treated in Section 4.1.

- Step I of Miller and Saurin's proof is based on local inference permutations. We show that in $\mu\mathsf{MALL}^\infty$, these local inferences are not sufficient anymore. We give a new global inference permutations and adapt Step I to our setting. This is done in Section 4.2.
- Step II of Miller and Saurin's proof is based on focalization graphs, whose acyclicity relies on the finiteness of maximal positive sub-trees of a proof: this invariant must be preserved in $\mu\mathsf{MALL}^\infty$, as we shall prove in Section 4.3.
- Contrarily to MALL where showing Step I and II gives immediately the focalization result, showing them for $\mu\mathsf{MALL}^\infty$ is not enough. Indeed, combining them shows only that any $\mu\mathsf{MALL}^\infty$ proof can be transformed into a focused pre-proof. It remains to show that this pre-proof satisfies the global validity condition of $\mu\mathsf{MALL}^\infty$. That is what we will do in Section 4.4.

We restrict our attention to cut-free proofs. This is not a restriction in a proof search perspective, as long as the the cut rule is admissible, which is the case for $\mu\mathsf{MALL}^\infty$.

4.1 Polarity of connectives

Let us first consider the question of polarizing $\mu\mathsf{MALL}^\infty$ connectives. The polarities of MALL connectives are well-known [And92]: $\parr, \&, \top, \bot$ are negative and $\oplus, \otimes, 1, 0$ are positive.

Now we need to assign polarities to the connectives μ and ν. Assigning opposite polarities to dual connectives is an invariant necessary to define properly cut-elimination in focused proof systems, that is to get focused proofs that are stable by cut elimination. This constraint leaves us with only two possibilities to polarize μ and ν.

We show that considering μ as negative as ν as positive is a bad choice. Indeed, let Γ be the sequent $\vdash \mu X.X, \nu Y.Y$, which is provable in $\mu\mathsf{MALL}^\infty$ by unfolding $\nu Y.Y$ infinitely often. Now, if we consider μ as positive and ν as negative, and since the focalization discipline tells us to decompose negative formulas whenever we see them, the focalization algorithm will output the pre-proof which always unfolds μ, which is obviously not a (valid) proof. Thus, the focalization algorithm would not be complete with this classification of connectives. We are left with only one choice of polarities: μ positive and ν negative. We will show in the rest of this chapter that the focalization algorithm is complete for provability with this choice.

Definition 4.1. ***Negative formulas*** are formulas of the form $\nu X.F$, $F \parr G$, $F \,\&\, G$, $\bot$ and $\top$. ***Positive formulas*** are formulas of the form $\mu X.F$, $F \otimes G$, $F \oplus G$, 1 and 0.

A $\mu\mathsf{MALL}^\infty$ sequent containing only positive formulas is said to be ***positive***. Otherwise, it is said to be ***negative***.

The following proposition will be useful in the following:

Proposition 4.1. *An infinite branch of a pre-proof containing ultimately only negative (resp. positive) rules is valid (resp. not valid).*

$$\pi \quad = \quad \dfrac{\dfrac{\dfrac{(\star)}{\vdash F, P \parr Q} \qquad \dfrac{\dfrac{\theta}{\vdash F, P, Q}}{\vdash F, P \parr Q}\,(\parr)}{\vdash F \,\&\, F, P \parr Q}\,(\&)}{\dfrac{\vdash (F \,\&\, F) \oplus 0, P \parr Q}{(\star) \;\vdash F, P \parr Q}\,(\nu)}\,(\oplus_1)$$

Figure 4.1: Example of a proof where negative inferences cannot be permuted down locally

Proof. An infinite negative branch cannot unfold μ-formulas, since they are positive, but only ν-formulas. Among its threads, some are not eventually constant and their minimal formulas are ν-formulas: they are valid threads.

An infinite positive branch unfolds only μ-formulas, thus the minimal formula of any non-constant thread t is a μ-formula, thus it is not valid. □

4.2 Reversibility of negative inferences

Step I of Miller and Saurin's proof relies on the fact that negative rules permute down with any rule. These permutations are defined as local proof transformations. For instance, the following transformation shows how to permute down the $\parr$ rule below the & rule.

$$\dfrac{\dfrac{\vdash \Delta, F, P, Q}{\vdash \Delta, F, P \parr Q}\,(\parr) \qquad \dfrac{\vdash \Delta, F, P, Q}{\vdash \Delta, G, P \parr Q}\,(\parr)}{\vdash \Gamma, F \,\&\, G, P \parr Q}\,(\&) \quad \longrightarrow \quad \dfrac{\vdash \Delta, F, P, Q \qquad \vdash \Delta, F, P, Q}{\dfrac{\vdash \Delta, F \,\&\, G, P, Q}{\vdash \Gamma, F \,\&\, G, P \parr Q}\,(\parr)}\,(\&)$$

This rule requires the two $\parr$ formulas to be principal in the premises of the & rule. In a finitary proof, in particular in a MALL proof, if one wants to permute down a $\parr$ under a & it is always possible to permute down the $\parr$ inferences of the two occurrences of $P \parr Q$ in order to get the redex of the above transformation. But this is not always the case for infinite proofs. For instance, let $F = \nu X.(X \,\&\, X) \oplus 0$ and P, Q any formulas such that the sequent $\vdash F, P, Q$ has a proof θ. Consider the proof π of Example 4.1. In this proof, no occurrence of the rule $(\parr)$ on $P \parr Q$ can be permuted below a (&) since it is never available in the left premise. We introduce in the following a global proof transformation, which turns every μMALL$^\infty$ proof of conclusion Γ into a proof where all the negative occurrences of Γ, and their negative sub-occurrences have being decomposed, which will fix this issue.

Definition 4.2. All negative rules have the following uniform structure:

$$\dfrac{(\vdash \Gamma, \mathcal{N}_i^N)_{1 \leq i \leq n}}{\vdash \Gamma, N}\,(\mathsf{r_N})$$

We define the ***sub-occurrence family*** of N by $\mathcal{N}(N) := (\mathcal{N}_i^N)_{1 \leq i \leq n}$. We define the ***slicing index*** of N to be $\mathsf{sl}(N) = \#\mathcal{N}(N)$. The following table summarizes the sub-occurrence

families of μMALL occurrences:

N	$F_1 \parr F_2$	$\bot$	$F_1 \,\&\, F_2$	$\top$	$\nu X.F$
$\mathcal{N}(N)$	$\{1 \mapsto \{F_1, F_2\}\}$	$\{1 \mapsto \emptyset\}$	$\{1 \mapsto \{F_1\}, 2 \mapsto \{F_2\}\}$	$\emptyset$	$\{1 \mapsto \{F[\nu X.F/X]\}\}$

We can now define, in two steps, how to transform any proof π into a proof $\mathsf{rev}(\pi)$ where all negative inferences are reversed. The first step is to define $\pi(i, N)$ for every proof π. The effect of this transformation on π is to substitute everywhere the occurrence N by $\mathcal{N}_i^N$, while keeping a well-formed $\mu\mathsf{MALL}^\infty$ proof.

Definition 4.3. Let π be a proof of $\vdash \Gamma$ of last rule (r) and premises $\pi_1, \dots, \pi_n$. If $1 \leq i \leq \mathsf{sl}(N)$, we define $\pi(i, N)$ coinductively:

- If N does not occur in $\vdash \Gamma$, then $\pi(i, N) = \pi$.
- If r is the inference on N, then $\pi(i, N) = \pi_i$ (which is legal since in this case $n = \mathsf{sl}(N)$).
- If Γ is of the form $\Delta \cup \{N\}$ and r is not the inference on N, then we set:

$$\pi(i, N) = \dfrac{\pi_1(i, N) \quad \dots \quad \pi_n(i, N)}{\vdash \Delta, \mathcal{N}_i^N}\ (\mathsf{r})$$

Example 4.1. The proof $\pi(1, P \parr Q)$, where π is the proof in Example 4.1 is the following:

$$\pi(1, N) \quad = \quad \dfrac{\dfrac{\dfrac{\dfrac{(\star)}{\vdash F, P, Q} \qquad \dfrac{\theta}{\vdash F, P, Q}}{\vdash F \,\&\, F, P, Q}\ (\&)}{\vdash (F \,\&\, F) \oplus 0, P, Q}\ (\oplus_1)}{(\star) \ \vdash F, P, Q}\ (\nu)$$

Definition 4.4. Let π be a $\mu\mathsf{MALL}^\infty$ proof of $\vdash \Gamma$. Then $\mathsf{rev}(\pi)$ is a pre-proof non-deterministically defined as π if $\vdash \Gamma$ is positive and, otherwise, when $N \in \Gamma$ and $n = \mathsf{sl}(N)$, as:

$$\mathsf{rev}(\pi) = \dfrac{\mathsf{rev}(\pi(1, N)) \quad \dots \quad \mathsf{rev}(\pi(n, N))}{\vdash \Gamma}\ (\mathsf{r_N})$$

Reversed proofs formalize the requirement for the whole negative layer to be reversed:

Definition 4.5. ***Reversed pre-proofs*** are defined to be the largest set of pre-proofs such that: (i) every pre-proof of a positive sequent is reversed; (ii) a pre-proof of a negative sequent is reversed if it ends with a negative inference and if each of its premises is reversed.

Example 4.2. The transformation rev is illustrated below on the proof of Example 4.1, where we suppose that P and Q are positive.

$$\mathsf{rev}(\pi) \quad = \quad \dfrac{\pi(1, P \parr Q)}{\vdash F, P \parr Q}\ (\parr) \quad = \quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(\star)}{\vdash F, P, Q} \qquad \dfrac{\theta}{\vdash F, P, Q}}{\vdash F \,\&\, F, P, Q}\ (\&)}{\vdash (F \,\&\, F) \oplus 0, P, Q}\ (\oplus_1)}{(\star) \ \vdash F, P, Q}\ (\nu)}{\vdash F, P \parr Q}\ (\parr)$$

$$\dfrac{\dfrac{\vdash A_i, B, \Gamma \quad \vdash C, \Delta}{\vdash A_i, B \otimes C, \Gamma, \Delta}\,(\otimes)}{\vdash A_1 \oplus A_2, B \otimes C, \Gamma, \Delta}\,(\oplus_{\mathsf{i}}) \quad \longleftrightarrow \quad \dfrac{\dfrac{\vdash A_i, B, \Gamma}{\vdash A_1 \oplus A_2, B, \Gamma}\,(\oplus_{\mathsf{i}}) \quad \vdash C, \Delta}{\vdash A_1 \oplus A_2, B \otimes C, \Gamma, \Delta}\,(\otimes)$$

$$\dfrac{\dfrac{\vdash A_i, B_j, \Gamma}{\vdash A_i, B_1 \oplus B_2, \Gamma}\,(\oplus_{\mathsf{j}})}{\vdash A_1 \oplus A_2, B_1 \oplus B_2, \Gamma}\,(\oplus_{\mathsf{i}}) \quad \longleftrightarrow \quad \dfrac{\dfrac{\vdash A_i, B_j, \Gamma}{\vdash A_1 \oplus A_2, B_j, \Gamma}\,(\oplus_{\mathsf{i}})}{\vdash A_1 \oplus A_2, B_1 \oplus B_2, \Gamma, \Delta}\,(\oplus_{\mathsf{j}})$$

$$\dfrac{\dfrac{\vdash A, C, \Gamma \quad \vdash D, \Delta}{\vdash A, C \otimes D, \Gamma, \Delta}\,(\otimes) \quad \vdash B, \Sigma}{\vdash A \otimes B, C \otimes D, \Gamma, \Delta, \Sigma}\,(\otimes) \quad \longleftrightarrow \quad \dfrac{\dfrac{\vdash A, C, \Gamma \quad \vdash B, \Sigma}{\vdash A \otimes B, C, \Gamma, \Sigma}\,(\otimes) \quad \vdash D, \Delta}{\vdash A \otimes B, C \otimes D, \Gamma, \Delta, \Sigma}\,(\otimes)$$

$$\dfrac{\dfrac{\vdash A_i, C[\mu X.C/X], \Gamma}{\vdash A_i, \mu X.C, \Gamma}\,(\mu)}{\vdash A_1 \oplus A_2, \mu X.C, \Gamma}\,(\oplus_{\mathsf{i}}) \quad \longleftrightarrow \quad \dfrac{\dfrac{\vdash A_i, C[\mu X.C/X], \Gamma}{\vdash A_1 \oplus A_2, C[\mu X.C/X], \Gamma}\,(\oplus_{\mathsf{i}})}{\vdash A_1 \oplus A_2, \mu X.C, \Gamma}\,(\mu)$$

$$\dfrac{\dfrac{\vdash A, C[\mu X.C/X], \Gamma \quad \vdash B, \Delta}{\vdash A \otimes B, C[\mu X.C/X], \Gamma, \Delta}\,(\otimes)}{\vdash A \otimes B, \mu X.C, \Gamma, \Delta}\,(\mu) \quad \longleftrightarrow \quad \dfrac{\dfrac{\vdash A, C[\mu X.C/X], \Gamma}{\vdash A, \mu X.C, \Gamma}\,(\mu) \quad \vdash B, \Delta}{\vdash A \otimes B, \mu X.C, \Gamma, \Delta}\,(\otimes)$$

Figure 4.2: Permutations of positive μMALL$^\infty$ inferences

Theorem 4.1. *If π be a μMALL$^\infty$ proof then $\mathsf{rev}(\pi)$ is a reversed proof of the same sequent.*

Proof. rev is obviously productive: each recursive call is guarded. Inferences of $\mathsf{rev}(\pi)$ are locally valid: if π is a pre-proof, so is $\mathsf{rev}(\pi)$.

If moreover π is a proof, infinite branches of $\mathsf{rev}(\pi)$ are valid: indeed, infinite branches of $\mathsf{rev}(\pi)$ are either fully negative (and therefore valid) or after a certain point they coincide with inferences of an infinite branch of π and their validity follows from that of π.

The resulting proof is obviously shown to be reversed: we do not find any positive inference on any branch of $\mathsf{rev}(\pi)$, until the first positive sequent is reached. □

4.3 Focalization of positives

In this section, we show that any proof of a positive sequent Γ, can be transformed into a proof of the same sequent which chooses one of the occurrences of Γ and starts by decomposing it hereditarily until reaching its negative sub-occurrences. This transformation is based on permutations of positive rules, shown in Figure 4.2. These rules are the one presented in [MS07], together with the rules permuting μ with the other positive inferences. Now, we

$$\pi \quad = \quad \dfrac{\dfrac{\dfrac{(\star)}{\vdash I', L'}}{(\star) \vdash I, L'}\,(\nu) \qquad \dfrac{\dfrac{\dfrac{(\dagger)}{\vdash G', J}}{(\dagger) \vdash G, J}\,(\nu) \qquad \dfrac{\dfrac{(\ddagger)}{\vdash H', K}}{(\ddagger) \vdash H, K}\,(\nu)}{\vdash G \otimes H, J, K}\,(\otimes)}{\vdash G \otimes H, J \parr K}\,(\parr)}{\vdash G \otimes H, I \otimes (J \parr K), L'}\,(\otimes)$$
$$\dfrac{\dfrac{\vdash F \oplus (G \otimes H), I \otimes (J \parr K), L'}{\vdash F \oplus (G \otimes H), I \otimes (J \parr K), L}\,(\mu) \qquad \dfrac{}{\vdash M}\,(1)}{\vdash F \oplus (G \otimes H), I \otimes (J \parr K), L \otimes M}\,(\otimes)$$

$$\pi^+ \quad = \quad \dfrac{\dfrac{\dfrac{\vdash I, L' \qquad \vdash G \otimes H, J \parr K}{\vdash G \otimes H, I \otimes (J \parr K), L'}\,(\otimes)}{\vdash F \oplus (G \otimes H), I \otimes (J \parr K), L'}\,(\oplus) \qquad \dfrac{}{\vdash M}\,(1)}{\vdash F \oplus (G \otimes H), I \otimes (J \parr K), L \otimes M}\,(\otimes)$$

Figure 4.3: Example of a $\mu\mathsf{MALL}^\infty$ proof of a positive sequent and its positive trunk.

will see how and to which inferences we should apply these permutations to get our result. For that purpose, we generalize will use *focalization graphs* [MS07] to our setting.

Definition 4.6. Let π be a $\mu\mathsf{MALL}^\infty$ proof of $\mathcal{S}$. The ***positive trunk*** π^+ of π is the tree obtained by cutting (finite or infinite) branches of π at the first occurrence of a negative rule. The ***positive border*** of π is the collection of lowest sequents in π which are conclusions of negative rules. ***P-active*** occurrences of π are those occurrences of $\mathcal{S}$ which are principal formulas of an inference in π^+.

Example 4.3. Let α, β and γ be three disjoint addresses. In Figure 4.3, we show a proof π of the positive sequent $\vdash F \oplus (G \otimes H), I \otimes (J \parr K), L \otimes M$ where:

$$\begin{array}{llll} F = 0_{\alpha.l} & I = (\nu X.X)_{\beta.l} & L = (\mu X.X)_{\gamma.l} & I' = (\nu X.X)_{\beta.l.i} \\ G = (\nu X.X)_{\alpha.r.l} & J = 1_{\beta.r.l} & M = 1_{\gamma.r} & G' = (\nu X.X)_{\alpha.r.l.i} \\ H = (\nu X.X)_{\alpha.r.r} & K = 0_{\beta.r.r} & L' = (\mu X.X)_{\gamma.l.i} & H' = (\nu X.X)_{\alpha.r.r.i} \end{array}$$

and its positive trunk π^+. The positive border of π is $\{\vdash I, L' \ ; \vdash G \otimes H, J \parr K\}$ and its P-active occurrences are $F \oplus (G \otimes H)$, $I \otimes (J \parr K)$ and $L \otimes M$.

Proposition 4.2. *The positive trunk of a $\mu\mathsf{MALL}^\infty$ proof is always finite.*

Proof. The positive trunk of a proof cannot have infinite branches, because they would be infinite positive branches of the original proof, thus necessarily invalid by proposition 4.1. □

Definition 4.7. Given a $\mu\mathsf{MALL}^\infty$ proof π, we define its ***focalization graph*** $\mathcal{G}(\pi)$ to be the graph whose vertices are the P-active formulas of π and such that there is an edge from

F to G iff there is a sequent $\mathcal{S}'$ in the positive border containing a negative sub-occurrence F' of F and a positive sub-occurrence G' of G.

Example 4.4. The focalization graph of the proof of Figure 4.3 is the following:

$$L \otimes M \longleftarrow I \otimes (J \parr K) \longrightarrow F \oplus (G \otimes H)$$

Proposition 4.3. *Focalization graphs are acyclic.*

Proof. We prove the result by *reductio ad absurdum*. Let $\mathcal{S}$ be a positive sequent with a proof π. Let π^+ be the corresponding positive trunk and $\mathcal{G}$ the associated focalization graph. Suppose that $\mathcal{G}$ has a cycle and consider such a cycle of minimal length $(F_1 \to F_2 \to \cdots \to F_n \to F_1)$ in $\mathcal{G}$ and let us consider $\mathcal{S}_1, \ldots, \mathcal{S}_n$ sequents of the border justifying the arrows of the cycle.

These sequents are actually uniquely defined for the exact same reason as in MALL [MS07]. With the same idea we can immediately notice that the cycle is necessarily of length $n \geq 2$ since two $\prec$-subformulas of the same formula can never be in the same sequent in the border of the positive trunk.

Let $\mathcal{S}_0 = \bigwedge_{i=1}^n \mathcal{S}_i$ be the highest sequent in π such that all the $\mathcal{S}_i$ are leaves of the tree rooted in $\mathcal{S}_0$. We will obtain the contradiction by studying $\mathcal{S}_0$ and we will reason by case on the rule applied to this sequent $\mathcal{S}_0$:

- The rule cannot be the rule (1) since this rule produces no premise and thus we would have an empty cycle which is non-sense. Any rule with no premise would lead to the same contradiction.

- If the rule is one of $(\oplus_{\mathsf{i}})$ or (μ), then the premise $\mathcal{S}'_0$ of the rule would also satisfy the condition required for $\mathcal{S}_0$ (all the $\mathcal{S}_i$ would be part of the proof tree rooted in $\mathcal{S}'_0$) contradicting the maximality of $\mathcal{S}_0$. If the rule is any other non-branching rule, maximality of $\mathcal{S}_0$ would also be contradicted.

- Thus the rule shall be branching: it shall be a $(\otimes)$. Write $\mathcal{S}_L$ and $\mathcal{S}_R$ for the left and right premises of $\mathcal{S}_0$. Let $G = G_L \otimes G_R$ be the principal formula in $\mathcal{S}_0$ and let F be the active formula of the trunk such that $F \prec G$.

  There are two possibilities:

  (i) either $F \in \{F_1, \ldots, F_n\}$ and F is the only formula of the cycle having at the same time $\prec$-subformulas in the left premise and in the right premise,

  (ii) or $F \notin \{F_1, \ldots, F_n\}$ and no formula of the cycle has $\prec$-subformulas in both premises.

  Let thus I_L (resp. I_R) be the sets of indices of the active formulas of the root $\mathcal{S}$ having ($\prec$-related) subfomulas only in the left (resp. right) premise. Clearly neither I_L nor I_R is empty since it would contradict the maximality of $\mathcal{S}_0$. Indeed if $I_L = \emptyset$, then $\mathcal{S}_R$ satisfies the condition of being dominated by all the $\mathcal{S}_i, 1 \leq i \leq n$ and $\mathcal{S}_0$ is not maximal anymore. By definition of the two sets of indices we have of course

$I_L \cap I_R = \emptyset$ and the only formula of the cycle possibly not in $I_L \cup I_R$ is F if we are in the case (i): all other formulas in the cycle have their index either in I_L or in I_R.

As a consequence there must be an arrow in the cycle (and thus in the graph) from a formula in I_L to a formula in I_R (or the opposite). Let $i \in I_L$ and $j \in I_R$ be such indices (say for instance $F_i \to F_j$ in $\mathcal{G}$) and let $\mathcal{S}'$ be the sequent of the border responsible for this edge. $\mathcal{S}'$ contains F'_i and F'_j and by definition of the sets I_L and I_R, $\mathcal{S}'$ cannot be in the tree rooted in $\mathcal{S}_0$ which is in contradiction with the way we constructed $\mathcal{S}_0$.

Then there cannot be any cycle in the focalization graph. □

Acyclicity of the focalization graph implies in particular that it has a source, that is a formula P of the conclusion sequent such that whenever one of its subformulas F appears in a border sequent, F is negative. This remark, together with the fact that the trunk is finite ensures that the positive layer of P is completely decomposed in the positive trunk.

Definition 4.8. Let π be a $\mu\mathsf{MALL}^\infty$ proof of $\vdash \Gamma, P$ with P a source of π's focalization graph. One defines $\mathsf{foc}(\pi, P)$ as the $\mu\mathsf{MALL}^\infty$ proof obtained by permuting down all the positive inferences on P and its positive sub-occurrences (all occurring in π^+).

Example 4.5. Let us consider again the proof π of Figure 4.3. Its focalization graph (Example 4.4) is indeed acyclic and has a single source $I \otimes (J \parr K)$, which we pick as focus. By permuting down the positive inferences on $I \otimes (J \parr K)$, we arrive at:

$$\dfrac{\dfrac{\pi_1}{\vdash I, L \otimes M} \qquad \dfrac{\pi_2}{\vdash F \oplus (G \otimes H), J \parr K}}{\vdash F \oplus (G \otimes H), I \otimes (J \parr K), L \otimes M}\ (\otimes)$$

with:

$$\pi_1 = \dfrac{\dfrac{\dfrac{(\star)}{\vdash I', L'}\ (\nu)}{(\star) \vdash I, L'}\ (\mu) \qquad \dfrac{}{\vdash M}\ (1)}{\vdash I, L \otimes M}\ (\otimes) \qquad \pi_2 = \dfrac{\dfrac{\dfrac{\dfrac{(\dagger)}{\vdash G', J}}{(\dagger) \vdash G, J}\ (\nu) \qquad \dfrac{\dfrac{(\ddagger)}{\vdash H', K}}{(\ddagger) \vdash H, K}\ (\nu)}{\vdash G \otimes H, J, K}\ (\otimes)}{\vdash G \otimes H, J \parr K}\ (\parr)}{\vdash F \oplus (G \otimes H), J \parr K}\ (\oplus)$$

Proposition 4.4. *Let $\mathcal{S}$ be a lowest sequent of $\mathsf{foc}(\pi, P)$ which is not the conclusion of a rule on a positive subformula of P. Then $\mathcal{S}$ contains exactly one subformula of P, which is negative.*

Proof. The proof $\mathsf{foc}(\pi, P)$ is such that the maximal prefix containing only rules applied to P and its positive subformulas decomposes P up to its negative subformulas. Uniqueness of the subformula in the case of MALL, treated in [MS07], can be directly adapted here. □

4.4 Productivity and validity of the focalization process

Reversibility of the negative inferences and focalization of the positive inferences allow to consider the following (non-deterministic) proof transformation process:

Focalization Process: Let π be a μMALL$^\infty$ proof of $\mathcal{S}$. Define $\mathsf{Foc}(\pi)$ as follows:

- **Negative phase:** If $\mathcal{S}$ is negative, transform π into $\mathsf{rev}(\pi)$.At least one negative inference has been brought to the root of the proof. Apply (corecursively) the positive phase to the proofs rooted in the lowest positive sequents of $\mathsf{rev}(\pi)$.
- **Positive phase:** If $\mathcal{S}$ is positive, let $P \in \mathcal{S}$ be a source of the associated focalization graph. Transform π into the proof $\mathsf{foc}(\pi, P)$. At least one positive inference on P has been brought to the root of the proof. Apply (corecursively) the negative phase to the proofs rooted in the lowest negative sequents of $\mathsf{foc}(\pi, P)$.

Each of the above phases produces a non-empty derivation, the above process is thus productive:

Proposition 4.5. *If π be a μMALL$^\infty$ proof, then $\mathsf{Foc}(\pi)$ is a pre-proof.*

It remains to show that the resulting pre-proof is actually a proof. The following property is easily seen to be preserved by both transformations foc and rev and thus holds for $\mathsf{Foc}(\pi)$:

Proposition 4.6. *Let π be a μMALL$^\infty$ proof, let r be a positive rule occurring in π and r' be a negative rule occurring below r in π. If r occurs in $\mathsf{Foc}(\pi)$, then r' occurs in $\mathsf{Foc}(\pi)$, below r.*

Proof. The proposition amounts to the simple remark that none of the transformation we perform to obtain $\mathsf{Foc}(\pi)$ will ever permute a positive **below** a negative one. Indeed, in the negative phase we permute negative rules below positive ones, and in the positive phase we permute positive rules with each other. □

Lemma 4.1. *For any infinite branch γ of $\mathsf{Foc}(\pi)$ containing an infinite number of positive rules, there exists an infinite branch in π containing infinitely many positive rules of γ.*

Proof. The lemma results from a simple application of König's lemma. □

Theorem 4.2. *If π is a μMALL$^\infty$ proof then $\mathsf{Foc}(\pi)$ is also a μMALL$^\infty$ proof.*

Proof. Let γ be an infinite branch of $\mathsf{Foc}(\pi)$. If, at a certain point, γ is obtained by reversibility only, then it contains only negative rules and is therefore valid.

Otherwise, γ has been obtained by alternating infinitely often focalization phases foc and reversibility phases rev as described above. It therefore contains infinitely many positive inferences. By Lemma 4.1, there exists an infinite branch δ of π containing an infinite number of positive rules of γ. Since δ is valid, it contains a valid thread t.

Let F_m be the minimal formula of thread t, a ν-formula, and $(\mathsf{r}_i)_{i\in\omega}$ the rules of δ in which F_m is the principal formula.

For any i, there exists a positive rule r'_i occurring in γ which is above r_i and r_i therefore also appears in γ by Proposition 4.6, which is therefore valid. □

4.5 Example

We conclude this chapter by presenting the whole focalization process on the proof π of Example 4.3.

Example 4.6. In Example 4.5, we performed the first positive phase on the proof π. Now, we enter a negative phase for the proofs π_1 and π_2. For that, we compute $\mathsf{rev}(\pi_1)$ and $\mathsf{rev}(\pi_2)$:

$$\mathsf{rev}(\pi_1) = \dfrac{\dfrac{(\star)}{(\star)\vdash I', L\otimes M}}{\vdash I, L\otimes M}\,(\nu) \qquad \mathsf{rev}(\pi_2) = \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(\dagger)}{\vdash G', J}}{(\dagger)\vdash G, J}\,(\nu) \quad \dfrac{\dfrac{(\ddagger)}{\vdash H', K}}{(\ddagger)\vdash H, K}\,(\nu)}{\vdash G\otimes H, J, K}\,(\otimes)}{\vdash F\oplus(G\otimes H), J, K}\,(\oplus)}{\vdash F\oplus(G\otimes H), J\parr K}\,(\parr)$$

Actually, $\mathsf{rev}(\pi_1)$ and $\mathsf{rev}(\pi_2)$ are already focused, thus $\mathsf{Foc}(\pi)$ is:

$$\dfrac{\dfrac{(\star)}{(\star)\vdash I', L\otimes M}}{\dfrac{\vdash I, L\otimes M}{\vdash I, L\otimes M}}\,(\nu) \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(\dagger)}{\vdash G', J}}{(\dagger)\vdash G, J}\,(\nu) \quad \dfrac{\dfrac{(\ddagger)}{\vdash H', K}}{(\ddagger)\vdash H, K}\,(\nu)}{\vdash G\otimes H, J, K}\,(\otimes)}{\vdash F\oplus(G\otimes H), J, K}\,(\oplus)}{\vdash F\oplus(G\otimes H), J\parr K}\,(\parr)}{\vdash F\oplus(G\otimes H), I\otimes(J\parr K), L\otimes M}\,(\otimes)$$

Remark 4.1. In Example 4.6, we have seen that the focused proof of the circular proof π is also circular. Actually, this is not always the case. For insance, let α, β be two disjoint addresses, let $\varphi = \nu X.X\parr X$ and $\psi = \mu X.X\otimes X$ and consider the proof θ below, where $F = \varphi_\alpha$, $G = \psi_\beta$, $F' = \varphi_{\alpha.i.l}$, $G' = \psi_{\beta.i.l}$, $F'' = \varphi_{\alpha.i.r}$ and $G'' = \psi_{\beta.i.r}$. The proof $\mathsf{Foc}(\theta)$ is not circular since the size of its sequents is unbounded.

$$\theta = \dfrac{\dfrac{(\dagger)}{\vdash F'', G''} \quad \dfrac{(\dagger)}{\vdash F', G'}}{(\dagger)\vdash F, G}\,(\mu),\,(\nu),\,(\parr),\,(\otimes) \qquad \mathsf{Foc}(\theta) = \dfrac{\dfrac{\vdots}{\vdash F', F'', G}\,(\nu),(\parr)}{\vdash F, G}\,(\nu),\,(\parr)$$

Chapter 5

Ludics for linear logic with fixed points

In this chapter, we carry out a semantical investigation for μMALL, the finitary proof system for linear logic with least and greatest fixed points. Our domain of interpretation is computational ludics [Ter11], a framework built around the notion of design, which can be seen as an analogue of the strategies of game semantics. The infinitary nature of designs makes them particularly well suited for representing computations over infinite data.

We work with the system μMALLP, which is a polarized version of μMALL. Going through a polarized syntax is a requirement when the domain of interpretation is ludics, but it is not a serious restriction as we shall discuss later.

The first contribution is a denotational semantics for μMALLP: we interpret every proof as a design and every formula as a well-behaved set of designs, then we show that if π is a proof of a formula φ, then the interpretation of π belongs to the interpretation of φ, this is called a **soundness result**. We show also that the interpretation of a proof is stable under cut-elimination steps, that is why we say that our semantics is **denotational**.

Completeness is the converse of soundness: every design belonging to the interpretation of a formula φ is the interpretation of a proof of φ. Stated this way, completeness obviously does not hold in our setting. A weaker form of completeness is **completeness relatively** to a condition C on designs: if a design satisfying C belongs to the interpretation of φ then it is the interpretation of a proof of φ. The condition we came with is being **essentially finite**: essentially finite designs (EFD) are designs performing a finite computation followed by a copycat, which is the design interpreting η-expansions. To prove completeness relatively to EFD, we investigate semantic inclusions for arbitrary formulas showing that: If φ and ψ are two formulas such that the interpretation of φ is a subset of that of ψ, then $\varphi \vdash \psi$ is provable in μMALLP. This last result relies on a circular proof system, as a stepping stone between infinite designs and finite proofs.

This chapter is organized as follows. We introduce μMALLP, the polarized variant of μMALL, in Section 5.1 followed by ludics in Section 5.2. We then define the interpretation of μMALLP proofs and formulas in ludics in Section 5.3, then we prove that it is a sound and denotational interpretation. Finally, we establish completeness for EFD in Section 5.4, building on the result on semantic inclusions in μMALLP.

5.1 Polarized linear logic with fixed points

As usual when aiming at interpretations in ludics [Gir01, BF11, Ter11] we will actually be working with a polarized version of μMALL [Bae12b], μMALLP, to which the present section is dedicated.

5.1.1 Formulas

We assume two infinite and disjoint sets $\mathcal{V}_P$ and $\mathcal{V}_N$, whose elements are respectively called positive and negative variables and denoted by X_P and X_N, or simply X when their polarity is irrelevant or can be inferred from the context.

Definition 5.1. The sets of ***positive formulas*** $\varphi, \psi, \ldots$ and of ***negative formulas*** $\Phi, \Psi, \ldots$ are inductively defined by the following grammar:

$$\begin{array}{rcl} \varphi,\ \psi & ::= & X_P \mid X_N^\perp \mid 1 \mid 0 \mid \Psi \oplus \Phi \mid \Psi \otimes \Phi \mid \downarrow \Psi \mid \mu X_N.\varphi \\ \Psi, \Phi & ::= & X_N \mid X_P^\perp \mid \bot \mid \top \mid \varphi \,\&\, \psi \mid \varphi \parr \psi \mid \uparrow \varphi \mid \nu X_P.\Psi \end{array}$$

Where $X_N \in \mathcal{V}_N$ and $X_P \in \mathcal{V}_P$. Free and bound variables, capture-free substitution and monotonicity of formulas are defined as usual. A ***positive occurrence*** (resp. ***negative occurrence***) is given by a positive (resp. negative) formula α and an address α and it is written φ_α. We use $P, Q, \ldots$ for positive occurrences and $M, N, \ldots$ for negative ones.

Our syntax classifies μ as positive and ν as negative, this is consistent with the observations made in the study of focusing for $\mu\mathsf{MALL}^\infty$ in Chapter 4.

Remark 5.1. Note that due to our polarity constraint, the μMALLP fixed point formulas cannot alternate immediately two different fixed points, in other words, every fixed point formula is of the form $\sigma X_1, \ldots \sigma X_n.c(\varphi_1, \ldots, \varphi_n)$ where σ is either μ or ν and c a MALLP connective. This is not a real constraint, since the interleaving of fixed points can be obtained by using shifts between two blocks of different fixed points. To simplify the presentation at some points, we will suppose that every layer of fixed points contains only one connective.

Definition 5.2. ***Negation*** is the involutive operation mapping positive to negative formulas, and vice versa, such that:

$$\begin{array}{c} (\varphi \parr \psi)^\perp = \psi^\perp \otimes \varphi^\perp \quad (\varphi \,\&\, \psi)^\perp = \psi^\perp \oplus \varphi^\perp \quad (\uparrow \varphi)^\perp = \downarrow \varphi^\perp \\ (\nu X_N.\varphi)^\perp = \mu Y_P.(\varphi^\perp[Y_P^\perp / X]) \quad (X_N)^\perp = X_N^\perp \quad \top^\perp = 0 \quad \bot^\perp = 1 \end{array}$$

When considering a substitution $F[\vec{G}/\vec{X}]$, we always assume implicitly that the polarities of $\vec{G}$ are adequate to those of $\vec{X}$.

5.1.2 The proof system μMALLP

Definition 5.3. The proof system μMALLP is given in Figure 5.1. It is a focused sequent calculus over our polarized syntax, meaning that its sequents must contain at most one negative occurrence. A sequent is said to be ***negative*** when it contains a negative occurrence, and it is ***positive*** otherwise. In Figure 5.1, Γ and Δ always denote positive sequents.

Identity rules

$$\frac{}{\vdash P, P^\perp}\ (\mathsf{Ax}) \qquad \frac{\vdash \Gamma, P^\perp \quad \vdash \Delta, P}{\vdash \Gamma, \Delta}\ (\mathsf{Cut})$$

Logical rules

$$\frac{}{\vdash \Gamma, \top}\ (\top) \qquad \frac{\vdash \Gamma}{\vdash \Gamma, \bot}\ (\bot) \qquad \frac{}{\vdash 1}\ (1)$$

$$\frac{\vdash \Gamma, P}{\vdash \Gamma, \uparrow P}\ (\uparrow) \qquad \frac{\vdash \Gamma, N}{\vdash \Gamma, \downarrow N}\ (\downarrow) \qquad \frac{\vdash \Gamma, P_1 \quad \vdash \Gamma, P_2}{\vdash \Gamma, P_1 \,\&\, P_2}\ (\&)$$

$$\frac{\vdash \Gamma, N_i}{\vdash \Gamma, N_1 \oplus N_2}\ (\oplus) \qquad \frac{\vdash \Gamma, P_1, P_2}{\vdash \Gamma, P_1 \parr P_2}\ (\parr) \qquad \frac{\vdash \Delta, N_1 \quad \vdash \Gamma, N_2}{\vdash \Gamma, \Delta, N_1 \otimes N_2}\ (\otimes)$$

Fixed points rules

$$\frac{\vdash \Gamma, P[\mu X.P/X]}{\vdash \Gamma, \mu X.P}\ (\mu) \qquad \frac{\vdash \Gamma, M \quad \vdash N^\perp, N[M/X]}{\vdash \Gamma, \nu X.N}\ (\nu)$$

In these rules, Γ and Δ denote positive sequents.

Figure 5.1: The μMALLP sequent calculus proof system.

Reading proofs in a proof search (bottom-up) fashion, the polarity restriction on sequents means that negative rules must be applied eagerly, *i.e.*, as soon as the sequent contains a negative formula. This constraint on the shape of proofs is a very mild form of focusing.

Relating μMALL and μMALLP. Note that we can simply translate between μMALL and μMALLP, in the same way as is done between MALL and MALLP: μMALLP formulas and proofs can be embedded into μMALL by erasing shifts. In the other direction, μMALL formulas are translated into μMALLP formulas by inserting shift connectives, and any μMALL proof can be turned into a μMALLP proof of the translated conclusion sequent by inserting shift rules. This translation essentially cancels the focusing constraint of the μMALLP proof system by inserting shifts. A more demanding task would be to establish completeness of the focused μMALLP proof system given here with respect to an unfocused proof system for μMALLP. We do not address this (unrelated) issue, but expect that it would be possible along the lines of the focusing result for μMALL [Bae12b].

Cut elimination for μMALLP. Cut elimination holds for μMALLP: the cut reduction system given in [Bae12b] can straightforwardly be adapted to the polarized setting of μMALLP. The only non-trivial case is the one involving least and greatest fixed points. This cut reduction step, shown in Figure 5.2, relies on the **functoriality**, a proof construction that is

$$\dfrac{\dfrac{\dfrac{\Pi_L}{\vdash \Gamma, N^\perp[(\nu X.N)^\perp/X^\perp]}}{\vdash \Gamma, (\nu X.N)^\perp}\,(\mu) \qquad \dfrac{\dfrac{\Pi_R}{\vdash \Delta, M} \qquad \dfrac{\Theta}{\vdash M^\perp, N[M/X]}}{\vdash \Delta, \nu X.N}\,(\nu)}{\vdash \Gamma, \Delta}\,(\mathsf{Cut})$$

$$\Downarrow$$

$$\dfrac{\dfrac{\Pi_R}{\vdash \Delta, M} \qquad \dfrac{\dfrac{\Theta}{\vdash M^\perp, N[M/X]} \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\vdash M^\perp, M}\,(\mathsf{Ax}) \qquad \dfrac{\Theta}{\vdash M^\perp, N[M^\perp/X]}}{\vdash M^\perp, \nu X.N}\,(\nu)}{\vdash N^\perp[M^\perp/X^\perp], N[\nu X.N/X]}\,(\mathsf{F_N}) \qquad \dfrac{\Pi_L}{\vdash N^\perp[(\nu X.N)^\perp/X^\perp], \Gamma}}{\vdash N^\perp[M^\perp/X^\perp], \Gamma}\,(\mathsf{Cut})}{\vdash M^\perp, \Gamma}\,(\mathsf{Cut})}{\vdash \Gamma, \Delta}\,(\mathsf{Cut})$$

Figure 5.2: Cut-elimination step $(\mu)/(\nu)$.

used to derive the following rule:

$$\dfrac{\vdash P, N}{\vdash B[P/X], B^\perp[N/X^\perp]}\,(\mathsf{F_B})$$

In the cut reduction step of Figure 5.2, functoriality on P has the effect of making the sub-occurrence $\nu X.N$ of $N[\nu X.N/X]$ appear at top level, so that we can apply the (ν) rule on it. To adapt the reduction rules of μMALL to the case of μMALLP, we have to show that the functoriality construction is also derivable in μMALLP, that is what we do in the following.

Definition 5.4. Let B be a μMALLP occurrence and X be a variable. Let G (resp. H) be an occurrence with the same (resp. opposite) polarity as X and let Π be a proof of $\vdash G, H$. We define $\mathsf{F_B}(\Pi)$ to be the μMALLP proof of conclusion $\vdash B[G/X], B^\perp[H/X^\perp]$, defined by induction on the maximum depth of the free occurrences of X in B as follows.

- If X is not free in B then we define $\mathsf{F_B}(\Pi)$ to be the axiom rule on B.
- If $B = X_\alpha$, then $\mathsf{F_B}(\Pi)$ is obtained from Π by relocating G in α and H in $\alpha^\perp$.

Otherwise, we perform an η-expansion based on the top-level connective of B and conclude by induction hypothesis. We only show half of the connectives, because dual connectives are treated symmetrically.

- If $B = B_1 \otimes B_2$, we define $\mathsf{F_B}(\Pi)$ to be:

$$\dfrac{\dfrac{\dfrac{\mathsf{F}_{\mathsf{B}_1}(\Pi)}{\vdash B_1[G/X], B_1^\perp[H/X^\perp]} \qquad \dfrac{\mathsf{F}_{\mathsf{B}_2}(\Pi)}{\vdash B_2[G/X], B_2^\perp[H/X^\perp]}}{\vdash B_1[G/X] \otimes B_2[G/X], B_1^\perp[H/X^\perp], B_2^\perp[H/X^\perp]}\,(\otimes)}{\vdash B_1[G/X] \otimes B_2[G/X], B_1^\perp[H/X^\perp] \parr B_2^\perp[H/X^\perp]}\,(\parr)$$

- If $B = B_1 \oplus B_2$, we define $\mathsf{F}_B(\Pi)$ to be:

$$\dfrac{\dfrac{\dfrac{\mathsf{F}_{B_1}(\Pi)}{\vdash B_1[G/X], B_1^\perp[H/X^\perp]}}{\vdash B_1[G/X] \oplus B_2[G/X], B_1^\perp[H/X^\perp]}\,(\oplus) \qquad \dfrac{\dfrac{\mathsf{F}_{B_2}(\Pi)}{\vdash B_2[G/X], B_2^\perp[H/X^\perp]}}{\vdash B_1[G/X] \oplus B_2[G/X], B_2^\perp[H/X^\perp]}\,(\oplus)}{\vdash B_1[G/X] \oplus B_2[G/X], B_1^\perp[H/X^\perp] \,\&\, B_2^\perp[H/X^\perp]}\,(\&)$$

- If $B = \mu Y.C$, where $C = C_1 \otimes C_2$, we set $S = \mu Y.C[G/X]$ and $\mathsf{F}_B(\Pi)$ to be:

$$\dfrac{\dfrac{}{\vdash S, S^\perp}\,(\mathsf{Ax}) \qquad \dfrac{\Theta}{\vdash S, C^\perp[S^\perp/Y^\perp][H/X^\perp]}}{\vdash \mu Y.C[G/X], \nu Y.C^\perp[Y/Y^\perp][H/X^\perp]}\,(\nu)$$

  where Θ is defined by:

$$\dfrac{\dfrac{\dfrac{\mathsf{F}_{C_1[S/Y]}(\Pi)}{\vdash C_1[S/X][G/X], C_1^\perp[S^\perp/Y^\perp][H/X^\perp]} \qquad \dfrac{\mathsf{F}_{C_2[S/Y]}(\Pi)}{\vdash C_2[S/Y][G/X], C_2^\perp[S^\perp/Y^\perp][H/X^\perp]}\,(\mu),(\otimes)}{\vdash S, C_1^\perp[S^\perp/Y^\perp][H/X^\perp], C_2^\perp[S^\perp/Y^\perp][H/X^\perp]}\,(\parr)}{\vdash S, C^\perp[S^\perp/Y^\perp][H/X^\perp]}$$

- If $B = \mu Y.C$, where $C = C_1 \oplus C_2$, we set $S = \mu Y.C[G/X]$ and $\mathsf{F}_B(\Pi)$ to be:

$$\dfrac{\dfrac{}{\vdash S, S^\perp}\,(\mathsf{Ax}) \qquad \dfrac{\Theta}{\vdash S, C^\perp[S^\perp/Y^\perp][H/X^\perp]}}{\vdash \mu Y.C[G/X], \nu Y.C^\perp[Y/Y^\perp][H/X^\perp]}\,(\nu)$$

  where Θ is the proof defined by:

$$\dfrac{\dfrac{\dfrac{\mathsf{F}_{C_1[S/Y]}(\Pi)}{\vdash C_1[S/Y][G/X], C_1^\perp[S^\perp/Y^\perp][H/X^\perp]}\,(\mu),(\oplus)}{\vdash S, C_1^\perp[S^\perp/Y^\perp][H/X^\perp]} \qquad \dfrac{\dfrac{\mathsf{F}_{C_2[S/Y]}(\Pi)}{\vdash C_2[S/Y][G/X], C_2^\perp[S^\perp/Y^\perp][H/X^\perp]}\,(\mu),(\oplus)}{\vdash S, C_2^\perp[S^\perp/Y^\perp][H/X^\perp]}\,(\&)}{\vdash S, C^\perp[S^\perp/Y^\perp][H/X^\perp]}$$

Remark 5.2. Notice that the derivation $\mathsf{F}_B(\Pi)$ is of the following form:

$$\dfrac{\dfrac{\Pi}{\vdash G, H}}{\vdash B[G/X], B^\perp[H/X^\perp]}$$

where the rules applied between the sequents $\vdash B[G/X], B^\perp[H/X^\perp]$ and $\vdash G, H$ do not depend on Π. That is why we usually write functoriality on B as a rule named B.

Remark 5.3. The main difference between the above construction of functoriality and the one introduced in Definition 2.23 is that, since we are in a polarized proof system, we cannot apply a μ rule to the blue sequents in the last two cases, so we were obliged to decompose the top level connective of $C^\perp$ before. We do not have such constraint in μMALL, thus to two last cases mege in a single one, since we do not need to figure out the shape of C.

The proof of cut-elimination given in [Bae12b] can be adapted in an easy way to μMALLP.

5.2 Computational ludics

Ludics is an interactive framework reminiscent from and somehow intermediate between game semantics [HO00] and realizability [Kri09]. We recall, in the setting of computational ludics [Ter11], the necessary definitions and properties of:

- designs (§ 5.2.1), which correspond to strategies,
- orthogonality (§ 5.2.2), which corresponds to interaction, and
- behaviours (§ 5.2.3), which correspond to arenas or interactive types.

In game semantics, arenas are defined first and strategies are defined as sets of plays (or as sets of views) compatible with these arenas. However, in ludics, arenas are a secondary notion, derived from that of designs since behaviours are obtained from designs by orthogonality; we develop this comparison in § 5.2.4.

5.2.1 Designs

Designs are built over a ***signature*** $\mathcal{A} = (A, \mathrm{ar})$, where A is a set of names $a, b, c, \dots$ and $\mathrm{ar} : A \to \mathbb{N}$ is a function which assigns to each name a its *arity* $\mathrm{ar}(a)$. Let V be a set of variables $V = \{x, y, z, \dots\}$.

Definition 5.5. For a fixed signature $\mathcal{A}$, the class of ***positive designs*** $p, q, \dots$ and ***negative designs*** $n, m, \dots$ are coinductively defined as follows (with $\mathrm{ar}(a) = \mathsf{card}(\vec{x}_a) = k$):

$$\begin{array}{rcccccc} p & ::= & \Omega & | & \maltese & | & (n_0 \mid \overline{a}\langle n_1, \dots, n_k\rangle) \\ n & ::= & x & | & \sum a(\vec{x}_a).p_a & & \end{array}$$

The formal sum $\sum a(\vec{x}_a).p_a$ is the $\mathcal{A}$-indexed family $\{a(\vec{x}_a).p_a\}_{a\in\mathcal{A}}$.

Notation 5.1. We introduce in the following some useful notations.

- We write $\sum_{K\subseteq\mathcal{A}} a(\vec{x}_a).p_a$ to denote the ***design*** $\sum a(\vec{x}_a).q_a$ where $q_a = p_a$ if $a \in K$ and $q_a = \Omega$ otherwise.
- We denote by Ω^- the design $\sum a(\vec{x}_a).\Omega$.

In a negative design $\sum a(\vec{x}_a).p_a$, $a(\vec{x}_a)$ binds the variables $\vec{x}_a$ appearing in p_a. Variables which are not under the scope of a binder are free. The free variables of a design d are denoted by $\mathsf{fv}(d)$. We identify two designs which are α-equivalent, *i.e.*, which are equal up to renaming of their bound variables. We denote by $d[\vec{n}/\vec{x}]$ the design obtained by a simultaneous and capture-free substitution of the variables $\vec{x}$ by the negative designs $\vec{n}$. The reader is referred to [Ter11] for precise definitions.

Definition 5.6.

- A design of the form $n_0 \mid \overline{a}\langle n_1, \ldots, n_k\rangle$ where n_0 is not a variable is called a ***cut***.
- An occurrence of a variable x is called an ***identity*** if it occurs as $n_0 \mid \overline{a}\langle n_1, \ldots, x, \ldots, n_k\rangle$.
- We call a design ***identity-free*** (resp. ***cut-free***) if it does not contain an identity (resp. a cut) as a subdesign.
- A design d is called ***linear*** if for every positive subdesign $n_0 \mid \overline{a}\langle n_1, \ldots, n_k\rangle$ of d, the sets $\mathsf{fv}(n_0), \ldots, \mathsf{fv}(n_k)$ are pairwise disjoint.
- An ***l-design*** is a design d which is linear, identity-free and such that $\mathsf{fv}(d)$ is finite.
- A ***standard*** design is a cut-free l-design.

In order to interpret μMALLP proofs, the following signature, and associated notations, are useful:

Definition 5.7. The MALL ***signature*** is given by the set of names $\mathcal{A} = \{\bot, \uparrow, \&_1, \&_2, \parr\}$ whose arities are:

$$\mathrm{ar}(\bot) = 0, \mathrm{ar}(\uparrow) = \mathrm{ar}(\&_1) = \mathrm{ar}(\&_2) = 1 \qquad \mathrm{ar}(\parr) = 2$$

When considering this signature, we write 1 rather than $\overline{\bot}$, $\downarrow$ rather than $\overline{\uparrow}$, $\oplus_i$ rather than $\overline{\&_i}$ and $\otimes$ rather than $\overline{\parr}$.

We define in the following the design η_F, the infinitary η-expansion of the axiom over F, called also a ***copycat*** design:

Definition 5.8. Let F be a μMALLP occurrence. The ***design*** η_F is coinductively defined as follows:

$$\begin{array}{lcll}
\eta_{F_1 \otimes F_2} & = & \eta_{F_1 \parr F_2} & = \parr(x_1, x_2).(x_0 \mid \otimes\langle \eta_{F_1}[x_1/x_0], \eta_{F_2}[x_2/x_0]\rangle) \\
\eta_{F_1 \oplus F_2} & = & \eta_{F_1 \& F_2} & = \sum_{i=1,2} \&_i(x_i).(x_0 \mid \oplus_i\langle \eta_{F_i}[x_i/x_0]\rangle) \\
\eta_{\downarrow F} & = & \eta_{\uparrow F} & = \uparrow(x_1).(x_0 \mid \downarrow\langle \eta_F[x_1/x_0]\rangle) \\
\eta_{\sigma Y.F} & = & \eta_{F[\sigma Y.F/Y]} & \text{for } \sigma \in \{\mu, \nu\}
\end{array}$$

Example 5.1. The following two designs are examples of designs defined on the MALL signature:

$$\begin{array}{lcl}
d_1 & = & \&_1(x_1).(x_1 \mid 1) + \&_2(x_2).(x_2 \mid \oplus_1\langle\uparrow(y).(y \mid 1)\rangle) \\
d_2 & = & \parr(x_1, x_2).(x_2 \mid \downarrow\langle d_2\rangle)
\end{array}$$

Remark 5.4. Designs on the MALL signature can be viewed as abstractions of μMALLP proofs. For instance d_1 abstracts the (unique) cut-free proof of $\vdash 1 \,\&\, (\uparrow 1 \oplus \bot)$.

The previous remark is the basis of the usual interpretation of MALL in ludics that we will extend, in the rest of the chapter, into an interpretation of μMALLP. But first, as ludics is all about interaction, we turn to cut-elimination and orthogonality.

5.2.2 Cut-elimination and Orthogonality

Cuts can be reduced by the relation $\rightarrow$ defined as follows:

Definition 5.9. The ***relation*** $\rightarrow$ is defined on positive designs as follows:

$$(\sum a(\vec{x}_a).p_a) \mid \overline{b}\langle\vec{n}\rangle \rightarrow p_b[\vec{n}/\vec{x}_b].$$

The reflexive and transitive closure of $\rightarrow$ is denoted $\rightarrow^\star$.

We write $p \Downarrow q$ if $p \rightarrow^\star q$ and q is neither a cut, nor the design Ω. If such a design q does not exist, we write $p \Uparrow$.

We define $\perp\!\!\!\perp$ to be the least set of positive designs containing $\maltese$ and closed by anti-reduction: $\perp\!\!\!\perp = \{d : d \rightarrow^\star \maltese\}$.

To eliminate cuts from a design, we coinductively propagate the relation $\Downarrow$ to it sub-designs, in a way much reminiscent to Böhm trees. The obtained design is called its normal form, we define this operation as follows:

Definition 5.10. The ***function*** $(\!|.|\!)$ on designs is coinductively defined by:

$$\begin{array}{lcll}
(\!|p|\!) & = & \maltese & if \quad p \Downarrow \maltese, \\
 & = & \Omega & if \quad p \Uparrow, \\
 & = & x \mid \overline{a}\langle(\!|n_1|\!), \ldots, (\!|n_k|\!)\rangle & if \quad p \Downarrow x \mid \overline{a}\langle n_1, \ldots, n_k\rangle, \\
(\!|x|\!) & = & x, & \\
(\!|\sum a(\vec{x}).p_a|\!) & = & \sum a(\vec{x}).(\!|p_a|\!). &
\end{array}$$

The normal form $(\!|d|\!)$ enjoys a weak form of Church-Rosser property, called **associativity**. We recall it in the following:

Proposition 5.1 ([Ter11]). *Let d be a design and $n_1, \ldots, n_k$ be negative designs. We have:*

$$(\!|d[n_1/x_1, \ldots, n_k/x_k]|\!) = (\!|(\!|d|\!)[(\!|n_1|\!)/x_1, \ldots, (\!|n_k|\!)/x_k]|\!)$$

We finally define an orthogonality relation on so-called atomic designs.

Definition 5.11. A positive standard design p is ***atomic*** if it has at most one free variable, that variable will be called x_0 in the rest of the paper.

A negative standard design n is ***atomic*** if it is closed, *i.e.*, $\mathsf{fv}(n) = \emptyset$.

Definition 5.12. Let p be a positive atomic design and n a negative atomic design. The designs p and n are said to be ***orthogonal*** (written $p \perp n$) if $p[n/x_0] \in \perp\!\!\!\perp$.

Given a set $\mathbf{X}$ of atomic designs of the same polarity, we define its orthogonal $\mathbf{X}^\perp$ as follows:

$$\mathbf{X}^\perp = \{\ e \mid \forall d \in \mathbf{X},\ d \perp e\ \}$$

The orthogonality relation enjoys the following useful properties:

Proposition 5.2 ([Ter11]). *Let* $\mathbf{X}$, $\mathbf{Y}$ *be sets of atomic designs of the same polarity. We have that:*

$$1)\ \mathbf{X} \subseteq \mathbf{X}^{\perp\perp} \qquad 2)\ \mathbf{X} \subseteq \mathbf{Y} \Rightarrow \mathbf{Y}^\perp \subseteq \mathbf{X}^\perp$$

$$3)\ \mathbf{X}^\perp = \mathbf{X}^{\perp\perp\perp} \qquad 4)\ (\mathbf{X} \cup \mathbf{Y})^\perp = \mathbf{X}^\perp \cap \mathbf{Y}^\perp$$

5.2.3 Behaviours, Sets of Designs

We define in the following **behaviours**, which are those sets of atomic designs closed by bi-orthogonality:

Definition 5.13. A ***behaviour*** is a set $\mathbf{X}$ of atomic designs of the same polarity such that $\mathbf{X} = \mathbf{X}^{\perp\perp}$. We denote by $\mathcal{C}_P$ (resp. $\mathcal{C}_N$) the set of all positive (resp. negative) behaviours.

The set $\mathcal{C}_P$, ordered by set inclusion, forms a complete lattice: using Proposition 5.2, we prove easily that every collection of positive behaviours $\vec{\mathbf{S}}$ has $(\bigcup \vec{\mathbf{S}})^{\perp\perp}$ as a least upper bound and $(\bigcap \vec{\mathbf{S}})^{\perp\perp} = \bigcap \vec{\mathbf{S}}$ as a greatest lower bound. Thus the Knaster-Tarski theorem guarantees the existence of least and greatest fixed points of monotonic operators on $\mathcal{C}_P$.

We generalize the relation $d \in \mathbf{C}$ between atomic designs and behaviours into the relation $d \models \mathbf{\Gamma}$ between designs with arbitrary free variables and contexts of behaviours:

Definition 5.14. A ***positive context*** $\mathbf{\Gamma}$ is a set of pairs $x_1 : \mathbf{P}_1, \ldots, x_k : \mathbf{P_k}$ where $x_1, \ldots, x_k$ are distinct variables and $\mathbf{P}_1, \ldots, \mathbf{P_k}$ are positive behaviours.

A ***negative context*** $\mathbf{\Gamma}, \mathbf{N}$ is a positive context $\mathbf{\Gamma}$ together with a negative behaviour $\mathbf{N}$, to which no variable is associated.

Definition 5.15. Let $\mathbf{\Gamma} = x_1 : \mathbf{P}_1, \ldots, x_k : \mathbf{P_k}$ be a positive context, $\mathbf{\Gamma}, \mathbf{N}$ be a negative context and p (resp. n) be a positive (resp. negative) standard design. We define the ***relation*** $\models$ between designs and contexts of the same polarity as follows:

$$p \models \mathbf{\Gamma} \quad \text{iff} \quad p[n_1/x_1, \ldots, n_k/x_k] \in \perp\!\!\!\perp \quad \text{for any} \quad n_1 \in \mathbf{P}_1{}^\perp, \ldots, n_k \in \mathbf{P_k}{}^\perp,$$

$$n \models \mathbf{\Gamma}, \mathbf{N} \quad \text{iff} \quad p[n[n_1/x_1, \ldots, n_k/x_k]/x_0] \in \perp\!\!\!\perp \quad \text{for any} \quad p \in \mathbf{N}^\perp, n_1 \in \mathbf{P}_1{}^\perp, \ldots, n_k \in \mathbf{P_k}{}^\perp.$$

Remark that $p \models x_0 : \mathbf{P}$ if and only if $p \in \mathbf{P}$ and $n \models \mathbf{N}$ if and only if $n \in \mathbf{N}$. More generally, the following closure principle [Gir01, Ter11] will be useful in the following sections.

Proposition 5.3 (Closure principle)**.**

$$d \models \mathbf{\Sigma}, z : \mathbf{P} \quad \textit{iff} \quad \forall m \in \mathbf{P}^\perp,\ (\!|d[m/z]|\!) \models \mathbf{\Sigma} \quad \textit{where } \mathbf{\Sigma} \textit{ is a positive or negative context.}$$

$$n \models \mathbf{\Gamma}, \mathbf{N} \quad \textit{iff} \quad \forall q \in \mathbf{N}^\perp,\ (\!|q[n/x_0]|\!) \models \mathbf{\Gamma} \quad \textit{where } \mathbf{\Gamma} \textit{ is a positive context.}$$

5.2.4 Designs as Strategies

We end this background section on ludics by providing some more details on the comparison between HO game semantics [HO00] and ludics.

In HO game semantics, one first defines arenas which specify the moves of the game and which induce plays. In a second step, strategies are defined, as sets of plays satisfying various conditions (such as totality, determinism, innocence, *etc.*) depending on what is modeled. While arenas interpret formulas (or types), strategies will interpret proofs (or programs).

In ludics, the construction proceeds in the other direction, more akin to realizability models: a notion of abstract proof (design) serves as our notion of strategy while arenas are replaced by behaviours, that are sets of designs closed under bi-orthogonality. Strategies and interaction thus come first and only afterwards comes the notion of arena: the moves of the game are defined as a by-product of the way the objects interact.

The comparison can be made more precise when comparing innocent game semantics and ludics [FH02, BF11]. Indeed, with *innocent* strategies, a player's move does not depend on the full play that precedes it but only on a restriction of the play, its *view*. A view typically excludes the part of the play which corresponds to intermediate computations of the opponent, retaining only opponent's results and not how the results were obtained. As a consequence, innocent strategies can be presented as *sets of views* with some conditions. Ludics fits this presentation as designs can be seen as sets of views: each branch of a design is a view.

To conclude this comparison, let us stress that on the one hand, game semantics puts constraints on the way arenas are built but it is then rather flexible on the definition of strategies (by enforcing or relaxing various constraints on the structure of strategies). On the other hand, ludics puts constraints on the design of strategies (for instance to preserve analytical theorems on which internal completeness depends) and is quite flexible on how arenas are defined. This difference explains why it revealed to be much more difficult to model LL exponentials in ludics than in HO game semantics. The very same reason explains why it will be smoother to interpret fixed points in ludics than in HO game semantics [Cla09].

5.3 Interpretation of μMALLP in Ludics

We now define a semantics for our system in ludics, extending the usual interpretation of MALL in computational ludics [Ter11]: formulas will be interpreted by behaviours and proofs by designs. From now on, we restrict to the MALL signature from Definition 5.7.

5.3.1 Interpretation of Formulas

Definition 5.16. Let θ be a formula and $\mathcal{E}$ an ***environment*** mapping each free variable of θ to a behaviour of the same polarity. We define by induction on θ a behaviour called

the ***interpretation of θ under $\mathcal{E}$*** and denoted by $[\![\theta]\!]^{\mathcal{E}}$.

$$[\![X]\!]^{\mathcal{E}} = \mathcal{E}(X), \quad [\![0]\!]^{\mathcal{E}} = \emptyset^{\perp\perp}, \quad [\![1]\!]^{\mathcal{E}} = \{(x_0 \mid 1)\}^{\perp\perp},$$

$$[\![\downarrow\varphi]\!]^{\mathcal{E}} = \{x_0 \mid \downarrow\langle r\rangle \mid r \in [\![\varphi]\!]^{\mathcal{E}}\}^{\perp\perp},$$

$$[\![\varphi_1 \otimes \varphi_2]\!]^{\mathcal{E}} = \{(x_0 \mid \otimes\langle r_1, r_2\rangle) \mid r_i \in [\![\varphi_i]\!]^{\mathcal{E}}\}^{\perp\perp},$$

$$[\![\varphi_1 \oplus \varphi_2]\!]^{\mathcal{E}} = \{(x_0 \mid \oplus_i\langle r_i\rangle) \mid i \in \{1,2\}, r_i \in [\![\varphi_i]\!]^{\mathcal{E}}\}^{\perp\perp},$$

$$[\![\mu X.\Phi]\!]^{\mathcal{E}} = \mathsf{lfp}(f) \quad \text{where} \quad f : \begin{array}{ccc} \mathcal{C}_P & \longrightarrow & \mathcal{C}_P, \\ \mathbf{C} & \longmapsto & [\![\Phi]\!]^{\mathcal{E}, X \mapsto \mathbf{C}} \end{array}$$

$$[\![\varphi]\!]^{\mathcal{E}} = ([\![\varphi^{\perp}]\!]^{\mathcal{E}})^{\perp} \quad \text{for all other cases.}$$

The interpretation of a formula θ in the empty environment is simply written $[\![\theta]\!]$. As usual, the interpretation of an occurrence $F = \theta_\alpha$ in the environment $\mathcal{E}$ is the interpretation of its underlying formula φ, we write it $[\![F]\!]^{\mathcal{E}}$.

The well-definedness of the interpretation of $\mu X.\Phi$ relies on the monotonicity of f, which is easily proved by induction on formulas. Our interpretation of formulas enjoys the usual substitution property, which entails that the interpretation of fixed points is stable under unfolding.

Proposition 5.4. *$[\![F[G/X]]\!]^{\mathcal{E}} = [\![F]\!]^{\mathcal{E}, X \mapsto [\![G]\!]^{\mathcal{E}}}$ and $[\![\mu X.P]\!]^{\mathcal{E}} = [\![P[\mu X.P/X]]\!]^{\mathcal{E}}$.*

The interpretation of MALL formulas relies on closure by orthogonality or by biorthogonality depending on their polarity. As a result, the shape of the elements of the resulting sets is not obvious. Nevertheless, the internal completeness theorem of ludics allows to characterize them. This is the role of Proposition 5.5, which can be shown in the same way as in [Ter11].

Proposition 5.5. *Let φ_1, φ_2 be two negative formulas and $p = x_0 \mid \overline{a}\langle\vec{n}\rangle$ a positive l-design.*

$$p \in [\![\varphi_1 \otimes \varphi_2]\!] \quad \textit{iff} \quad \overline{a} = \otimes, \vec{n} = (n_1, n_2) \textit{ and each } n_i \in [\![\varphi_i]\!],$$

$$p \in [\![\varphi_1 \oplus \varphi_2]\!] \quad \textit{iff} \quad \overline{a} = \oplus_i, \vec{n} = n_i \textit{ and } n_i \in [\![\varphi_i]\!] \textit{ for some } i,$$

$$p \in [\![\downarrow\varphi_1]\!] \quad \textit{iff} \quad \overline{a} = \downarrow, \vec{n} = n_1 \textit{ and } n_1 \in [\![\varphi_1]\!],$$

$$p \in [\![1]\!] \quad \textit{iff} \quad \overline{a} = 1, \vec{n} = \emptyset.$$

Let Φ_1, Φ_2 be two positive formulas and $n = \sum a(\vec{x}_a).p_a$ a negative l-design.

$$n \in [\![\Phi_1 \parr \Phi_2]\!] \quad \textit{iff} \quad \vec{x}_{\parr} = (x_1, x_2) \textit{ and } p_{\parr} \models x_1{:}[\![\Phi_1]\!], x_2{:}[\![\Phi_2]\!],$$

$$n \in [\![\Phi_1 \,\&\, \Phi_2]\!] \quad \textit{iff} \quad \vec{x}_{\&_i} = x_i \textit{ and } p_{\&_i} \models x_i{:}[\![\Phi_i]\!] \textit{ for all } i \in \{1,2\},$$

$$n \in [\![\uparrow\Phi_1]\!] \quad \textit{iff} \quad \vec{x}_{\uparrow} = x_1 \textit{ and } p_{\uparrow} \models x_1 : [\![\Phi_1]\!],$$

$$n \in [\![\bot]\!] \quad \textit{always holds.}$$

Note that these conditions constrain at most two designs in the sum, all others are arbitrary.

Similarly, the interpretation of a ν-formula is defined as the orthogonal of a least fixed point, but that is equivalent to the following more direct description as a greatest fixed point.

Proposition 5.6. $[\![\nu X.\varphi]\!]^{\mathcal{E}} = \mathsf{gfp}(f)$ *where* $f : \mathcal{C}_N \to \mathcal{C}_N$ *is such that* $f(\mathbf{C}) = [\![\varphi]\!]^{\mathcal{E},X\mapsto\mathbf{C}}$.

5.3.2 Interpretation of μMALLP Proofs

We interpret proofs compositionally, each rule corresponding to a construction on designs. Again, this extends the interpretation of MALL rules by Terui and Basaldella [BT09]. Proofs of positive sequents are interpreted by positive designs, and negative sequents by negative designs. In order to do so, we need to assign distinct name variables to positive occurrences.

Definition 5.17. We associate to every μMALLP positive occurrence P, a design variable denoted x_P, in such a way that $x_P = x_Q$ if and only if P is the unfolding of Q.

We first give the structure of the interpretation, in order to fix the ideas. Then we define the design construction $\mathbb{G}_{F,d}$ that is needed to interpret rule (ν).

Definition 5.18. Let π be a proof of a sequent Γ. The ***interpretation of*** π***, written*** $[\![\pi]\!]$, is defined by the rules of Figure 5.3. Each of these rules has the following form:

$$\frac{\{d_i \vdash \Gamma_i\}_{i\in I}}{d \vdash \Gamma}\ (r)$$

and stands for the following implication: If a proof π is obtained from the proofs $(\pi_i)_{i\in I}$ by applying rule (r), and if $[\![\pi_i]\!] = d_i$, then $[\![\pi]\!] = d$.

This interpretation may be understood by thinking of designs as proof terms, in the same way that, in intuitionistic logic, hypotheses are annotated by variables and proofs by λ-terms.

The interpretation of rules (Ax) and (Cut) is quite natural. The axiom over P is interpreted by the copycat design η_P and cut is interpreted by the normal form of the cut between the interpretations of the two sub-proofs. The interpretation of MALL rules is the same as in [BT09]. The interpretation of the (μ) rule is trivial, based on the fact that fixed point unfolding is transparent in our interpretation.

The main difficulty lies in the interpretation of the (ν) rule. Our goal is to interpret proofs by designs that reflect the computational behaviour of these proofs, thus we will derive the interpretation of rule (ν) from the cut reduction rule between μ and ν formulas presented in Figure 5.2. More precisely, our interpretation of rule (ν) is a design defined by an equality which expresses that the interpretation of the two proofs in Figure 5.2 are equal.

As this reduction rule involves the functoriality construction, we show first a construction which is the counterpart of functoriality in Ludics. More precisely, the construction $\mathbb{F}_{Q,d}$ is the functoriality of Q applied to a design d, and it is the counterpart in ludics of the construction $\mathsf{F_Q}(\Pi)$, the functoriality of Q applied to a proof Π.

Definition 5.19. Let d be a negative l-design and F an occurrence such that $\mathsf{fv}(d) \subseteq \{x\}$ and $\mathsf{fv}(F) \subseteq \{X\}$. The ***functoriality*** of F applied to d is the negative l-design $\mathbb{F}_{F,d}$

Identity rules

$$\frac{}{\eta_P[x_P/x_0] \vdash P, P^\perp}\ (\mathsf{Ax}) \qquad \frac{n \vdash \Gamma, P^\perp \qquad d \vdash \Delta, P}{(\!|d[n/x_P]|\!) \vdash \Gamma, \Delta}\ (\mathsf{Cut})$$

Logical rules

$$\frac{}{\Omega^- \vdash \Gamma, \top}\ (\top) \qquad \frac{p_1 \vdash \Gamma, P \qquad p_2 \vdash \Gamma, Q}{\&_1(x_P).p_1 + \&_2(x_Q).p_2 \vdash \Gamma, P \,\&\, Q}\ (\&) \qquad \frac{n \vdash \Gamma, N}{x_{\downarrow N} \mid \downarrow\langle n\rangle \vdash \Gamma, \downarrow N}\ (\downarrow)$$

$$\frac{p \vdash \Gamma}{\bot.p \vdash \Gamma, \bot}\ (\bot) \qquad \frac{p \vdash \Gamma, P, Q}{\parr(x_P, x_Q).p \vdash \Gamma, P \parr Q}\ (\parr) \qquad \frac{p \vdash \Gamma, P}{\uparrow(x_P).p \vdash \Gamma, \uparrow P}\ (\uparrow)$$

$$\frac{}{(x \mid 1) \vdash x : 1}\ (1) \qquad \frac{n_1 \vdash \Delta, N_1 \qquad n_2 \vdash \Gamma, N_2}{x \mid \otimes\langle n_1, n_2\rangle \vdash \Gamma, \Delta, N_1 \otimes N_2}\ (\otimes) \qquad \frac{n \vdash \Gamma, N_i}{x \mid \oplus_i\langle n\rangle \vdash \Gamma, N_1 \oplus N_2}\ (\oplus)$$

Fixed point rules

$$\frac{p \vdash \Gamma, P[\mu X.P/X]}{p \vdash \Gamma, \mu X.P}\ (\mu) \qquad \frac{n \vdash M^\perp, N[M/X] \qquad m \vdash \Gamma, M}{(\!|\mathbb{G}_{N,n}[m/x_{M^\perp}]|\!) \vdash \Gamma, \nu X.N}\ (\nu)$$

Where $x = x_{N_1 \otimes N_2}$ in the $\otimes$ rule and $x = x_{N_1 \oplus N_2}$ in the $\oplus$ rule.

Figure 5.3: Interpretation of μMALLP proofs.

coinductively defined by $\mathbb{F}_{F,d} = \eta_F$ when $fv(F) = \emptyset$, and otherwise:

$$\begin{aligned}
\mathbb{F}_{X,d} &= \mathbb{F}_{X^\perp,d} &&= d[x_0/x],\\
\mathbb{F}_{F_1 \otimes F_2,d} &= \mathbb{F}_{F_1 \parr F_2,d} &&= \parr(x_1, x_2).(x_0 \mid \otimes\langle \mathbb{F}_{F_1,d}[x_1/x_0], \mathbb{F}_{F_2,d}[x_2/x_0]\rangle),\\
\mathbb{F}_{F_1 \oplus F_2,d} &= \mathbb{F}_{F_1 \& F_2,d} &&= \textstyle\sum_{i=1,2} \&_i(x_i).(x_0 \mid \oplus_i\langle \mathbb{F}_{F_i,d}[x_i/x_0]\rangle),\\
\mathbb{F}_{\downarrow F,d} &= \mathbb{F}_{\uparrow F,d} &&= \uparrow(x_1).(x_0 \mid \downarrow\langle \mathbb{F}_{F,d}[x_1/x_0]\rangle),\\
\mathbb{F}_{\sigma Y.F,d} &= \mathbb{F}_{F[\sigma Y.F/Y],d} && \text{for } \sigma \in \{\mu, \nu\}.
\end{aligned}$$

Example 5.2. Let d be a negative design with $\mathsf{fv}(d) = x_0$ and $F = (\mu Y.X \parr Y)_\varepsilon$. The functoriality of F applied to d is the design defined by the following equation:

$$\mathbb{F}_{F,d} = \parr(x, y).(x_0 \mid \otimes\langle d[x/x_0], \mathbb{F}_{F,d}[y/x_0]\rangle)$$

The definition of functoriality in ludics naturally expresses the intended computational behaviour of that operation: $\mathbb{F}_{Q,d}$ is a modified η-expansion which behaves as d on occurrences of X in Q. This should be contrasted with the very involved formulation of $\mathsf{F_Q}(\Pi)$ in sequent calculus 5.4, which notably uses the (ν) rule to deal with fixed points encountered in Q.

Actually, the rule (ν) is obtained by agregating the cut rule to the following rule:

$$\frac{\vdash M^{\perp}, N[M/X]}{\vdash M^{\perp}, \nu X.N}\ (\nu)$$

If n is the interpretation of $\vdash M^{\perp}, N[M/X]$, we set $\mathbb{G}_{N,n}$ to be the (unknown) interpretation of $\vdash M^{\perp}, \nu X.N$, in other words, we have the following derivation:

$$\frac{n \vdash M^{\perp}, N[M/X]}{(\!|\mathbb{G}_{N,n}|\!) \vdash M^{\perp}, \nu X.N}\ (\nu)$$

Thus the interpretation of the rule (ν) is the following:

$$\frac{n \vdash M^{\perp}, N[M/X] \qquad m \vdash \Gamma, M}{(\!|\mathbb{G}_{N,n}[m/x_{M^{\perp}}]|\!) \vdash \Gamma, \nu X.N}\ (\nu)$$

When we interpret the proofs of Figure 5.2, the equality between to the resulting designs yields the following equation, which defines entirely $\mathbb{G}_{N,n}$.

Definition 5.20. Let n be a negative design and N a negative occurrence such that $\mathsf{fv}(n) \subseteq \{x\}$, $\mathsf{fv}(N) \subseteq \{X\}$ and $N \not\equiv X$.

The ***action*** of N on n is the design $\mathbb{G}_{N,n}$ coinductively defined by the following (productive) equation: $\mathbb{G}_{N,n} = \mathbb{F}_{N,\mathbb{G}_{N,n}}[n/x_0]$.

Since the (ν) rule is obtained by agregating a cut to the above simplified rule, we interpret it as in Figure 5.3.

5.3.3 Soundness and Invariance by Cut Elimination

We now show that the interpretation given above is sound, that is, if π is a proof of an occurrence F then the interpretation of π belongs to the interpretation of F. Before formally stating and proving this theorem, we generalize its statement to proofs of arbitrary sequents.

Definition 5.21. If $\Gamma = P_1, \ldots, P_n$ is a positive sequent, we interpret it by the positive context $[\![\Gamma]\!] = x_{P_1} : [\![P_1]\!], \ldots, x_{P_n} : [\![P_n]\!]$.

If Γ, N is a negative sequent, its interpretation is the negative context $[\![\Gamma, N]\!] = [\![\Gamma]\!], [\![N]\!]$.

Theorem 5.1. *If π is a proof of Γ, then $[\![\pi]\!] \models [\![\Gamma]\!]$.*

The theorem is proved by induction on π, and case analysis on its last rule. Soundness of rule (Ax) follows from the fact that $\eta_P \models x_0 : [\![P]\!], [\![P]\!]^{\perp}$. Soundness for (Cut) follows from the closure principle 5.3. The cases of MALL rules easily follow from the definition of formula interpretations. Soundness for rule (μ) is a direct consequence of Proposition 5.4. The difficulty lies in the (ν) rule, whose soundness relies on the following lemma, stating that $\mathbb{F}_{F,d}$ is sound.

Lemma 5.1. *Let d be a negative design, $\mathbf{P}, \mathbf{N}$ be two behaviours and F be a negative occurrence such that $\mathsf{fv}(d) \subseteq \{x\}$, $\mathsf{fv}(F) \subseteq \{X\} \subseteq \mathcal{V}_N$ and $d \models x : \mathbf{P}, \mathbf{N}$. Then we have:*

$$(\!|\mathbb{F}_{F,d}|\!) \models x_0 : [\![F^{\perp}]\!]^{X \mapsto \mathbf{P}^{\perp}}, [\![F]\!]^{X \mapsto \mathbf{N}}$$

Proof. We shall prove a generalized form of the proposition. Fix d, $\mathbf{P}$ and $\mathbf{N}$ as in the above statement. We say that a list of occurrences $\vec{G}$ is ***adequate*** to a list of variables $\vec{X}$ if they have the same length and, for all i, G_i and X_i have the same polarity.

Let F be a negative occurrence such that $\mathsf{fv}(F) \subseteq \vec{Y}$, and $\vec{U}$ be a list of occurrences that is adequate to $\vec{Y}$ and such that $\mathsf{fv}(\vec{U}) \subseteq \{X\}$. Finally, let $\vec{\mathbf{C}}$ and $\vec{\mathbf{E}}$ be lists of behaviours respectively adequate to $\vec{Y}$ and $\vec{Y}^\perp$ such that, for all $U_i \in \vec{U}$,

$$\begin{array}{ll} (\!|\mathbb{F}_{U_i,d}|\!) \models x_0 : \mathbf{C_i}, \mathbf{E_i} & \text{if } \mathbf{C_i} \text{ is positive} \\ (\!|\mathbb{F}_{U_i,d}|\!) \models x_0 : \mathbf{E_i}, \mathbf{C_i} & \text{if } \mathbf{C_i} \text{ is negative} \end{array}$$

We shall establish that:

$$(\!|\mathbb{F}_{F[\vec{U}/\vec{Y}],d}|\!) \models x_0 : [\![F^\perp]\!]^{\vec{Y}\mapsto\vec{\mathbf{C}}^\perp}, [\![F]\!]^{\vec{Y}\mapsto\vec{\mathbf{E}}}$$

We proceed by induction on F. We show the only difficult case, which is when $F = \nu Z.G$. We set $V = F[\vec{U}/\vec{Y}]$, and set out to show that:

$$(\!|\mathbb{F}_{V,d}|\!) \models x_0 : [\![F^\perp]\!]^{\vec{Y}\mapsto\vec{\mathbf{C}}^\perp}, [\![F]\!]^{\vec{Y}\mapsto\vec{\mathbf{E}}}$$

Let us consider the behaviour $\mathbf{S}$ defined by:

$$\mathbf{S} := \{\ (\!|(\mathbb{F}_{V,d})[m/x_0]|\!)\ :\ m \in ([\![F^\perp]\!]^{\vec{Y}\mapsto\vec{\mathbf{C}}^\perp})^\perp\ \}^{\perp\perp}$$

By the closure principle, it suffices to prove $\mathbf{S} \subseteq [\![F]\!]^{\vec{Y}\mapsto\vec{\mathbf{E}}}$. By Proposition 5.6, $[\![F]\!]^{\vec{Y}\mapsto\vec{\mathbf{E}}}$ is the greatest post-fixed point of $\varphi := \mathbf{C} \mapsto [\![G]\!]^{\vec{Y}\mapsto\vec{\mathbf{E}},Z\mapsto\mathbf{C}}$. Thus, it suffices to show that $\mathbf{S}$ is a post-fixed point of φ, which finally amounts, by the closure principle, to prove that:

$$(\!|\mathbb{F}_{V,d}|\!) \models x_0 : [\![F^\perp]\!]^{\vec{Y}\mapsto\vec{\mathbf{C}}^\perp}, [\![G]\!]^{\vec{Y}\mapsto\vec{\mathbf{E}},Z\mapsto\mathbf{S}}$$

Or equivalently, by unfolding in the functoriality and interpretation:

$$(\!|\mathbb{F}_{G[\vec{U}/\vec{Y},V/Z],d}|\!) \models x_0 : [\![G^\perp]\!]^{\vec{Y}\mapsto\vec{\mathbf{C}}^\perp, Z\mapsto[\![F^\perp]\!]^{\vec{Y}\mapsto\vec{\mathbf{C}}^\perp}}, [\![G]\!]^{\vec{Y}\mapsto\vec{\mathbf{E}},Z\mapsto\mathbf{S}}$$

This is obtained by induction hypothesis on G, since by definition of $\mathbf{S}$ we have that:

$$(\!|\mathbb{F}_{V,d}|\!) \models x_0 : [\![F^\perp]\!]^{\vec{Y}\mapsto\vec{\mathbf{C}}^\perp}, \mathbf{S}$$

Which concludes the proof. □

Let us show now the soundness of the rule (ν).

Proposition 5.7. *Let $\nu X.F$ be an occurrence, d be a design and $\mathbf{S}$ be a behaviour such that $d \models x_0 : \mathbf{S}, [\![F]\!]^{X\mapsto\mathbf{S}^\perp}$. We have that:*

$$(\!|\mathbb{G}_{F,d}|\!) \models x_0 : \mathbf{S}, [\![\nu X.F]\!]$$

Proof. By closure principle, we have that:

$$\begin{aligned} & (\!|\mathbb{G}_{F,d}|\!) \models x_0 : \mathbf{S}, [\![\nu X.F]\!] \\ \Leftrightarrow \quad & \forall m \in \mathbf{S}^\perp, (\!|\mathbb{G}_{F,d}[m/x_0]|\!) \models [\![\nu X.F]\!] \\ \Leftrightarrow \quad & \mathbf{S}_1 := \{\ (\!|\mathbb{G}_{F,d}[m/x_0]|\!)\ :\ m \in \mathbf{S}^\perp\ \}^{\perp\perp} \subseteq [\![\nu X.F]\!] \end{aligned}$$

But $[\![\nu X.F]\!] = \mathsf{gfp}(\varphi)$ where $\varphi = \mathbf{C} \mapsto [\![F]\!]^{X \mapsto \mathbf{C}}$, thus it suffices to establish that $\mathbf{S}_1$ is a post-fixed point of φ, *i.e.*, $\mathbf{S}_1 \subseteq [\![F]\!]^{X \mapsto \mathbf{S}_1}$. This is equivalent to:

$$\forall m \in \mathbf{S}^\perp, (\!|\mathbb{G}_{F,d}|\!)[m/x_0] \models [\![F]\!]^{X \mapsto \mathbf{S}_1}$$

and by closure principle to:

$$(\!|\mathbb{G}_{F,d}|\!) \models x_0 : \mathbf{S}, [\![F]\!]^{X \mapsto \mathbf{S}_1}$$

Remark that by definition of $\mathbf{S}_1$ we have $(\!|\mathbb{G}_{F,d}|\!) \models x_0 : \mathbf{S}, \mathbf{S}_1$. By Proposition 5.1, this gives us

$$(\!|\mathbb{F}_{F,\mathbb{G}_{F,d}}|\!) \models x_0 : [\![F^\perp]\!]^{X \mapsto \mathbf{S}^\perp}, [\![F]\!]^{X \mapsto \mathbf{S}_1}$$

By hypothesis, $d \models x_0 : \mathbf{S}, [\![F]\!]^{X \mapsto \mathbf{S}^\perp}$ so by the closure principle we have, as expected:

$$(\!|\mathbb{G}_{F,d}|\!) = (\!|\mathbb{F}_{F,\mathbb{G}_{F,d}}[d/x_0]|\!) \models x_0 : \mathbf{S}, [\![F]\!]^{X \mapsto \mathbf{S}_1}$$

□

As a second soundness result, we show that our semantics is denotational, *i.e.*, the interpretation is invariant by cut elimination. The proof of this theorem relies on the following lemma, which expresses that ludics functoriality $\mathbb{F}_{F,d}$ is the semantical counterpart of the functoriality in sequent calculus (Definition 5.4).

Lemma 5.2. *Let* Π *be a proof of* $\vdash P, N$ *and* Q *a negative occurrence such that* $\mathsf{fv}(Q) \subseteq \{X\} \subseteq \mathcal{V}_N$. *We have:*

$$[\![\mathsf{F}_{\mathsf{Q}}(\Pi)]\!] = (\!|\mathbb{F}_{Q,[\![\Pi]\!]}|\!)$$

The proof of this lemma relies on a bisimulation between the two designs . It can be found in [BDS15].

Theorem 5.2. *If* Π' *is obtained from* Π *by* $\mu\mathsf{MALLP}$ *cut elimination rules, then* $[\![\Pi]\!] = [\![\Pi']\!]$.

Proof. The auxiliary cases together with the main cases for MALL are easy to treat and follow essentially from the associativity of design normalization. The only technical case is the one of Figure 5.2. Let us show how to treat one such reduction. We set that Π_1 to be the upper proof in Figure 5.2(the redex), and Π_2 the lower one (the reduct). We set also that:

$$d := [\![\Pi_L]\!], \qquad n := [\![\Theta]\!] \quad \text{and} \quad m := [\![\Pi_R]\!],$$

and:

$$x = x_{(\nu X.N)^\perp}, \qquad y = x_{M^\perp}.$$

Notice first that $\mathbb{G}_{P^\perp,n} = \mathbb{G}_{P,n}$. We have by definition that:

$$[\![\Pi_1]\!] = (\!|d[(\!|\mathbb{G}_{P,n}[m/x_0]|\!)/x]|\!)$$

By Lemma 5.2, we have that:

$$[\![\Pi_2]\!] = (\!|d[\mathbb{F}_{P,(\!|\mathbb{G}_{P,n}|\!)}[n[m/y]/x_0]/x]|\!)$$

But by definition of $\mathbb{G}_{P,n}$ we have that:

$$\mathbb{G}_{P,n}[y/x_0] = \mathbb{F}_{P,\mathbb{G}_{P,n}}[n/x_0]$$

Thus:

$$(\!|\mathbb{G}_{P,n}[y/x_0]|\!) = (\!|\mathbb{F}_{P,(\!|\mathbb{G}_{P,n}|\!)}[n/x_0]|\!)$$

The result follows from this remark and from associativity of designs normalization. □

5.4 On Completeness

Clearly, not all designs are interpretations of proofs, since some designs are not even computable. More generally, it is highly non-trivial whether (or when) one can recover coinvariants from a design in order to finitely express it as a proof. Indeed, coinvariants are completely hidden in the process of normalizing the interpretation of the (ν) rule. This is essentially the same difficulty that Girard encounters with second-order existential quantification in ludics [Gir01], and which lead him to give a completeness result for Π_1 formulas only. In our setting, that would correspond to handle least fixed points only, which would be rather weak. Fortunately, we can do better thanks to our direct treatment of fixed points in the semantics, by showing completeness with respect to class of designs called **essentially finite designs (EFD)**. These designs perform a finite computation followed by a copycat, we introduce them below.

5.4.1 Essentially finite designs

Definition 5.22. ***Essentially finite designs*** (EFD) are *inductively* defined by:

$$\begin{array}{rcl} p & ::= & (x \mid 1) \mid (x \mid \oplus_i\langle n\rangle) \mid (x \mid \otimes\langle n_1, n_2\rangle) \mid (x \mid \downarrow\langle n\rangle) \\ n & ::= & \bot.p_0 \mid \&_1(x_1).p_1 + \&_2(x_2).p_2 \mid \parr(x_1, x_2).p \mid \uparrow(x_1).p_1 \mid \eta_F \mid \Omega^- \end{array}$$

with $x_1 \in fv(p_1)$, $x_2 \in fv(p_2)$ and $x_1, x_2 \in fv(p)$.

Even though they are inductively defined, EFDs can be infinite, due to the presence of the copycat designs. The main theorem of this section is the following:

Theorem 5.3 (Completeness for EFD)**.** *Let d be an EFD and Γ be a sequent. If $d \models [\![\Gamma]\!]$ there is a μMALLP proof π of conclusion Γ such that $d = [\![\pi]\!]$.*

Proof. The proof of this theorem is by induction on the structure of the EFD, using internal completeness. We treat only the case where d is the design to give an example. Suppose that $d \models [\![\Gamma]\!]$. Since d is negative, Γ is also negative, that is $\Gamma = \Delta, N$. It is easy to check that N has necessarily the form $P_1 \,\&\, P_2$. Using internal completeness and the closure principle, $p_i \models [\![\Delta, P_i]\!]$, for $i = 1, 2$. By induction hypothesis, there is a proof π_i of Δ, P_i for $i = 1, 2$. Let π be the proof of conclusion $\vdash \Delta, N$ ending by an application of the rule (&) on N, and

whose premises are π_1 and π_2. All the other cases are treated in the same way, except the one where d is an η-expansion: one needs to prove that if $\eta_F \models x_0 : Q, P^\perp$ then there is a proof π of $\vdash Q, P^\perp$ such that $\eta_F = [\![\pi]\!]$. We dedicate the rest of the section to the proof of this result, by reducing it to a new statement. □

Definition 5.23. Let F and G be two closed formulas. We define the ***relation*** $\approx$ between occurrences inductively as follows: $F \approx G$ if and only if

$$F = \sigma X.c(F_1, \dots, F_n) \text{ and } G = \sigma Y.c(G_1, \dots, G_n)$$

Moreover, if we set $F \to c(H_1, \dots, H_n)$ and $G \to c(I_1, \dots, I_n)$ then $H_i \approx I_i$ for every $1 \leq i \leq n$.

In other words, $F \approx G$ if and only if F and G have the same infinite unfolding. The following proposition is easy to show.

Proposition 5.8. *Let P, Q and F be three occurrences: then the following holds:*

- *If $\eta_F \models x_0 : Q, P^\perp$ then $P \approx Q \approx F$.*
- *If $\eta_F \models x_0 : Q, P^\perp$ then $\eta_F = \eta_P = \eta_Q$.*
- *$\eta_F \models x_0 : Q, P^\perp$ if and only if $[\![P]\!] \subseteq [\![Q]\!]$.*

Proof. The first point is obtained by applying recursively the closure principle and internal completeness. The second points is a consequence of the first one, it simply follows from the remark that the η-expansion of a formula correspond to its infinite unfolding. Since F, P and Q share the same unfolding by the first point, they have also the same η-expansion. The third point is a direct consequence of the closure principle and the fact that $(\!|\eta_F[d/x_0]|\!) = (\!|d|\!)$ for every $d \in [\![G]\!]$ for every $G \approx P$, which can be verified easily. □

As a consequence, we only need to prove the following theorem, to obtain the last inductive case of the proof of Theorem 5.3.

Theorem 5.4 (Completeness for semantic inclusion)**.** *If P, Q are two positive occurrences such that $[\![P]\!] \subseteq [\![Q]\!]$, then there is a proof π of $\vdash Q, P^\perp$ satisfying $[\![\pi]\!] = \eta_P$.*

The remainder of this section is dedicated to the proof of this result.

5.4.2 Semantics inclusions in μMALLP

In order to show the provability of semantic inclusions in μMALLP, we shall use as an intermediate proof system: the circular proof system $\mu\mathsf{MALLP}^\omega$. This proof system is obtained by adding the unfolding rules for fixed points, allowing derivations having the shape graphs and equipping them with a validity condition (See Section 2.4.3). The proof of Theorem 5.4 is made in three steps:

i) We show first that semantic inclusions are provable in $\mu\mathsf{MALL}^\omega$ using translatable proofs. In other words we show that if $[\![P]\!] \subseteq [\![Q]\!]$, then there is a $\mu\mathsf{MALL}^\omega$ proof Π of $\vdash Q, P^\perp$ satisfying the translatability criterion (See Section 2.4.5, Definition 2.39).

ii) We show that the proof systems μMALLP and μMALLP$^\omega$ have the invariant property (Definition 2.40). By i) and ii), we are in the domain of applicability of the translation procedure (Theorem 2.5), that is to say, we can translate Π into a μMALLP proof π using the procedure described in the proof of Theorem 2.5.

iii) We finally show that π has the same interpretation as η_P, that is $[\![\pi]\!] = \eta_P$. This concludes the proof of Theorem 5.4.

Semantic inclusions in μMALLP$^\omega$

We show in this section that semantic inclusions are provable in μMALLP$^\omega$. For that, we need a few technical definitions, which we introduce in the following.

Definition 5.24. Let F, H be two occurrences. Let X_0 be a variable of the same polarity as H, not occurring in F nor H.

We define $\mathcal{O}_H^{X_0}(F)$ as the unique occurrence such that:

$$\mathcal{O}_H^{X_0}(F)[H/X_0] = F \text{ and } H \not\leq \mathcal{O}_H^{X_0}(F)$$

We shall simply write $\mathcal{O}_H(F)$ instead of $\mathcal{O}_H^{X_0}(F)$ when there is no ambiguity.

This operator commutes with μMALL connectives. Further, it commutes with unfolding under some subformula condition, as shown by the following proposition.

Lemma 5.3. *Let F, H be two formulas such that $H < F$. For every* MALL *connective s and $\sigma \in \{\mu, \nu\}$, one has:*

- *If $F = s(F_1, \ldots, F_n)$ then $\mathcal{O}_H(s(F_1, \ldots, F_n)) = s(\mathcal{O}_H(F_1), \ldots, \mathcal{O}_H(F_n))$.*
- *If $F = \sigma Y.G$ then $\mathcal{O}_H(\sigma Y.G) = \sigma Y.\mathcal{O}_H(G)$.*
- *If $F = \sigma Y.G$ then $\mathcal{O}_H(G[(\sigma Y.G)/Y]) = \mathcal{O}_H(G)[\mathcal{O}_H(\sigma Y.G)/Y]$.*

Proof. Let F, H be two formulas such that $H < F$.

- If $F = s(F_1, \ldots, F_n)$ then $E := s(\mathcal{O}_H(F_1), \ldots, \mathcal{O}_H(F_n))$ verifies that $E[H/X_0] = F$. We have that $H \not\leq E$, otherwise we would have either $H = E$ and then:

  $$H = H[H/X_0] = s(\mathcal{O}_H(F_1)[H/X_0], \ldots, \mathcal{O}_H(F_n)[H/X_0]) = F$$

  which is not possible, otherwise $H \leq \mathcal{O}_H(F_i)$ for some i which is not possible neither by definition.

- If $F = \sigma Y.G$, we apply the same argument to $E := \sigma Y.\mathcal{O}_H(G)$.

- We check that the right-hand side $E := \mathcal{O}_H(G)[\mathcal{O}_H(\sigma Y.G)/Y]$ verifies the two conditions of Definition 5.24. The first one is obvious: $E[H/X_0] = G[\sigma Y.G/Y]$. It remains to check that $H \not\leq E$. Since we obviously have $H \not\leq \mathcal{O}_H(G)$ and $H \not\leq \mathcal{O}_H(\sigma Y.G)$, it only remains to consider the case where $\mathcal{O}_H(\sigma Y.G) \leq H$. But, since X_0 does not occur free in H, this would mean that X_0 is not free in $\mathcal{O}_H(\sigma Y.G)$, which is equivalent to $H \not\leq \sigma Y.G$, contradicting our hypothesis.

□

Now we state and show our result of semantic inclusions in$\mu\mathsf{MALLP}^\omega$:

Proposition 5.9. *If* $[\![P]\!] \subseteq [\![Q]\!]$ *then there is a translatable* $\mu\mathsf{MALLP}^\omega$ *proof* Π *of* $\vdash P^\perp, Q$.

Proof. By Proposition 5.8, we have that $P \approx Q$. Thus

$$P = \mu X.c(I_1, \ldots, I_n) \text{ and } Q = \mu Y.c(J_1, \ldots, J_n)$$

And if we set $P \to c(M_1, \ldots, M_n)$ and $G \to c(N_1, \ldots, N_n)$ then $M_i \approx N_i$ for every $1 \leq i \leq n$. Let $\Theta(P^\perp, Q)$ be the $\mu\mathsf{MALLP}^\infty$ pre-proof coinductively as follows:

- If $c = \otimes$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Theta(N_1, M_1^\perp)}{\vdash M_1^\perp, N_1} \qquad \dfrac{\Theta(N_2, M_2^\perp)}{\vdash M_2^\perp, N_2}}{\vdash M_1^\perp, M_2^\perp, N_1 \otimes N_2}\,(\otimes)}{\vdash M_1^\perp, M_2^\perp, Q}\,(\mu)}{\vdash M_1^\perp \parr M_2^\perp, Q}\,(\parr)}{\vdash P^\perp, Q}\,(\nu)$$

- If $c = \oplus$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\Theta(N_1, M_1^\perp)}{\vdash M_1^\perp, N_1}\,(\oplus_1)}{\vdash M_1^\perp, N_1 \oplus N_2}\,(\mu) \qquad \dfrac{\dfrac{\Theta(N_2, M_2^\perp)}{\vdash M_2^\perp, N_2}\,(\oplus_2)}{\vdash M_1^\perp, N_1 \oplus N_2}\,(\mu)}{\vdash M_1^\perp, Q \qquad \vdash M_2^\perp, Q}\,(\parr)}{\vdash M_1^\perp \,\&\, M_2^\perp, Q}\,(\nu)}{\vdash P^\perp, Q}$$

- If $c = \downarrow$:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Theta(N_1, M_1^\perp)}{\vdash M_1^\perp, N_1}\,(\downarrow)}{\vdash M_1^\perp, \downarrow N_1}\,(\mu)}{\vdash M_1^\perp, Q}\,(\uparrow)}{\vdash \uparrow M_1^\perp, Q}\,(\nu)}{\vdash P^\perp, Q}$$

In all these cases, the proof $\Theta(P^\perp, Q)$ has the following form:

$$\dfrac{\Theta(N_1, M_1^\perp) \quad \ldots \quad \Theta(N_n, M_n^\perp)}{\vdash P^\perp, Q}$$

Let us show that $\Theta(P^\perp, Q)$ satisfies the validity condition. We proceed by contradiction, and suppose that $\Theta(P^\perp, Q)$ has an infinite branch $\gamma = (\gamma_k)_{0 \leq k}$ which is not valid. Note that the

branch γ has exactly two infinite threads, one starting from $P^\perp$ we denote it by $t = (F_i^\perp)_{i\in\omega}$ and the other from Q we denote it by $t' = (G_i)_{i\in\omega}$. Let $L^\perp = \mu X.K^\perp = \mathsf{min}(\mathsf{Inf}(\bar{t}))$. To get a contradiction we will construct a design d such that $d \in [\![P]\!]$ and $d \notin [\![Q]\!]$.

We set $\gamma_k = (\vdash F_k^\perp, G_k)$ and we associate to every γ_k a design d_k coinductively defined by the following equations:

- If $F_k^\perp \to^\star F_{k+1}^\perp \otimes E^\perp$ and $G_k \to^\star G_{k+1} \parr H$ then $d_k = \parr(x_1, x_2).d_{k+1}[x_1/x_0]$.
- If $F_k^\perp \to^\star E^\perp \otimes F_{k+1}^\perp$ and $G_k \to^\star H \parr G_{k+1}$ then $d_k = \parr(x_1, x_2).d_{k+1}[x_2/x_0]$.
- If $F_k^\perp \to^\star F_{k+1}^\perp \,\&\, E^\perp$ and $G_k \to^\star G_{k+1} \oplus H$ then $d_k = x_0 \mid \oplus_1\langle d_{k+1}\rangle$.
- If $F_k^\perp \to^\star E^\perp \,\&\, F_{k+1}^\perp$ and $G_k \to^\star H \oplus G_{k+1}$ then $d_k = x_0 \mid \oplus_2\langle d_{k+1}\rangle$.
- We proceed similarly for $\otimes$, & and shifts.

For every k, the design d_k "follows" the branch starting from γ_k. In particular, d_0 can be seen as a design representing the branch γ. In the following, we will show that:

(**P1**) $d_0 \in [\![P]\!]$

(**P2**) $d_0 \notin [\![Q]\!]$

Let us show (P1). Since $P = F_0$, (**P1**) amounts to show that $d_0 \in [\![F_0]\!]$. Let I be the following set of indices:

$$I = \{m \mid F_m \equiv L\}$$

First remark that by applying internal completeness iteratively, we have that:

$$\forall k \in \omega \qquad d_0 \in [\![F_0]\!] \Leftrightarrow d_1 \in [\![F_1]\!] \Leftrightarrow \cdots \Leftrightarrow d_k \in [\![F_k]\!]$$

In particular for every $i \in I$, we have that $d_0 \in [\![P]\!]$ if and only if $d_i \in [\![F_i]\!] = [\![L]\!]$. But $[\![L]\!] = \mathsf{gfp}(\varphi)$ where $\varphi : \mathbf{C} \mapsto [\![K]\!]^{X\mapsto\mathbf{C}}$. Hence, to prove that $d_i \in [\![L]\!]$ for $i \in I$ it suffices to find a post-fixed point $\mathbf{A}$ of φ such that $d_i \in \mathbf{A}$. We shall establish this for $\mathbf{A} = \{\ d_m \mid m \in I\ \}^{\perp\perp}$. This follows from the two following facts that we shall prove next:

1) If n, m are two consecutive indices in I, then for every $i \in]n, m]$ we have that:

$$d_i \ \in\ [\![\mathcal{O}_L(F_i)]\!]^{X\mapsto\mathbf{A}}$$

2) If $n \in I$, then $d_{n+1} \in [\![\mathcal{O}_L(F_{n+1})]\!]^{X\mapsto\mathbf{A}} \Rightarrow d_i \in [\![K]\!]^{X\mapsto\mathbf{A}}$.

To show the result 1) we proceed by a decreasing induction on i.

- **Base case.** If $i = m$, we have that $F_i = L$ thus $\mathcal{O}_L(F_i) = X_0$, from which $d_i \in [\![X_0]\!]^{X_0\mapsto\mathbf{A}}$ immediately follows.

- **Inductive case.** Let us treat the case where $F_i^\perp \to^\star F_{i+1}^\perp \otimes E^\perp$, the other ones being similar. For simplicity of the presentation, we assume that $F_i = \nu X.I \parr I'$, thus $F_{i+1} = I[F_i/X]$ and $E = I'[F_i/X]$. By construction, we have that $d_i = \parr(x_1, x_2).d_{i+1}[x_1/x_0]$. By induction hypothesis, we have that $d_{i+1} \in [\![\mathcal{O}_L(F_{i+1})]\!]^{X_0 \mapsto \mathbf{A}}$. Therefore, for every formula H such that $\mathsf{fv}(H) \subseteq \{X_0\}$, we ave that:

$$\parr(x_0, x_1).d_{i+1} \in [\![\mathcal{O}_L(F_{i+1}) \parr H]\!]^{X_0 \mapsto \mathbf{A}}$$

In particular we set $H = \mathcal{O}_L(E)$. Which gives us that:

$$\begin{array}{rcl} d_i & \in & [\![\mathcal{O}_L(F_{m+1}) \parr \mathcal{O}_L(E)]\!]^{X \mapsto \mathbf{A}} \\ & \overset{\star}{=} & [\![\mathcal{O}_L(F_{m+1} \parr E)]\!]^{X \mapsto \mathbf{A}} \\ & = & [\![\mathcal{O}_L((I \parr I')[F_i/X])]\!]^{X \mapsto \mathbf{A}} \\ & \overset{\star\star}{=} & [\![\mathcal{O}_L(I \parr I')[\mathcal{O}_L(F_i)/X]]\!]^{X \mapsto \mathbf{A}} \\ & \overset{\dagger}{=} & [\![\mathcal{O}_L(I \parr I')[\nu X.\mathcal{O}_L(I \parr I')/X]]\!]^{X \mapsto \mathbf{A}} \\ & \overset{\ddagger}{=} & [\![\nu X.\mathcal{O}_L(I \parr I')]\!]^{X \mapsto \mathbf{A}} \\ & \overset{\dagger}{=} & [\![F_i]\!]^{X \mapsto \mathbf{A}} \end{array}$$

($\star$) First item of Lemma 5.3.
($\star\star$) Third item of Lemma 5.3.
($\dagger$) Second item of Lemma 5.3.
($\ddagger$) The interpretation of a fixed point is equal to the interpretation of its unfolding.

The result 2) is shown in a similar way as to the inductive case of 1., using Lemma 5.3. Now that we have shown that $d \in [\![P]\!]$, we show that $d \notin [\![Q]\!]$.

Let us show P2. We show now that $d_0 \notin [\![Q]\!]$. To do so, we construct a design d_0' such that $d_0' \in [\![G]\!]^\perp$ and $d_0 \not\perp d_0'$. Let us consider the branch γ' of $\Theta(Q, P^\perp)$ that follows the same occurrences as γ. We can thus construct as before a design d_0' such that $d_0' \in [\![Q^\perp]\!]$. It is easy to verify that the interaction between d_0 and d_0' diverges and therefore that $d_0 \not\perp d_0'$.

The proof $\Theta(P^\perp, Q)$ being regular and having exactly two threads that do not "communicate", we can represent it by a translatable circular proof Π. □

From μMALLP$^\omega$ to μMALLP

To make the translation procedure from Section 2.4.5 work, a condition on the origin and the target proof systems, called the invariant property (Definition 2.40), should be satisfied. We show in the following that μMALLP and μMALLP$^\omega$ satisfy the invariant property.

Proposition 5.10. *The proof systems μMALLP and μMALLP$^\omega$ satisfy the invariant property.*

Proof. Let Δ be a positive sequent. We define $\mathsf{P}(\Delta)$ as follows:

- If $\Delta = \{P\}$ then $\mathsf{P}(\Delta) = P$.
- If $\Delta = \{P\} \cup \Delta'$ then $\downarrow(P \parr \mathsf{P}(\Delta'))$

$\mathsf{P}(\Delta)$ is the $\parr$ of Δ where we inserted some $\downarrow$ to adjust the polarities.
Let $\nu X.F$ be an occurrence and Δ be a set of positive occurrences. We choose the invariant of $(\nu X.F, \Delta)$ to be $\mathcal{I} = \nu X.\downarrow(F \oplus (\mathsf{P}(\Delta))^{\perp})$ if Δ is non-empty and $\mathcal{I} = \nu X.F$ otherwise. We derive the rules (Subst), (Unfold), (Close) and (Replace) in μMALLP and the rules (Subst$^\omega$) and (Unfold$^\omega$) in μMALLP$^\omega$ exactly as we did in μMALL and μMALL$^\omega$ in the proof of Proposition 2.15, by adding the following minor modifications:

- We insert the rules $\downarrow$ and $\uparrow$ coming from $\mathsf{P}(\Delta)$ and $\mathcal{I}$.
- We use the polarized functoriality introduced in Definition 5.4.

□

As a consequence, we can apply the translation procedure to the proof Π obtained by Proposition 5.9 to get a μMALLP proof π of $\vdash P^{\perp}, Q$.

The interpretation of π is η_P.

Proposition 5.11. *Let π be the proof obtained by applying the translation procedure to the proof Π obtained by Proposition 5.9. The interpretation of π is η.*

The proof of this result is given in [BDS15] and relies on a bisimulation between π and Π. Actually, we hope to prove a more general result, saying that the interpretation of a μMALLP$^\omega$ proof and its translation have the same interpretation. In other words, our translation result preserves the computational behaviour of proofs. To state this conjecture, we need to define the interpretation of a μMALLP$^\omega$ proof.

Definition 5.25. The ***interpretation*** $[\![\Pi]\!]$ of a μMALLP$^\infty$ proof Π is obtained by reading coinductively the rules of Figure 5.3, and by replacing the interpretation of the rule (ν) by the following one:

$$\frac{n \vdash \Gamma, N[\mu X.N/X]}{n \vdash \Gamma, \nu X.N}\ (\nu)$$

The interpretation of a μMALLP$^\omega$ proof is the interpretation of its unfolding.

Conjecture 5.1. *Let Π be a translatable μMALLP$^\omega$ proof and let π be the μMALLP proof obtained by the translation procedure of Section 2.4.5. We have that $[\![\Pi]\!] = [\![\pi]\!]$.*

Part II

Constructive completeness for the linear-time μ-calculus

Chapter 6

The completeness problem for the linear-time μ-calculus

Verification and the μ-calculus. The linear-time μ-calculus [Koz83] is a temporal logic that extends Pnueli's *Linear Temporal Logic* (LTL) [BKP] with least and greatest fixed points. This increases considerably its expressive power while keeping the decidability properties of LTL, which makes it a very suitable logic for verification. The linear-time μ-calculus has infinite words as models, thus it can be used to express trace properties of reactive systems.

There exist, among others, two approaches to verification using temporal logics [Wal94]. The first one, called "model-theoretic", describes both the system S and the property P to check as formulas φ_S and φ_P; then verifying whether S satisfies P is reduced to checking the validity of the formula $\varphi_S \to \varphi_P$. The other approach, called "proof-theoretic" reduces the verification problem to the provability of the formula $\varphi_S \to \varphi_P$.

The advantage of the second approach is that it gives, besides the boolean answer to the verification problem, a *certificate* that supports the decision of the verification tool, which is the proof of the formula $\varphi_S \to \varphi_P$.

The completeness problem and constructiveness. To make the proof-theoretic approach to verification work with the linear-time μ-calculus, two conditions should be satisfied: the first is the existence of a sound and complete deductive system for the linear-time μ-calculus; the second is the existence of algorithms that produce proofs for valid formulas. The first condition is satisfied, since the linear-time μ-calculus enjoys a deductive system, that we call $\mu\mathsf{LK}\odot$, which is the restriction of Kozen's axiomatization [Koz83] for the modal μ-calculus to the linear time. This system was proved to be sound and complete by Kaivola [Kai95]. But the second condition is not really met, since the only existing algorithm is the naive one, that enumerates all $\mu\mathsf{LK}\odot$ proofs.

A proof of completeness is a mathematical argument showing that every valid formula is provable, but it is not always possible to extract from such an argument an algorithm that produces proofs for valid formulas. Indeed, completeness proofs may involve complex, non-constructive arguments yielding no method for actually constructing a proof. On the contrary, a constructive proof of completeness, specifying a proof search method, would readily provides a realistic algorithm.

Analyzing earlier proofs. None of the existing proofs of completeness for the μ-calculus with respect to Kozen's axiomatization ([Kai95], [Wal95]) is constructive in this sense. Walukiewicz showed in [Wal93] that the branching-time μ-calculus is complete *w.r.t.* an axiomatization tha he introduced there, which is different from Kozen's, and his argument exhibits proofs for valid formulas. However, this axiomatization is much less natural than Kozen's, the later being, to use his terms, "as natural as the notion of Kripke structures".

In this part of the thesis, we provide a constructive proof for the full linear-time μ-calculus with respect to $\mu\mathsf{LK}\odot$. To do so, we go back to the earlier proofs of completeness, and try to understand where constructiveness is lost, to better solve this problem.

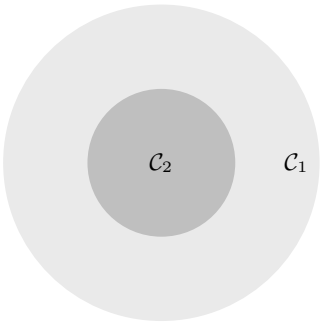
Existing proofs of completeness for the μ-calculus rely schematically on the following idea. Find a subset $\mathcal{C}_2$ of the set of μ-calculus formulas $\mathcal{C}_1$ such that:

1) For every valid formula φ_1 in $\mathcal{C}_1$, there is a valid formula φ_2 in $\mathcal{C}_2$ such that $\varphi_2 \vdash_{\mu\mathsf{LK}\odot} \varphi_1$.

2) Every valid formula of $\mathcal{C}_2$ is provable. This is the completeness result restricted to $\mathcal{C}_2$.

Completeness is proved by combining 1) and 2) via a cut rule:

$$\dfrac{\dfrac{2)}{\vdash \varphi_2} \qquad \dfrac{1)}{\varphi_2 \vdash \varphi_1}}{\vdash \varphi_1}\ (\mathsf{Cut})$$



The complexity of problems 1) and 2) depends on the class $\mathcal{C}_2$: the larger it is, the more difficult problem 2) becomes, since it gets close to the original completeness problem. On the contrary, when $\mathcal{C}_2$ gets smaller, problem 1) becomes difficult. Kaivola's proof uses the class of *banan form formulas*, and Walukiewicz' one uses the class of *negatitions of disjunctive formulas*. These classes are very small and problem 2) is easy to prove, but problem 1) is much more involved, and this is where constructiveness is lost in both proofs.

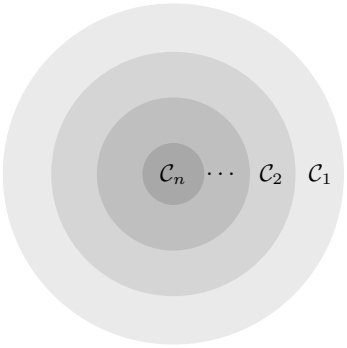
Our solution. Instead of splitting the difficulty in two by introducing one intermediate class, we introduce several classes $\mathcal{C}_n \subseteq \cdots \subseteq \mathcal{C}_1$ and generalize the proof scheme used earlier:

1) For all $i \in [1, n[$ and for every valid formula $\varphi_i \in \mathcal{C}_i$, there is a valid formula $\varphi_{i+1} \in \mathcal{C}_{i+1}$ such that $\varphi_{i+1} \vdash_{\mu\mathsf{LK}\odot} \varphi_i$.

2) Every valid formula of $\mathcal{C}_n$ is provable.

As before, we combine these results to get completeness. The interest of this approach is to split the difficult problem of completeness into several easier problems, for which we can hope to construct effectively a proof.

$$\frac{\vdash \varphi_n \quad \varphi_n \vdash \varphi_{n-1} \quad \dots \quad \varphi_2 \vdash \varphi_1}{\vdash \varphi_1}\ (\mathsf{Cut})$$



Now the question is how to find these classes. For that, we identified three sources of complexity that make a valid formula hard to prove:

i) The alternation of disjunctions and conjunctions.

ii) The interleaving of least and greatest fixed points.

iii) The presence of disjunctions.

In automata theory, these sources of complexity also exist with different names:

i) Alternation (of universal and existential non-determinism).

ii) The use of parity conditions.

iii) Non-determinism.

In automata over infinite words, all these difficulties can be reduced through effective algorithms, transforming automata with one of these features into others without. For example, one has algorithms to eliminate alternation, to reduce the number of priorities for a parity condition or to get rid of non-determinism. The correspondence between linear-time μ-calculus formulas and alternating parity word automata (APW) is now very well established. This is fortunate since our idea is to import these techniques from the automata side to the logical one. Concretely, it is known that we can encode every APW $\mathcal{A}$ by a formula $[\mathcal{A}]$ such that the language of $\mathcal{A}$ equals the set of models of $[\mathcal{A}]$. The intermediate classes we will use are the following: The largest class, denoted [APW], is the image of APW by this encoding; this class embodies all the difficulties indicated above. The next class is [NPW], the image of non-deterministic parity automata (NPW) by this encoding. The formulas of this class do not contain the first level of complexity which is the alternation $\vee, \wedge$. The third class is [NBW], the encoding of non-deterministic Büchi automata (NBW). Büchi automata are particular cases of parity automata where only the two priorities 0 and 1 are allowed. We can say then that in this class we simplified the two difficulties i) and ii). The smallest class is [DBW], the image of deterministic Büchi automata, where the three difficulties are eliminated. The proof will be carried out in the following 5 steps:

I. $\forall \varphi \in \mathcal{C}_1, \exists \mathcal{A} \in$ APW such that $\mathcal{L}(\mathcal{A}) = \mathcal{M}(\varphi)$ and $[\mathcal{A}] \vdash_{\mu\mathsf{LK}_\odot} \varphi$.

II. $\forall \mathcal{A} \in$ APW, $\exists \mathcal{P} \in$ NPW such that $\mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{A})$ and $[\mathcal{P}] \vdash_{\mu\mathsf{LK}_\odot} [\mathcal{A}]$.

III. $\forall \mathcal{P} \in$ NPW, $\exists \mathcal{B} \in$ NBW such that $\mathcal{L}(\mathcal{B}) = \mathcal{L}(\mathcal{P})$ and $[\mathcal{B}] \vdash_{\mu\mathsf{LK}_\odot} [\mathcal{P}]$.

IV. $\forall \mathcal{B} \in \text{NBW}$, if $\mathcal{L}(\mathcal{B}) = \Sigma^\omega$ then $\exists \mathcal{D} \in \text{DBW}\ \mathcal{L}(\mathcal{D}) = \Sigma^\omega$ and $[\mathcal{D}] \vdash_{\mu\mathsf{LK}\odot} [\mathcal{B}]$.

V. $\forall \mathcal{D} \in \text{DBW}$, if $\mathcal{L}(\mathcal{D}) = \Sigma^\omega$ then $\vdash_{\mu\mathsf{LK}\odot} [\mathcal{D}]$.

Step IV is a bit special because NBW cannot in general be determinized into DBW. But if a NBW $\mathcal{B}$ recognizes the universal language Σ^ω, there is obviously a DBW $\mathcal{D}$ with the same language: for instance the complete Büchi automaton with exactly one (accepting) state. This is enough for our needs, since we start in the proof of completeness from a valid formula φ (*i.e.*, $\mathcal{M}(\varphi) = \Sigma^\omega$), hence the automata $\mathcal{A}, \mathcal{P}$ and $\mathcal{B}$ constructed in steps I-III all recognize the language Σ^ω. To show that $[\mathcal{D}] \vdash_{\mu\mathsf{LK}\odot} [\mathcal{B}]$ in step IV, we use a more general result from:

IV′. $\forall \mathcal{B}_1, \mathcal{B}_2 \in \text{NBW}$, if $\mathcal{L}(\mathcal{B}_2) \subseteq \mathcal{L}(\mathcal{B}_1)$ then $[\mathcal{B}_2] \vdash_{\mu\mathsf{LK}\odot} [\mathcal{B}_1]$.

The proof of this result will rely on determinization of NBW into deterministic parity automata, but contrarily to steps I-III where automata transformations are used to build an automaton, determinization is used in this step as a proof-search algorithm.

We now give a general idea of how to prove the sequents of these steps. Actually, what makes the proof search difficult in $\mu\mathsf{LK}\odot$, is Park's rule (shown below) where S should be guessed.

$$\frac{\Gamma \vdash \Delta, S \qquad S \vdash F[S/X]}{\Gamma \vdash \Delta, \nu X.F}\ (\nu)$$

To circumvent this problem, we go through an intermediate proof system where the rule (ν) just unfolds the ν-formula:

$$\frac{\Gamma \vdash \Delta, F[\nu X.F/X]}{\Gamma \vdash \Delta, \nu X.F}\ (\nu)$$

Two examples of such proof systems are the one introduced in [DHL06] which we call $\mu\mathsf{LK}^\omega_{\mathsf{DHL}}$, and the one introduced in Chapter 2, Section 2.4.3, called $\mu\mathsf{LK}\odot^\omega$. The idea is to first find a proof for the sequent to prove in the intermediate system, then to transform this proof into a $\mu\mathsf{LK}\odot$ one.

The advantage of $\mu\mathsf{LK}^\omega_{\mathsf{DHL}}$ is that it is completely invertible and the proof search is a trivial task. However, no algorithm is known to transform effectively $\mu\mathsf{LK}^\omega_{\mathsf{DHL}}$ proofs into $\mu\mathsf{LK}\odot$ ones.

In contrast, we have given in Chapter 2, Section 2.4.5 a strong translation result for $\mu\mathsf{LK}\odot^\omega$, based on a general geometric condition on proofs. Building on this, we shall work with $\mu\mathsf{LK}\odot^\omega$. To get this stronger translatability criterion, $\mu\mathsf{LK}\odot^\omega$ uses sequents of a particular shape. Indeed, sequents are not sets of formulas, as it is the case for $\mu\mathsf{LK}^\omega_{\mathsf{DHL}}$, but are rather sets of *occurrences*. The difficulty of using such sequents is that the proof system is not invertible and proving the sequents of steps I-V in $\mu\mathsf{LK}\odot^\omega$ is not immediate.

Let us finally emphasize that the implications appearing in steps I-V are well known at the semantical level, but lifting them to the provability level is not immediate and strongly depends on the encoding $[\_]$ and the shape of automata obtained by the different automata transformations. To illustrate this by an extrem example, any valid formula φ is semantically equivalent to $\top$ and to itself, but proving $\top \vdash \varphi$ is as difficult as proving $\vdash \varphi$, while proving $\varphi \vdash \varphi$ is immediate. In general, given a formula ψ semantically equivalent to φ, the closer ψ

is to φ, the easier $\psi \vdash \varphi$ will be to prove. That is why we will provide for our development an encoding of automata that follows closely their structure, and automata transformations that do not change brutally the input automaton (or the input formula for step I). That is also why we cannot treat these transformations as black boxes and will recall them in detail.

Organization of Part II. This Part is organized as follows. In Chapter 7 we introduce the linear-time μ-calculus and its semantics, and recall the proof systems $\mu\mathsf{LK}\odot$ and $\mu\mathsf{LK}\odot^{\omega}$. In Chapter 8, we present the different models of automata (alternating, non-determinsitc, deterministic) over words that we shall work with, equipped with various acceptance conditions (parity, Büchi, etc). Then we discuss the links between these different kinds of automata. We present in Chapter 9 the encoding $[\_]$ of APW automata into linear-time μ-calculus formulas. Conversely, we give a way to build for every μ-calculus formula φ an APW $\mathcal{A}_\varphi$ that recognizes the set of its models. The main result of this chapter is that $[\mathcal{A}_\varphi] \vdash_{\mu\mathsf{LK}\odot} \varphi$. In Chapter 10, we recall the automata transformations that turn an APW $\mathcal{A}$ into an NPW $\mathcal{P}$, and $\mathcal{P}$ into an NBW $\mathcal{B}$, all having the same language. The main results of this chapter are $[\mathcal{B}] \vdash_{\mu\mathsf{LK}\odot} [\mathcal{P}]$ and $[\mathcal{P}] \vdash_{\mu\mathsf{LK}\odot} [\mathcal{A}]$. In Chapter 11, we show that Büchi inclusions can be reflected in the logic $\mu\mathsf{LK}\odot$, that is, for every Büchi automata $\mathcal{B}_1$ and $\mathcal{B}_2$, we can construct a $\mu\mathsf{LK}\odot$ proof of $[\mathcal{B}_1] \vdash_{\mu\mathsf{LK}\odot} [\mathcal{B}_2]$. Using this last result, we show in Chapter 12 that for every NBW $\mathcal{B}$ recognizing the language Σ^ω, there is a DBW $\mathcal{D}$ recognizing also Σ^ω, such that $[\mathcal{D}] \vdash_{\mu\mathsf{LK}\odot} [\mathcal{B}]$ and $\vdash_{\mu\mathsf{LK}\odot} [\mathcal{D}]$. We finally bring these pieces together to get a constructive proof of completeness.

Chapter 7

The linear-time μ-calculus

Linear-time temporal logics were introduced to express trace properties of reactive programs. This paradigm, first proposed by Pnueli in [Pnu77], is based on viewing an execution of a reactive program as a sequence of states, then describing each individual state by some means, specific to the temporal logic under consideration. For instance, in a propositional logic such as the linear-time μ-calculus, each state is described by a set of propositional atoms. This abstraction of an execution sequence into a sequence of sets of propositional atoms yields a *model*. Models are structures on which linear temporal formulas can be interpreted, thus, these formulas can be used to specify properties of the executions of a program. We have seen in Chapter 1 an example of such a propositional linear-time logic: the logic LTL (Definition 1.17). Although commonly used in practice, the logic LTL cannot capture some useful properties such as *at every even moment* φ. The linear-time μ-calculus was introduced by Barringer, Kuiper and Pnueli in [BKP84, BKP] to make up for this lack of expressivity. Contrarily to LTL, the linear-time μ-calculus contains only but one modal operator *next*, but the addition of fixed point operators ν and μ increases considerably its expressive power, allowing in particular to encode all other temporal operators of LTL.

Another way to model the execution of a program is to use a tree of states. A branching of such a tree represents the points where non-deterministic choices between various courses of execution are made. If we represent a state by a set of propositional atoms as before, we get structures which are the models of *branching-time temporal logic* formulas. While linear-time formulas describe one execution at time, the branching-time formulas permit to reason about all executions simultaneously. The branching-time μ-calculus, introduced by Kozen in [Koz83] is the branching-time version of the aforementioned linear-time μ-calculus. The study of this logic is beyond the scope of this thesis, but we will sometimes import some tools and ideas used in the branching-time setting to the linear-time one (Section 9.1, Section 10.1.1).

Since the introduction of the linear and the branching-time μ-calculus, the problem of the existence of a complete deductive system has emerged. For the branching-time μ-calculus, the best-known axiomatization is the one introduced by Kozen [Koz83], which relies on Park's rules (Section 2.4.1). It was shown to be complete by Walukiewicz [Wal95] ten years later. For the linear-time μ-calculus, Kaivola specialized Kozen's axiomatization to the linear-time case and showed completeness in [Kai95]. We will be interested in this last deductive system in the rest of the part, but we will write it in the formalism of sequent calculus. We call the obtained proof system $\mu\mathsf{LK}\odot$.

More recently, Dax *et al.* introduced another deductive system for the linear-time μ-calculus, based on circular proofs [DHL06]. We will use later a similar system that we call $\mu\mathsf{LK}\odot^{\omega}$. Contrarily to Dax *et al.*, we will not use $\mu\mathsf{LK}\odot^{\omega}$ as an axiomatization, but rather as an intermediate proof system for the proof of completeness *w.r.t.* $\mu\mathsf{LK}\odot$.

This Chapter is devoted to the introduction of the linear-time μ-calculus and its proofs systems.

Organization of the chapter. In Section 7.1 we introduce the linear-time μ-calculus and its semantics. We recall in Section 7.2 the proof systems $\mu\mathsf{LK}\odot$ and $\mu\mathsf{LK}\odot^{\omega}$. Then we compare $\mu\mathsf{LK}\odot^{\omega}$ to the proof system of Dax *et al.*, emphasizing the difficulty of the proof search in the former compared to the latter. Finally, we show a sufficient condition that ensures the translatability of $\mu\mathsf{LK}\odot^{\omega}$ proofs into $\mu\mathsf{LK}\odot$ ones.

7.1 Syntax and semantics

We have seen in Chapter 2, Section 2.2, how to extend the syntax of a logic $\mathcal{L}$ with least and greatest fixed points, to get a logic $\mu\mathcal{L}$. The linear-time μ-calculus is obtained by such an extension, form the logic $\mathsf{LK}\odot$ (Definition 1.18), that is, the propositional classical logic with the modality *next* ($\odot$). We recall below the syntax of the linear-time μ-calculus formulas for clarity:

Definition 7.1. Let $\mathcal{V} = \{X, Y, \dots\}$ be a set of variables and $\mathcal{P} = \{\mathfrak{p}, \mathfrak{q}, \dots\}$ a set of atoms. The linear-time μ-calculus ***formulas*** $\varphi, \psi, \dots$, called here $\mathsf{LK}\odot$ formulas, are given by:

$$\varphi ::= \mathfrak{p} \mid \neg\mathfrak{p} \mid X \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \odot\varphi \mid \mu X.\varphi \mid \nu X.\varphi$$

The connectives μ and ν bind the variable X in φ. From there, bound variables, free variables and capture-avoiding substitution are defined in a standard way. The sub-formula ordering is denoted $\leq$ and $\mathsf{fv}(\bullet)$ denotes free variables. Atoms and their negations are called **literals**. We shall use σ to denote either μ or ν.

Note that we do not allow negations on variables. This is not a restriction since we are mostly interested in closed formulas. All the results presented here extend to the general case, where negations are allowed under a positivity condition on bound variables.

We do not consider the Boolean constants $\top, \bot$ as they can be encoded by $\top := \nu X.\odot X$ and $\bot := \mu X.\odot X$.

As for $\mathsf{LK}\odot$, the models of our formulas are the ω-words over the alphabet $\Sigma := 2^{\mathcal{P}}$. We extend the interpretation of $\mathsf{LK}\odot$ formulas (Definition 1.19) to handle least and greatest fixed points.

Definition 7.2. The ***semantics*** $\|\varphi\|^u_\rho$ of a formula φ *w.r.t.* a word $u \in \Sigma^\omega$ and a valuation $\rho : \mathcal{V} \mapsto 2^\omega$ is a subset of natural numbers inductively defined as follows:

$$\begin{array}{llll}
\|\mathfrak{p}\|^u_\rho = \{i \in \omega \mid \mathfrak{p} \in u_i\} & \|\neg\mathfrak{p}\|^u_\rho = \{i \in \omega \mid \mathfrak{p} \notin u_i\} \\
\|X\|^u_\rho = \rho(X) & \|\odot\varphi\|^u_\rho = \{i \in \omega \mid i+1 \in \|\varphi\|^u_\rho\} \\
\|\varphi \vee \psi\|^u_\rho = \|\varphi\|^u_\rho \cup \|\psi\|^u_\rho & \|\varphi \wedge \psi\|^u_\rho = \|\varphi\|^u_\rho \cap \|\psi\|^u_\rho
\end{array}$$

$$\|\nu X.\varphi\|^u_\rho = \bigcup\{\, W \subseteq \omega \mid W \subseteq \|\varphi\|^u_{\rho[X\leftarrow W]} \,\}$$
$$\|\mu X.\varphi\|^u_\rho = \bigcap\{\, W \subseteq \omega \mid \|\varphi\|^u_{\rho[X\leftarrow W]} \subseteq W \,\}$$

Suppose that φ is a closed formula. We write $\|\varphi\|^u$ instead of $\|\varphi\|^u_\rho$, since the semantics of φ does not depend on ρ. We say that φ is ***true*** in u, and we write $u \models \varphi$, if $0 \in \|\varphi\|^u$. The set of ***models of*** φ is defined by $\mathcal{M}(\varphi) = \{u \in \Sigma^\omega \mid u \models \varphi\}$. A formula is ***valid*** if it is true in every model, ie. $\mathcal{M}(\varphi) = \Sigma^\omega$.

Remark 7.1. The interpretation $\|\mu X.\varphi\|^u_\rho$ of the least fixed point formula $\mu X.\varphi$ can be seen as the least fixed point of the operator Φ over the complete lattice $(2^\omega, \subseteq)$ defined by:

$$\Phi := W \mapsto \|\varphi\|^u_{\rho::[X\to W]}$$

Indeed, one can prove by induction on the formula φ that Φ is monotonic, hence by Knaster-Tarski theorem, one has that $\mathsf{lfp}(\Phi)$ is the least pre-fixed point of Φ. Since the least element of a subset $S \subseteq 2^\omega$ in the lattice $(2^\omega, \subseteq)$ is $\bigcap S$, one has:

$$\|\mu X.\varphi\|^u_\rho = \mathsf{lfp}(\Phi)$$

Similarly, one can show that the interpretation of a ν-formula is the greatest fixed point of Φ:

$$\|\nu X.\varphi\|^u_\rho = \mathsf{gfp}(\Phi)$$

Remark 7.2. A way of understanding the meaning of the fixed point operators is viewing the least fixed points as finite iterations, and the greatest fixed points as infinite loopings. This intuition is made clear when we translate LTL operators into the μ-calculus. For example, the operators $\mathbf{F}$, $\mathbf{G}$ and $\mathbf{U}$ be encoded as follows:

$$\begin{array}{rcl} \mathbf{F}\varphi & = & \mu X.\varphi \vee \odot X \\ \mathbf{G}\varphi & = & \nu X.\varphi \wedge \odot X \\ \psi \mathbf{U} \varphi & = & \mu X.\varphi \vee (\psi \wedge \odot X) \end{array}$$

This understanding of fixed points as finite and infinite loopings is useful when we deal with simple examples. When it comes to understand the meaning of a formula with multiple interleavings of fixed points of different natures, the actual definition of the semantics is not very helpful. We present in Section 9.1 an alternative definition of the semantics of a formula, which is more operational and more enlightening.

7.2 Proof systems for the linear-time μ-calculus

Given a proof system S over a signature $\mathcal{L}$, we have shown in Chapter 2, Section 2.4, two ways to extend it, in order to get a proof system over the signature $\mu\mathcal{L}$. We apply that to the proof system $\mathsf{LK}\odot$ (Definition 1.20 or Figure 7.1).

The first way to extend $\mathsf{LK}\odot$ is to add Park's rules for the μ and ν connectives (Figure 7.2), inspired by the Knaster-Tarski characterization, yielding a finitary proof system denoted by $\mu\mathsf{LK}\odot$ (see Section 2.4.1). This proof system is the sequent calculus version of Kaivola's axiomatization [Kai95]. The second way to extend $\mathsf{LK}\odot$ with extremal fixed

$$\frac{F \equiv G}{F \vdash G}\ (\mathsf{Ax}) \qquad \frac{F \equiv G}{\vdash F^{\perp}, G}\ (\mathsf{Ax}) \qquad \frac{F \equiv G}{F^{\perp}, G \vdash}\ (\mathsf{Ax})$$

$$\frac{\Gamma, F \vdash \Delta \quad \Gamma \vdash F, \Delta}{\Gamma \vdash \Delta}\ (\mathsf{Cut}) \qquad \frac{\Gamma \vdash \Delta}{\Sigma, \odot\Gamma \vdash \Theta, \odot\Delta}\ (\odot)$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, F \vdash \Delta}(\mathsf{W_l}) \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash F, \Delta}(\mathsf{W_r}) \qquad \frac{\Gamma \vdash F, \Delta \quad \Gamma \vdash G, \Delta}{\Gamma \vdash F \wedge G, \Delta}\ (\wedge_{\mathsf{r}})$$

$$\frac{\Gamma, F \vdash \Delta \quad \Gamma, G \vdash \Delta}{\Gamma, F \vee G \vdash \Delta}\ (\vee_{\mathsf{l}}) \qquad \frac{\Gamma \vdash F, G, \Delta}{\Gamma \vdash F \vee G, \Delta}\ (\vee_{\mathsf{r}}) \qquad \frac{\Gamma, F, G \vdash \Delta}{\Gamma, F \wedge G \vdash \Delta}\ (\wedge_{\mathsf{l}})$$

Figure 7.1: Inference rules for $\mathsf{LK}\odot$.

$$\frac{F[S/X] \vdash S \quad S \vdash \Gamma}{\mu X.F \vdash \Gamma}\ (\mu_{\mathsf{l}}) \qquad \frac{\Gamma \vdash F[\mu X.F/X], \Delta}{\Gamma \vdash \mu X.F, \Delta}\ (\mu_{\mathsf{r}})$$

$$\frac{\Gamma, F[\nu X.F/X] \vdash \Delta}{\Gamma, \nu X.F \vdash \Delta}\ (\nu_{\mathsf{l}}) \qquad \frac{\Gamma \vdash S \quad S \vdash F[S/X]}{\Gamma \vdash \nu X.F}\ (\nu_{\mathsf{r}})$$

Figure 7.2: Fixed point rules for the $\mu\mathsf{LK}\odot$ proof system.

$$\frac{\Gamma, F[\sigma X.F/X] \vdash \Delta}{\Gamma, \sigma X.F \vdash \Delta}\ (\sigma_{\mathsf{l}}) \qquad \frac{\Gamma \vdash F[\sigma X.F/X], \Delta}{\Gamma \vdash \sigma X.F, \Delta}\ (\sigma_{\mathsf{r}})$$

Figure 7.3: Fixed point rules for the $\mu\mathsf{LK}\odot^{\infty}$ proof system.

points is to add unfolding rules for μ and ν (Figure 7.3) to $\mathsf{LK}\odot$, and to allow infinitary derivations (see Section 2.4.3). These infinitary derivations are called pre-proofs. To get a sound proof system, we have equipped these pre-proofs with a validity condition, requiring that in every branch, there is either a minimal ν-formula that gets unfolded infinitely often in the right hand-side of the branch, or a minimal μ-formula that gets unfolded infinitely often in the left-hand side of it. The obtained proof system is called $\mu\mathsf{LK}\odot^{\infty}$. The regular proofs of $\mu\mathsf{LK}\odot^{\infty}$ can be represented as graphs, called circular proofs. The corresponding proof system is called $\mu\mathsf{LK}\odot^{\omega}$.

All the technical material for this part has been introduced in Chapters 1 and 2. Namely, the notion of an occurrence (of a formula) (Definition 1.21), the Fischer-Ladner closure (Definition 2.14), that od a thread (Definition 2.19), the existence of the minimum of a thread in some situations (Proposition 2.5, Proposition 2.7), will not be recalled here. It

$$\pi = \dfrac{\dfrac{\dfrac{\vdots}{\vdash \varphi \vee \psi}}{\vdash \varphi, \psi}\ (\mu),(\nu),(\nu)}{\vdash \varphi \vee \psi}\ (\vee) \qquad \theta = \dfrac{\dfrac{\dfrac{\vdots}{\vdash (\varphi \vee \psi)_{lii}, (\varphi \vee \psi)_{ri}}}{\vdash \varphi_l, \psi_r}\ (\mu),(\nu),(\nu)}{\vdash (\varphi \vee \psi)_\varepsilon}\ (\vee)$$

$$\theta_1 = \dfrac{\dfrac{\dfrac{(\star)}{\vdash (\varphi \vee \psi)_{lii}}\ (\mu),(\nu)}{\vdash \varphi_l}\ (\mathsf{W})}{(\star) \vdash (\varphi \vee \psi)_\varepsilon}\ (\vee) \qquad \theta_2 = \dfrac{\dfrac{\dfrac{(\star)}{\vdash (\varphi \vee \psi)_{ri}}\ (\nu)}{\vdash \psi_r}\ (\mathsf{W})}{(\star) \vdash (\varphi \vee \psi)_\varepsilon}\ (\vee)$$

Figure 7.4: $\mu\mathsf{LK}^{\omega}_{\mathsf{DHL}}$, $\mu\mathsf{LK}\odot^{\infty}$ and $\mu\mathsf{LK}\odot^{\omega}$ derivations of $\varphi \vee \psi$

is strongly advised to begin with the reading of these two introductory chapters. We have simply recalled the rules of $\mu\mathsf{LK}\odot$ and $\mu\mathsf{LK}\odot^{\omega}$ for clarity.

Notation 7.1. We sometimes write sequents of the form $\varphi_\varepsilon \vdash \psi_\varepsilon$ simply as $\varphi \vdash \psi$. This is in line with the convention 1.2 announced in the beginning of the thesis.

Definition 7.3. In any proof system, we call ***open derivation*** any derivation that contains a leaf sequent which is not justified by any rule. We call such a leaf a ***premise*** of the open derivation.

Comparison of $\mu\mathsf{LK}\odot^{\omega}$ with Dax *et al.* proof system. The proof system $\mu\mathsf{LK}\odot^{\omega}$ is very close to the one introduced by Dax *et al.* in [DHL06], that we call here $\mu\mathsf{LK}^{\omega}_{\mathsf{DHL}}$. In $\mu\mathsf{LK}^{\omega}_{\mathsf{DHL}}$, the sequents are sets of formulas, the rules are the same as those of $\mu\mathsf{LK}\odot^{\omega}$, the proofs as infinite derivations build using these rules and equipped with the same validity condition as $\mu\mathsf{LK}\odot^{\omega}$. Althought infinite, $\mu\mathsf{LK}^{\omega}_{\mathsf{DHL}}$ proofs are circular in the spirit, since a $\mu\mathsf{LK}^{\omega}_{\mathsf{DHL}}$ proof contains only finitely many sequents. The main difference between $\mu\mathsf{LK}^{\omega}_{\mathsf{DHL}}$ and $\mu\mathsf{LK}\odot^{\omega}$ is then the shape of sequents, and this is not a minor difference. Indeed, in $\mu\mathsf{LK}^{\omega}_{\mathsf{DHL}}$ the proof search is trivial, and this is not the case for $\mu\mathsf{LK}\odot^{\omega}$. For instance, let $\varphi = \mu X.\nu Y.X \vee Y$ and $\psi = \nu Y.\varphi \vee Y$ its unfolding. A $\mu\mathsf{LK}^{\omega}_{\mathsf{DHL}}$ proof π of $\varphi \vee \psi$ (Fig. 7.4) can be obtained by applying bottom up all the possible logical rules. If we apply the same rules in $\mu\mathsf{LK}\odot^{\omega}$, we get the $\mu\mathsf{LK}\odot^{\infty}$ proof θ (Fig. 7.4), which is not regular since the size of its sequents is unbounded. To get a regular proof, we have to apply some weakenings. But the choice of which formula to weaken is crucial, since a bad choice may lead to a non valid proof. For instance, if we weaken the formula ψ_r, the obtained derivation θ_1 (Fig. 7.4) is regular but does not satisfy the validity condition. The good choice of weakening is the one that fires φ_l, yielding the proof θ_2 (Fig. 7.4). Proving a valid sequent in $\mu\mathsf{LK}\odot^{\omega}$ is not trivial, since we have to do some clever choices to get derivations which are regular and valid.

Restriction on the syntax of formulas. In the rest of this part, we will restrict our attention to a specific class of formulas, the ***guarded*** formulas. This restriction simplifies the proofs at some places (for instance in Definition 9.2), but it is not a restriction since every formula is provably equivalent to a guarded formula.

Definition 7.4. A formula φ is said to be ***guarded*** if every bound variable of φ appears under the scope of a $\odot$ connective.

In [Koz83], it is shown that every formula if provably equivalent to a guarded formula in $\mu\mathsf{LK}\odot^{\omega}$, and this proof is constructive.

Proposition 7.1 (Kozen [Koz83])**.** *For every closed formula φ there is a semantically equivalent guarded formula ψ, that is $\mathcal{M}(\varphi) = \mathcal{M}(\psi)$, such that we can construct effectively a proof of the sequent $\psi \vdash \varphi$ in $\mu\mathsf{LK}\odot$.*

Proviso 7.1. Unless otherwise stated, all formulas are assumed to be closed, guarded, $\top$ and $\bot$ free. By earlier observations and by Proposition 7.1, this is not a restriction.

Relating the infinitary and the finitary proof systems. As announced in the introduction, we will use the circular proof system $\mu\mathsf{LK}\odot^{\omega}$ as an intermediary step to prove sequents in $\mu\mathsf{LK}\odot$. That is, to show a sequent s in $\mu\mathsf{LK}\odot$, we will build first a proof of s in $\mu\mathsf{LK}\odot^{\omega}$, then we will transform this proof into a $\mu\mathsf{LK}\odot$ proof. We have seen that the proof search in $\mu\mathsf{LK}\odot^{\omega}$ is already not trivial, but the situation gets more complicated since we do not have a general algorithm that translates all $\mu\mathsf{LK}\odot^{\omega}$ proofs into $\mu\mathsf{LK}\odot$ ones. We only have a sufficient condition on $\mu\mathsf{LK}\odot^{\omega}$ proofs, the *translatability criterion* (Definition 2.39), which guarantees their translatability into $\mu\mathsf{LK}\odot$ ones (Theorem 2.5). Thus, the proof search should take this parameter into account, and output circular proofs that satisfy this condition.

The translatability condition is much involved and sometimes we do not need it in all its generality. A weaker condition, presented below, will be used instead in some cases.

Definition 7.5. A $\mu\mathsf{LK}\odot^{\omega}$ (resp. $\mu\mathsf{LK}\odot^{\infty}$) derivation is ***meager*** if all the occurrences of (σ_{r}) and (σ_{l}) are of the following form, *i.e.*, there is no context around the unfolded formulas:

$$\frac{F[\sigma X.F/X] \vdash \Delta}{\sigma X.F \vdash \Delta}\ (\sigma_{\mathsf{l}}) \qquad \frac{\Gamma \vdash F[\sigma X.F/X]}{\Gamma \vdash \sigma X.F}\ (\sigma_{\mathsf{r}})$$

Proposition 7.2. *If π is a meager proof of a sequent s, then it can be transformed effectively into a $\mu\mathsf{LK}\odot$ proof of s.*

Proof. First, note that in a meager proof, every infinite branch contains at most one right trace and one left trace. Each of these traces is strongly valid, since whenver the trace meets a sequent, it meets it at the level of the same formula. Using Proposition 2.17, we can translate it effectively into a $\mu\mathsf{LK}\odot$ proof. □

Chapter 8

Automata over infinite words

Both the branching and the linear-time μ-calculus enjoy close and deep relationships with automata theory. Any formula can be compiled into an automaton which accepts exactly the models of this formula, and conversely. This correspondence turned out to be a valuable tool to investigate these logics. In fact, most of the deep results on the μ-calculus have been (can be) obtained using automata theory. Among these results, the exact complexity of the satisfiability problem [EJ91, Mad94], the strictness of the fixed point alternation hierarchy in the branching-time case [Bra98], the fact that this hierarchy collapses at level 0 in the linear-time case [Lan05], the correspondence with the monadic second order logic [Mad94], all build on the correspondence with automata theory.

As announced previously, we are going to use in our turn the correspondence between the μ-calculus and automata to get a constructive completeness result. This chapter is dedicated to the introduction of the different models of automata which we shall use for that purpose.

Organization of the chapter. We introduce in Section 8.1 the different models of automata, equipped with various acceptance conditions. Then we discuss in Section 8.2 the relationships between these different kinds of automata.

8.1 Alternating and (non-)deterministic automata

Automata over infinite words, also called ω-automata, are finite sate machines that run over infinite words, and depending on their models, accept or reject some of these words. For that purpose, when an ω-automaton reads a word, it outputs at the same time a structure usually called a ***run*** over this word. To assess whether this run is accepting or not, the ω-automaton is also equipped with an ***acceptance condition***. Thus, one can classify ω-automata according to two dimensions. The first one is the shape of their runs: if an automaton outputs runs which are trees, then it is called ***alternating***; when these runs are words, it is called ***non-deterministic***. The second dimension is the type of their acceptance condition, which can be more or less permissive.

In this section, we first introduce alternating automata with various acceptance conditions, and then we present non-deterministic automata as special cases of alternating ones. It is more usual and perhaps more progressive to start by introducing non-deterministic

automata which are conceptually easier, and then generalize them to alternating ones; we did here the opposite choice since it shortens the presentation and makes the notations more homogeneous and compact.

8.1.1 Alternating automata

Before going to the formal definitions, let us describe informally how an alternating automaton works. The computation of an APW over an infinite word proceeds in rounds. At the beginning of every round, there are several copies of the APW in some position of the word, each of them in its own state. During a round, each copy splits up in several new copies, which are sent to the successor of the current position, and change their states, all this is done according to the transition function. Initially, there is only one copy of the APW, it resides in the first position of the word, and starts in the initial state of the automaton. Every computation induces a tree, labelled with the states through which the copies of the automaton went during the computation; this tree is called a run and it witnesses the causality between a copy of the automaton and the new copies it yields.

To determine acceptance or rejection of a computation of an APW over an infinite word, the entire run is inspected; acceptance is then defined via path conditions for the infinite branches of the run. Namely, an infinite branch of the run will be accepting if it satisfies the acceptance condition; a run will be accepting if each of its infinite branches is accepting. Thus, the branching of the run can be thought of as a universal non-determinism. Alternating automata support also the other kind of non-determinism which is existential non-determinism. For example, during a round, a copy of the automaton may have the choice to split to different sets of copies. Hence an automaton may have many possible runs over a single word.

Formally, an alternating automaton is defined as follows.

Definition 8.1. An ***alternating automaton*** $\mathcal{A}$ is given by a ***transition structure*** (Σ, Q, Δ, q_I), where:

- Σ is an alphabet,
- Q is a finite set of states,
- $\Delta : Q \times \Sigma \times 2^Q$ is the transition relation,
- $q_I \in Q$ is the initial state

Together with an ***acceptance condition*** $Acc \subseteq Q^\omega$.

The transition structure is the part of the alternating automaton that reads the words and outputs runs. The acceptance condition will help to determine which runs are accepting.

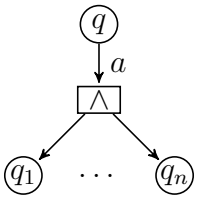
Remark 8.1. Notice that the transition structure is specified in a finite way, while the acceptance condition is a possibly infinite set. We will present later some acceptance conditions that can be specified in a finite way.

Example 8.1. Let $\Sigma_0 = \{a, b\}$ and let $\mathcal{A}_0 = (\Sigma_0, \{p, q\}, \Delta, p, Acc_0)$ be the alternating automaton such that:

$$\Delta = \{(p, a, \{p, q\}), (p, a, \{q\}), (p, b, \{p\}), (q, a, \{q\}), (q, a, \{p\})\}$$

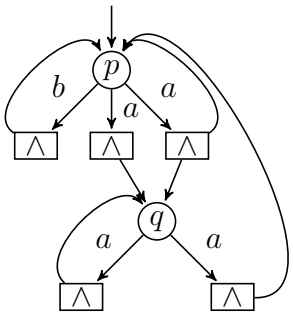
and $Acc_0 = (\{p, q\}^\star.q)^\omega$. We will use the alphabet Σ_0, the automaton $\mathcal{A}_0$ and its acceptance condition Acc_0 in the example of this section without recalling their definitions.

We give a graphical presentation of the transition structures of alternating automata by representing every transition $(q, a, \{q_1, \ldots, q_n\})$ as follows:



and by marking the initial state with an incoming edge without source.

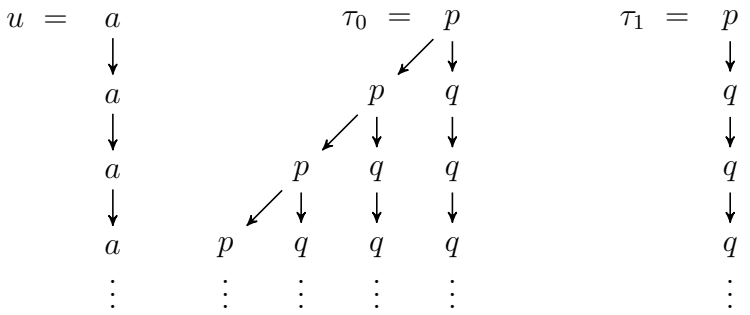
Example 8.2. The transition structure of $\mathcal{A}_0$ is represented as follows:



Definition 8.2. Let $\mathcal{A} = (\Sigma, Q, \Delta, q_I, Acc)$ be an alternating automaton and let $u = u_0u_1\ldots$ be an infinite Σ-word. A ***run*** of $\mathcal{A}$ over u is a Q-labelled tree $\tau = (T, l)$ satisfying the following conditions:

- $l(r) = q_I$ where r is the root of T.
- Let v be a node of T at level n and $v_1, \ldots, v_k$ be the sons of v. If we set $q = l(v)$ and $\forall i \leq k, q_i = l(v_i)$, then $(q, u_n, \{q_1, \ldots, q_k\}) \in \Delta$.

Example 8.3. The trees τ_0 and τ_1 are two runs of the automaton $\mathcal{A}_0$ over the word $u = a^\omega$.



Sometimes we want to talk about runs without mentioning the words, this is the role of the following definition:

Definition 8.3. A labelled tree τ is a ***run*** of an alternating automaton $\mathcal{A}$ if there is a word u such that τ is the run of $\mathcal{A}$ over u.

Later in this thesis, we will need to consider the branches of runs, we define them below.

Definition 8.4. Let $\mathcal{A}$ be an alternating automaton with a set of states Q. A Q-word ρ is a ***run-branch*** of $\mathcal{A}$ if there is a run τ of $\mathcal{A}$ such that ρ is a branch of τ.

Definition 8.5. Let $\mathcal{A} = (\Sigma, Q, \Delta, q_I, Acc)$ be an alternating automaton, τ be a run of $\mathcal{A}$ over a word u and let $\rho = q_0 q_1 \dots$ be a run-branch of τ. Let $\mathsf{Inf}(\rho)$ be the set of states that occur infinitely often in ρ.

- The ***run-branch ρ is accepting*** if $\rho \in Acc$.
- A ***run is accepting*** if all its run-branches are.
- The ***word u is accepted*** by $\mathcal{A}$ if there is an accepting run over it.
- The ***language of the automaton*** $\mathcal{A}$, denoted $\mathcal{L}(\mathcal{A})$, is the set of words accepted by $\mathcal{A}$.
- A ***language $\mathcal{L}$ is recognized by*** $\mathcal{A}$ if $\mathcal{L} = \mathcal{L}(\mathcal{A})$.

Example 8.4. In Example 8.3, the run τ_0 is not accepting since the path $p^\omega \notin Acc_0$. On the contrary the run τ_1 is accepting. It is easy to check that $\mathcal{L}(\mathcal{A}_0) = (b^\star.a.a^+)^\omega$.

To remain within the scope of finite automata, acceptance conditions have to be specified in a finite way. Various such finitary acceptance conditions exist and we give in the following some of them, together with the way they are specified:

Definition 8.6. Let $\mathcal{T} = (Q, \Delta, q_I)$ be a transition structure.

- A ***Büchi condition*** over $\mathcal{T}$ is given by a set $F \subseteq Q$ of ***accepting states***. An infinite Q-word ρ satisfies the Büchi condition if it contains infinitely many occurrences of states from F. The Büchi acceptance condition is then $Acc = \{\rho \in Q^\omega \mid \mathsf{Inf}(\rho) \cap F \neq \emptyset\}$, where $\mathsf{Inf}(\rho)$ is the set of states that appear infinitely often in ρ. A ***Büchi automaton*** is given by $(\mathcal{T}, F)$ instead of $(\mathcal{T}, Acc)$.
- A ***Rabin condition*** over $\mathcal{T}$ is given by a set $\mathcal{P} = \{(E_1, F_1), \dots, (E_k, F_k)\}$ of pairs of sets of states. An infinite Q-word ρ satisfies the Rabin condition if there is an index i such that ρ contains only finitely many states from E_i and infinitely many occurrences of states from F_i. The Rabin acceptance condition is then:

$$Acc = \bigcup_{1 \leq i \leq k} \{\rho \in Q^\omega \mid \mathsf{Inf}(\rho) \cap E_i = \emptyset \text{ and } \mathsf{Inf}(\rho) \cap F_i \neq \emptyset\}.$$

  A ***Rabin automaton*** is given by $(\mathcal{T}, \mathcal{P})$.

- A ***Streett condition*** over $\mathcal{T}$ is given by a set $\mathcal{P} = \{(E_1, F_1), \ldots, (E_k, F_k)\}$ of pairs of set states. An infinite Q-word ρ satisfies the Streett condition if for all index i, if ρ contains infinitely many occurrences of states from E_i then it contains also infinitely many occurrences of states from F_i. The Streett acceptance condition is then:

$$Acc = \bigcap_{1 \leq i \leq k} \{\rho \in Q^\omega \mid \mathsf{Inf}(\rho) \cap E_i \neq \emptyset \Rightarrow \mathsf{Inf}(\rho) \cap F_i \neq \emptyset\}.$$

  A ***Streett automaton*** is given by $(\mathcal{T}, \mathcal{P})$.

- A ***parity condition*** over $\mathcal{T}$ is given by a ***priority function*** $\{c : Q \to \omega\}$ that assigns to every state a natural number called its ***priority***. An infinite Q-word ρ satisfies the parity condition if the minimal priority that appears infinitely often is even. The parity acceptance condition is then $Acc = \{\rho \in Q^\omega \mid \mathsf{min}(\mathsf{Inf}(c(\rho)))$ is even$\}$, where $c(\rho)$ is the sequence $c(\rho_0)c(\rho_1)\ldots$. A ***parity automaton*** is given by $(\mathcal{T}, c)$.

Even though these acceptance conditions may look very different, they actually capture the same set of languages. Indeed, a language is recognized by an alternating Büchi automaton iff it is recognized by an alternating Rabin (resp. Streett, resp. Parity) automaton [**?**]. We will come back to the issue of comparing acceptance conditions in Section 8.2.

Example 8.5. The acceptance condition of the automaton $\mathcal{A}_0$ can be expressed either as a:

- Büchi condition where $F = \{q\}$.
- Parity condition where $c(p) = 1, c(q) = 0$.

Remark 8.2. What happens in the previous example is a general fact: every Büchi automaton $\mathcal{B} = (\mathcal{T}, F)$ can be seen as the parity automaton $\mathcal{P} = (\mathcal{T}, c)$, where $c(q) = 0$ if $q \in F$ and $c(q) = 1$ otherwise. It is indeed easy to check that $\mathcal{L}(\mathcal{B}) = \mathcal{L}(\mathcal{P})$.

Remark 8.3. Notice that the validity condition on threads (Definition 2.19) has the flavour of a parity condition: the formulas of the thread can be seen as states of an automaton, the least (resp. greatest) fixed point formulas as the odd (resp. even) states and the subformula ordering corresponds to the priority ordering. This analogy between formulas and alternating parity automata will be explored later.

8.1.2 Non-deterministic automata

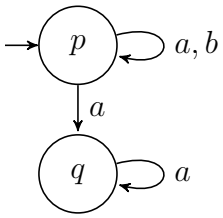
Non-deterministic automata are particular cases of alternating automata where the existential non-determinism is allowed but not the universal one. This is reflected by the fact that when a state processes a word, it reads its first letter and sends the rest of the word to at most one state, while in the alternating case, it was allowed to send it to many states. This restriction affects also the shape of the runs which become words instead of trees. Formally, non-deterministic automata are defined as follows:

Definition 8.7. An alternating automaton $\mathcal{A} = (\Sigma, Q, \Delta, q_I, Acc)$ is said to be ***non-deterministic*** if the elements of Δ are of the form $(p, a, \{q\})$.

Remark 8.4. If there is no ambiguity, we consider Δ as a subset of $Q \times \Sigma \times Q$. Notice that the runs of non-deterministic automata can be seen as infinite words over Q.

Example 8.6. Let $\mathcal{N} = (T, Acc_0)$ be the non-deterministic automaton where $T = (\{p, q\}, \Delta_N, \{p\})$ and $\Delta_N = \{(p, a, p), (p, b, p), (p, a, q), (q, a, q)\}$.

We simplify our graphical presentation of transition structures in the case of non-deterministic automata, by representing every transition (p, a, q) by an edge labelled a between the nodes p and q. The transition structure of $\mathcal{N}$ is then represented as follows:



We show in the following two runs of $\mathcal{N}$ over $u = a^\omega$:

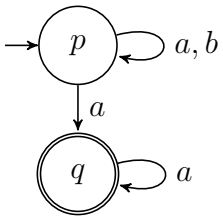
$$u = a \to a \to a \to a \cdots$$

$$\rho_0 = p \to p \to p \to p \cdots$$

$$\rho_1 = p \to q \to q \to q \cdots$$

The run ρ_0 is not valid while ρ_1 is. It is easy to see that the language of $\mathcal{N}$ is $\{a, b\}^\star.\{a\}^\omega$. Notice that we can specify $\mathcal{N}$ as the Büchi automaton $(T, \{q\})$.

In the case of Büchi automata, the graphical representation of the transition structure becomes a graphical representation of the whole automaton by double-circling the accepting states. For example, the Büchi automaton $(T, \{q\})$ is displayed as follows:



One can equip the transition structure T with other acceptance conditions, for example with the parity condition $c(p) = 2$ and $c(q) = 1$, the resulting automaton has Σ_0^ω as a language.

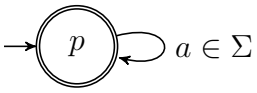
8.1.3 Deterministic automata

For a non-deterministic automaton, one may have many possible runs over a single word, as shown in the previous example. A deterministic automaton is a non-deterministic automaton for which there is only one possible run over a given word.

Definition 8.8. A non-deterministic automaton $(\Sigma, Q, \Delta, q_I, Acc)$ is said to be ***deterministic*** if and only if for every transitions $(q, a, q_1), (q, a, q_2) \in \Delta$ we have that $q_1 = q_2$.

Remark 8.5. It is easy to check that for a deterministic automaton, every word has at most one run.

Example 8.7. Let Σ be an alphabet. The following Büchi automaton, denoted $\mathcal{U}_\Sigma$, is deterministic:



Notation 8.1. We denote automata over infinite words by three letters:

- The first one (D, N or A) indicates if the automaton is deterministic, non-deterministic or alternating,
- the second one (B, R, S or P) specifies the acceptance condition: Büchi, Rabin, Streett or Parity,
- the third one (W or T) precises the nature of the input: word or tree. In this thesis we only deal with automata over words.

For example, we denote the class of non-deterministic Büchi automata by NBW, that of deterministic Rabin automata by DRW and that of alternating parity automata by APW.

8.2 Comparing ω-automata

Every class $\mathcal{C}$ of automata defines a class of languages $\mathcal{L}_\mathcal{C}$ which is the set of languages that can be recognized by the automata of this class. We can compare the power of two classes of automata by comparing their corresponding classes of languages. It turns out that all the models of automata presented before (alternating, deterministic, non-deterministic) with the various acceptance conditions (Büchi, Parity, Streett and Rabin) are equivalent, that is, they define the same class of languages, except for DBW which is strictly less powerful. Indeed, one can show that the language $\{a,b\}^\star\{a\}^\omega$ which is recognized by the non-deterministic Büchi automaton $\mathcal{N}$ of Example 8.6 is not recognized by any deterministic Büchi automaton.

Proposition 8.1. *The language $\{a,b\}^\star\{a\}^\omega$ is not recognized by any deterministic Büchi automaton.*

Proof. ([Var95]) Assume by contradiction that there is a DBW $\mathcal{D} = (\Sigma, Q, \Delta, q_I, F)$ recognizing this language. If w is an infinite word, ρ a run of $\mathcal{D}$ over w and u a finite prefix of w of length k, we denote by $\rho(u)$ the k^{th} element of ρ.

Consider the infinite word $w_0 = a^\omega$. Clearly, w_0 is accepted by $\mathcal{D}$, so $\mathcal{D}$ has an accepting run ρ_0 on w_0 . Thus, w_0 has a finite prefix u_0 such that $\rho_0(u_0) \in F$. Consider now the infinite word $w_1 = u_0.b.a^\omega$. Clearly, w_1 is also accepted by $\mathcal{D}$, so $\mathcal{D}$ has an accepting run ρ_1 on w_1. Thus, w_1 has a finite prefix $u_0.b.u_1$ such that $\rho_1(u_1) \in F$. In a similar way, we can continue to find a word w_i, a run ρ_i and finite prefix u_i of w_i such that $\rho_i(u_0.b.u_1.b\ldots.u_i) \in F$. Since Q is finite, there are i and j , where $0 \leq i < j$, such that $\rho_i(u_0.b.u_1\ldots.u_i) = \rho_i(u_0.b.u_1\ldots.u_i.b\ldots u_j)$. It follows that the word $u_0.b.u_1\ldots.u_i.(b\ldots u_j)^\omega$

has an accepting run over $\mathcal{D}$. But this word contains infinitely many b's, so it is not in our initial language. □

The set of languages recognized by these different (yet equivalent) classes of automata are called ***ω-regular*** languages. These languages enjoy a bunch of nice stability properties: stability by union, intersection, complementation, etc. Moreover, they can be defined independently from any notion of machine or automata [GTW02a].

The equivalences between these different classes of ω-automata are usually established by showing automata transformations that convert an automaton of a given model into an automaton in the other models, by preserving the language. We will be interested, in the proof of completeness, in the following three equivalence results:

Theorem 8.1. *For every APW we can construct a NPW recognizing the same language.*

Theorem 8.2. *For every NPW we can construct a NBW recognizing the same language.*

Theorem 8.3. *For every NBW we can construct a DRW recognizing the same language.*

The first result is called non-determinization. It expresses the fact that when we deal with infinite words, the universal branching is unnecessary, and we can restrict the shape of runs to be simply words.

The second result can be considered as part of a wider problem: the Motskowski-Rabin index problem, which for a given ω-regular language of infinite trees or words, asks about the minimal number of priorities needed to recognise this language with a non-deterministic parity automaton. This minimal index orders regular languages into a hierarchy called Motskowski-Rabin hierarchy. For tree languages, whether this problem can be answered effectively is a long-standing open problem, solved so far only for languages recognisable by deterministic automata. For word languages, the answer is much more easier, and we can reformulate Theorem 8.2 as: the Motskowski-Rabin hierarchy of non-deterministic automata collapses at the level of Büchi automata, that is at index 2, since we have seen that Büchi automata can be viewed as Parity automata with the two priorities 0 and 1 (Remark 8.2).

The third result is called a determinization result. Compared to the two previous theorems, it has a particular shape: when we determinize a Büchi automaton, we fall out the class of Büchi automata, and we cannot do better because of the language of Proposition 8.1. Note that this is particular to the Büchi condition, since the other acceptance condition can be determinized internally: for every NRW (NPW) we can construct effectively a DRW (DPW) recognizing the same language.

Automata have a certain similarity with formulas. The language of an automaton, which is a set of words over an alphabet Σ can be compared with the semantics of a formula which is a set of words over the alphabet $2^{\mathcal{P}}$. If we consider automata over this last alphabet, we can really compare automata and formulas. In the next chapter, we will do this by encoding every automaton $\mathcal{A}$ by a formula $[\mathcal{A}]$ such that $\mathcal{L}(\mathcal{A}) = \mathcal{M}([\mathcal{A}])$.

With this encoding, Theorems 8.1 and 8.2 give us, for every APW (resp. NPW) $\mathcal{A}_1$, a NPW (resp. NBW) $\mathcal{A}_2$ such that $\models [\mathcal{A}_1] \Leftrightarrow \mathcal{A}_2$. To get the steps II and III of our proof of completeness, we have to go further and prove that this equivalence can be established at the provability level as well, that is $\vdash_{\mu\mathsf{LK}^{\odot}} [\mathcal{A}_1] \Leftrightarrow \mathcal{A}_2$. This is what we shall do in Chapter 10.

We can relate the particular shape of Theorem 8.3 compared to the two other theorems to the particular shape of Step IV compared to Steps II-III.

As announced in the introduction, we will establish a different result (we called it Step IV'), from which Step IV will follow easily: if $\mathcal{B}_1, \mathcal{B}_2$ are NBW such that $\mathcal{L}(\mathcal{B}_1) \subseteq \mathcal{L}(\mathcal{B}_2)$, then one can construct a $\mu\mathsf{LK}\odot$ proof of $[\mathcal{B}_1] \vdash [\mathcal{B}_2]$.

One can wonder what this result has to do with Theorem 8.3, since there is no DRW automaton in its statement. Actually, we will use the algorithm that generates the states of the DRW of Theorem 8.3, called Safra's construction, as a guide to perform a proof search. This is will be made clear in Chapter 11.

Chapter 9

Alternating parity automata and the μ-calculus

As said before, there is a fruitful relationship between the μ-calculus and automata theory, at the core of which lies the equivalence between linear-time μ-calculus formulas and alternating parity automata (APW). This equivalence is always shown at a semantical level: every APW $\mathcal{A}$ can be encoded by a formula $[\mathcal{A}]$ such that $\mathcal{M}([\mathcal{A}]) = \mathcal{L}(\mathcal{A})$, and conversely for every formula φ, there is an APW $\mathcal{A}_\varphi$ such that $\mathcal{M}(\varphi) = \mathcal{L}(\mathcal{A}_\varphi)$. We show in this section that beyond this semantical equivalence, there is also an equivalence at the level of provability, that is, φ and $[\mathcal{A}_\varphi]$ are provably equivalent in $\mu\mathsf{LK}\odot$. Lifting this semantical equivalence to the provability level relies very precisely on the encoding and the shape of the automaton $\mathcal{A}_\varphi$. That is why we introduce our own encoding of automata in the μ-calculus, that respects the shape of the input automaton; and our construction of $\mathcal{A}_\varphi$, which yields an automaton that sticks the most to the structure of φ. This last construction relies heavily on ***operational semantics***, which is an alternative definition of formula semantics, introduced by Janin and Walukiewicz in [JW95], which makes us look at formulas as automata-like devices, which is precisely what we need.

Organization of the chapter. We recall in Section 9.1 the operational semantics for the linear-time mu-calculus formulas. Then we define in Section 9.2 our encoding of APW in the the μ-calculus, and conversely we construct in Section 9.3 for every μ-calculus formula an APW. We finally show in Section 9.4 that the encoding of this automaton is provably equivalent to the original formula in $\mu\mathsf{LK}\odot$.

9.1 Operational semantics

Definition 7.2 of semantics is not very convenient to compute the models of a formula or to compare the models of two formulas, namely because the interpretation of fixed points uses arbitrary intersections and unions. We recall an alternative presentation of the semantics of μ-calculus, introduced by Walukiewicz and Janin in [JW95] under the name of ***operational semantics***, which is equivalent to the original definition. Their definition being introduced for the modal μ-calculus, we specialize it to the linear-time case.

Definition 9.1. We call ***constraint*** any set of literals. The ***constraint of a letter*** $a \in \Sigma$, denoted $c(a)$, is the constraint $a \cup \{\mathfrak{p}^\perp \mid \mathfrak{p} \notin a\}$. A letter a ***satisfies a constraint*** c, and we write $a \models c$, iff $c \subseteq c(a)$.

Definition 9.2. Let F be an occurrence. The ***F-derivation*** is a $\mu\mathsf{LK}\odot^\infty$ derivation of conclusion $F \vdash$ obtained by applying all the possible logical rules $(\mu, \nu, \vee, \wedge)$, except for the $\odot$ rule, to F and its sub-occurrences, until reaching sequents of the form $C, \odot\Delta \vdash$, where the set $\overline{C}$ obtained by forgetting the addresses of C, is a constraint. The set of ***successors of F***, denoted $S(F)$, is the set of premises of the F-derivation.

Remark 9.1. Notice that the F-derivation is finite, because F is guarded.

Remark 9.2. There are many possible F-derivations for a given occurrence F, differing only by the order of application of the logical rules. They all share the same set of premises and the same structure of threads. We choose any representative of this class of derivations to be ***the*** F-derivation.

Example 9.1. Let $F = \nu X.\mathfrak{p} \wedge ((\odot X)_\alpha \vee G) \wedge (\odot X)_\beta$, where $G = \mu Y.\mathfrak{q} \wedge (\odot Y)_\gamma$. The F-derivation is the following:

$$\dfrac{\dfrac{\mathfrak{p}, (\odot F)_\alpha, (\odot F)_\beta \vdash \qquad \dfrac{\mathfrak{p}, \mathfrak{q}, (\odot G)_\gamma, (\odot F)_\beta \vdash}{\mathfrak{p}, G, (\odot F)_\beta \vdash}\ (\mu),(\wedge_1)}{\mathfrak{p}, (\odot F)_\alpha \vee G, (\odot F)_\beta \vdash}\ (\vee)}{F \vdash}\ (\nu),(\wedge),(\wedge)$$

We sometimes write an F-derivation as a rule named F:

$$\dfrac{\mathfrak{p}, (\odot F)_\alpha, (\odot F)_\beta \vdash \qquad \mathfrak{p}, \mathfrak{q}, (\odot G)_\gamma, (\odot F)_\beta \vdash}{F \vdash}\ (F)$$

Recall that if F is an occurrence and β is an address, F_β is the relocation of F in β (Definition 1.26). Recall also that we made the choice not to write explicitly the addresses of atoms.

Definition 9.3. Let $\Delta := \{F_i\}_{1 \leq i \leq n}$ be a set of occurrences. The ***Δ-derivation*** is a $\mu\mathsf{LK}\odot^\infty$ open derivation of conclusion $\Delta \vdash$ obtained by applying successively, and for all $1 \leq i \leq n$, the F_i-derivations until reaching sequents of the form $C, \odot\Delta \vdash$ where $\overline{C}$ is a constraint. The set of ***successors of Δ***, denoted $S(\Delta)$, is the set of premises of the Δ-derivation.

Remark 9.3. Here again, there are many possible Δ-derivations but they differ only by the order in which F_i-derivations are applied. They all share the same structure of threads and the same set of premises.

Example 9.2. Let F be the occurrence introduced in Example 9.1 and $H = \nu X.\mathfrak{r} \wedge (\odot X)_\delta$. The $\{F, H\}$-derivation is the following:

$$\dfrac{\dfrac{\mathfrak{p}, \mathfrak{r}, (\odot F)_\alpha, (\odot F)_\beta, (\odot H)_\delta \vdash}{\mathfrak{p}, (\odot F)_\alpha, (\odot F)_\beta, H \vdash}\ (H) \qquad \dfrac{\mathfrak{p}, \mathfrak{q}, \mathfrak{r}, (\odot G)_\gamma, (\odot F)_\beta, (\odot H)_\delta \vdash}{\mathfrak{p}, \mathfrak{q}, (\odot G)_\gamma, (\odot F)_\beta, H \vdash}\ (H)}{F, H \vdash}\ (F)$$

We sometimes write a Δ-derivation as a rule named Δ:

$$\frac{\mathfrak{p}, \mathfrak{r}, (\odot F)_\alpha, (\odot F)_\beta, (\odot H)_\delta \vdash \qquad \mathfrak{p}, \mathfrak{q}, \mathfrak{r}, (\odot G)_\gamma, (\odot F)_\beta, (\odot H)_\delta \vdash}{F, H \vdash} \ (\{F, H\})$$

Definition 9.4. We define $\Pi(\Delta)$ to be the $\mu\mathsf{LK}\odot^\infty$ pre-proof of conclusion $\Delta \vdash$, obtained by applying coinductively the following scheme:

$$\Pi(\Delta) \quad = \quad \dfrac{\dfrac{\Pi(\Delta_1)}{C_1, \odot\Delta_1 \vdash}\ (\odot) \qquad \ldots \qquad \dfrac{\Pi(\Delta_n)}{C_n, \odot\Delta_n \vdash}\ (\odot)}{\Delta \vdash}\ (\Delta)$$

Example 9.3. Let $\varphi = \mu X.\nu Y.\odot((\mathfrak{p} \wedge Y) \vee X)$, $\psi = \nu Y.\odot((\mathfrak{p} \wedge Y) \vee \varphi)$ and $\delta = (\mathfrak{p} \wedge \psi) \vee \varphi$. The derivation $\Pi(\delta)$ is:

$$\dfrac{\dfrac{\dfrac{(\dagger)}{\delta_{lrii} \vdash}}{\mathfrak{p}, (\odot\delta)_{lri} \vdash}\ (\odot) \qquad \dfrac{\dfrac{(\dagger)}{\delta_{riii} \vdash}}{(\odot\delta)_{rii} \vdash}\ (\odot)}{(\dagger)\ \delta_\varepsilon \vdash}\ (\delta_\varepsilon)$$

Remark 9.4. Notice that $\Pi(F)$ is not a $\mu\mathsf{LK}\odot^\infty$ proof in general, because it does not always satisfy the validity condition. This is not a problem, since the goal is to use it only as a support to compute the set of models of F.

Definition 9.5. Let β be a branch of $\Pi(F)$ and $((C_i, \odot\Delta_i))_{i\in\omega}$ be the sequence of the conclusions of the $\odot$ rules in β. A word $u = (a_i)_{i\in\omega}$ is said to be ***induced*** by β iff $\forall i$, $a_i \models \overline{C_i}$. The branch β is said to be ***accepting*** if it contains no μ-thread. The ***language of*** F, denoted $\mathcal{L}(F)$, is the set of words induced by the accepting branches of $\Pi(F)$.

Example 9.4. If $\mathcal{P} = \{\mathfrak{p}\}$ then $\Sigma = \{a, b\}$ where $a = \emptyset$ and $b = \{\mathfrak{p}\}$. The language of δ from Example 9.3 is $\{a, b\}^\star.b^\omega$.

Remark 9.5. The use of automata-theoretic vocabulary is due to the fact that we can see an occurrence F as a sort of APW. Indeed, when we put F in the left-hand side of the sequent, the left disjunction rules become branching, and we can see each branch as a non-deterministic choice of a run, while the conjunction rule is non branching and we can see the application of the conjunction rule as constructing an alternating run. Hence, every infinite branch of $\Pi(F)$ can be seen as a run of an alternating automaton. Moreover, the acceptance condition of a branch of $\Pi(F)$ recalls the parity condition for APW. We will make this remark more precise in the next sections.

Proposition 9.1 (Janin and Walukiewicz [JW95])**.** *For every formula F, $\mathcal{L}(F) = \mathcal{M}(F)$.*

9.2 From APW automata to formulas

The linear-time μ-calculus contains all the ingredients to encode APW : disjunction and conjunction to simulate existential and universal non-determinism; least and greatest fixed points to encode odd and even states.

To get a match between the language of an automaton and the models of the formula encoding it, we will consider automata over the alphabet $\Sigma = 2^{\mathcal{P}}$ where $\mathcal{P}$ is the set of atoms, whose elements will simply be denoted by a, b, etc.

9.2.1 The encoding

We first show our encoding of the letters of Σ in the μ-calculus.

Definition 9.6. Let $a \in \Sigma$. The encoding of a, denoted $[a]$, is the following formula:

$$[a] := (\wedge_{\mathfrak{p} \in a} \mathfrak{p}) \wedge (\wedge_{\mathfrak{q} \notin a} \mathfrak{q}^{\perp})$$

We eventually simply write a for $[a]$ if there is no ambiguity.

Remark 9.6. The encoding of a letter a corresponds to the conjunction of the elements of $c(a)$, the constraint of a (Definition 9.1).

Definition 9.7. Let $\mathcal{A} = (\Sigma, Q, \Delta, q_I, c)$ be an APW. A ***run-section*** is a sequence $\Gamma = ((q_i, t_i))_{0 \leq i \leq n}$ of pairs in $Q \times \Delta$ *s.t.* $q_0 = q_I$ and $\forall i \leq n,\ \exists a_i, E_i$ st. $t_i = (q_i, a_i, E_i)$ and $\forall i < n,\ q_{i+1} \in E_i$. We say that Γ ***enables*** a state q if $q \in E_n$.

Definition 9.8. Let $\mathcal{A} = (Q, q_I, \Delta, c)$ be an APW . We assume a collection of variables $(X_q)_{q \in Q}$. We define the ***formula*** $[q]^{\Gamma}$ ***encoding*** the state $q \in Q$ under the run-section Γ, as follows:

$$[q]^{\Gamma} = X_q \quad \text{if} \quad \begin{cases} (1)\ \Gamma = \Gamma', (q,t), (q_1,t_1), \ldots, (q_n,t_n) \text{ and} \\ \\ (2)\ q_i \neq q, c(q_i) \geq c(q) \text{ for all } 1 \leq i \leq n, \end{cases}$$

$$[q]^{\Gamma} = \sigma X_q. \bigvee_{a \in \Sigma, t=(q,a,E) \in \Delta} [a] \wedge \bigwedge_{p \in E} \odot [p]^{\Gamma,(q,t)} \text{ otherwise}$$

with $\sigma = \nu$ iff $c(q)$ is even.

We finally set $[\mathcal{A}] = [q_I]^{\emptyset}$.

The encoding starts from the initial state and traverses the automaton, encoding every state q by a fixed point μX_q, if q is a state with an odd priority and νX_q otherwise. The environment Γ remembers the states that have been visited from the initial state to the current state. If the current state q has been seen before (and if the states seen since the last time q was visited have bigger priorities) then we encode it by its corresponding variable X_q, which ensures that the encoding algorithm will halt at some point.

Example 9.5. Let $\mathcal{A} = (T, c)$ be the APW automaton where T is the transition structure shown in Figure 9.1 and c is the priority function defined by $c(p) = 1$ and $c(q) = 2$. The encoding of $\mathcal{A}$ is the formula:

$$[\mathcal{A}] = \mu X_p.([a] \wedge \odot X_p) \vee ([a] \wedge \odot(\nu X_q.[b] \wedge \odot X_q \wedge \odot X_p) \wedge \odot X_p)$$

whose tree is presented next to the transition structure T to stress the similarities between the automaton and its encoding. We omitted the edge labels in the tree of the formula for clarity. To simplify the presentation, we merged the chain of nodes of the following shape into one n-ary node labelled $[a] \wedge \odot$ as follows:

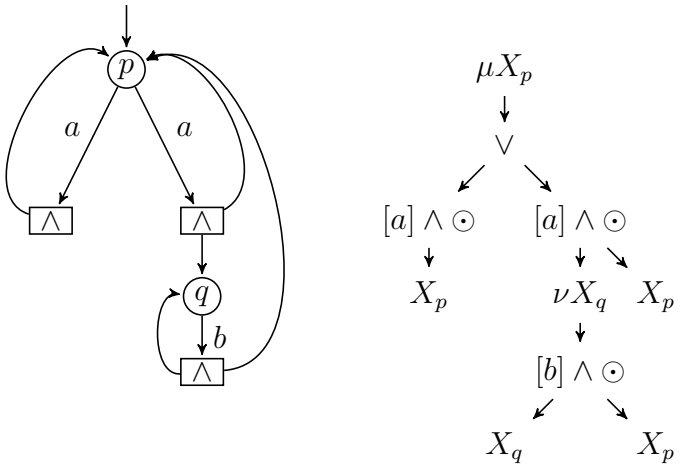
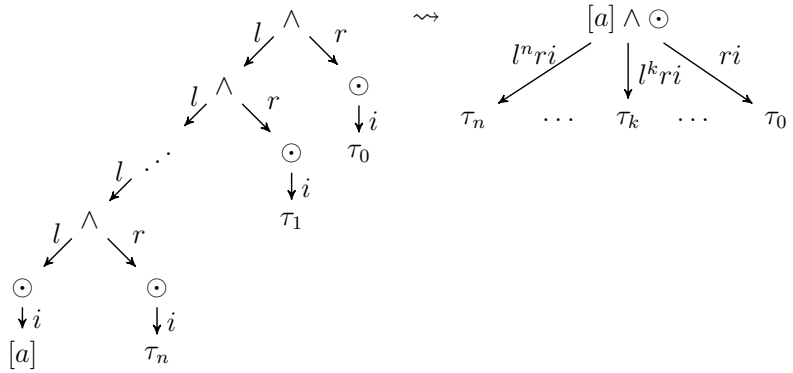


Figure 9.1: The automaton $\mathcal{A}$ and its encoding.



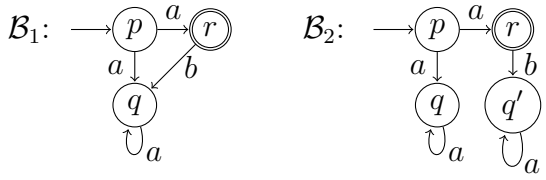
We adopt this simplification for the rest of this section.

Remark 9.7. Our encoding works also for NPW automata since they are special cases of APW automata. We have seen in Section 8 (Remark 8.2) that every Büchi automaton $\mathcal{B} = (\mathcal{T}, F)$ can be transformed into as Parity automaton $P_{\mathcal{B}} = (\mathcal{T}, c)$ where c is the priority function that assigns 0 to final states and 1 to non-final ones. We set that the encoding of a Büchi automaton $\mathcal{B}$ is the encoding of $P_{\mathcal{B}}$.

Remark 9.8. In automata, there is a possibility of sharing, in the sense that two different states can have the same successor. In formulas there is no such possibility since formulas have the shape of trees with back edges. What the encoding morally does, is to duplicate some states of the automaton in order to get a transition structure having the shape of a tree with back-edges. This transformation yields a bisimilar automaton, thus it preserves the language. For example, consider the Büchi automaton $\mathcal{B}_1$ below. Its encoding is the formula:

$$[\mathcal{B}_1] = \mu X_p.([a] \wedge \odot \nu X_r.[a] \wedge \odot \mu X_q.[a] \wedge \odot X_q) \vee ([b] \wedge \odot \mu X_q.[a] \wedge \odot X_q)$$

which is exactly the encoding of the automaton $\mathcal{B}_2$, obtained by duplicating the state q in $\mathcal{B}_1$.



A key ingredient in this encoding is the side condition of the first case of the definition. The aim of this condition is to bridge the gap between the acceptance condition on runs of the automaton and the validity condition on threads. The latter is almost a parity condition, but with a parity ordering corresponding to the subformula ordering. To obtain a match between the two orderings, we need to control the creation of cycles.

Let us discuss now why a naive encoding of APW cannot work, and why do we need such a complicated side condition in our encoding. A naive encoding traverses the automaton and encodes states with odd priorities with least fixed point formulas, states with even priorities with greatest fixed point formulas exactly as our encoding does, but when it encounters a state that has been visited before, it encodes it by its corresponding variable. Formally, one can define the naive encoding of APW as follows:

Definition 9.9. Let $\mathcal{A} = (Q, q_I, \Delta, c)$ be an alternating parity automaton. We assume a collection of variables $(X_q)_{q\in Q}$. We define the formula $\lfloor q \rfloor^\Gamma$, the ***naive encoding*** of the state $q \in Q$ under the environment Γ consisting of a list of states, as follows:

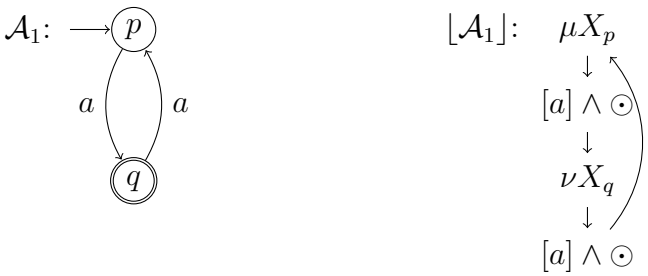
$$\lfloor q \rfloor^\Gamma = X_q \qquad \text{if } q \in \Gamma,$$

$$\lfloor q \rfloor^\Gamma = \sigma X_q. \bigvee_{a\in\Sigma,(q,a,E)\in\Delta} [a] \wedge \bigwedge_{p\in E} \odot \lfloor p \rfloor^{\Gamma,q}$$

otherwise, with $\sigma = \nu$ iff $c(q)$ is even.

When unspecified, Γ is taken to be empty. We finally set $\lfloor \mathcal{A} \rfloor = \lfloor q_I \rfloor$.

Consider the following Büchi automaton $\mathcal{A}_1$. Its naive encoding is the formula $\lfloor \mathcal{A}_1 \rfloor = \mu X_p.(a \wedge \odot(\nu X_q.a \wedge \odot X_p))$, whose graph is drawn next to the graph of $\mathcal{A}_1$.



The problem is that the language of $\mathcal{A}_1$ is $\{a^\omega\}$ while the set of models of $\lfloor \mathcal{A}_1 \rfloor$ is $\emptyset$. This is problematic since our goal is to have an encoding [.] that preserves the language of the automaton, that is $\mathcal{L}(\mathcal{A}) = \mathcal{M}([\mathcal{A}])$. We will show in Proposition 9.5 that our encoding satisfies this property, but for the moment let us try to understand why there is a gap between the language of the automaton $\mathcal{A}_1$ and the models of its naive encoding. To compute the models of the formula $\lfloor \mathcal{A}_1 \rfloor$, we will use the operational semantics. We show

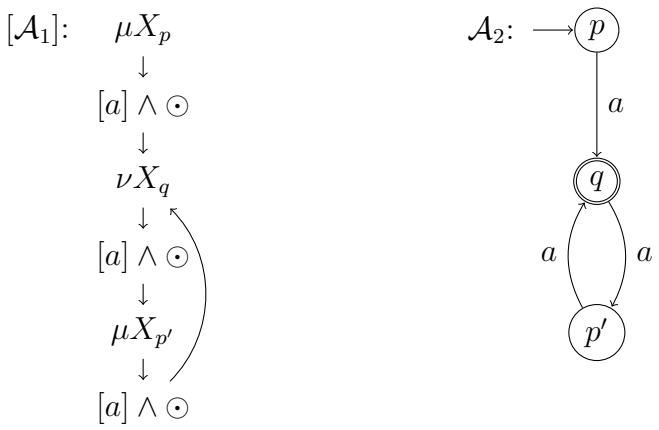
below the derivation $\Pi(\lfloor \mathcal{A}_1 \rfloor)$, where $\varphi^p = \lfloor p \rfloor$ and $\varphi^q = \nu X_q.[a] \wedge \odot \lfloor p \rfloor$. We omitted the formulas addresses in this derivation for clarity.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(\dagger)}{\varphi^p \vdash}}{c(a), \odot\varphi^p \vdash}\,(\odot)}{\varphi^q \vdash}\,(\varphi^q)}{c(a), \odot\varphi^q \vdash}\,(\odot)}{(\dagger)\ \varphi^p \vdash}\,(\varphi^p)$$

The unique branch of $\Pi(\lfloor \mathcal{A}_1 \rfloor)$ induces the word a^ω, but this branch is not accepting because the unique thread t of this branch is a μ-thread. Indeed, the thread t correspond to the graph traversal of $\lfloor \mathcal{A}_1 \rfloor$ that visits infinitely often the nodes μX_p and νX_q, hence by Proposition 2.7 the minimum of t is the formula whose node is the nearest to the root, which is φ_p. On the other hand, in the unique run of $\mathcal{A}_1$ over a^ω which is $(pq)^\omega$, the state with the minimal color is q whose node is not the closest to the root.

Let us see now our encoding of $\mathcal{A}_1$. The graph of the formula $[\mathcal{A}_1]$ is shown below. Morally, what our encoding does is to encode $\mathcal{A}_1$ exactly as the automaton $\mathcal{A}_2$, which is an unfolding of $\mathcal{A}_1$. One has:

$$[\mathcal{A}_1] = [\mathcal{A}_2] = \lfloor \mathcal{A}_2 \rfloor = \mu X_p.\ a \wedge \odot(\nu X_q.\ a \wedge \odot(\mu X_{p'}.\ a \wedge \odot X_q))$$



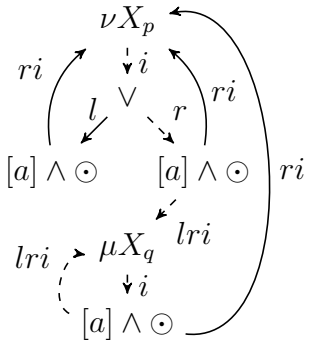
Actually, what our encoding does is to first rewrite the graph of the automaton as a tree with back edges (Remark 9.8), then thanks to the side conditions 1) and 2), it unfolds this graph in such way that for every loop $q_0 q_1 \dots q_n q_0$ where q_0 is the nearest node to the root in this loop, the states q_i have bigger priorities than q_0. This ensures that in an infinite run, the state that occurs infinitely often and has the minimal priority is the nearest one to the root among those appearing infinitely often in the run. This transformation yields an automaton which is bisimilar to the original one, hence it preserves the language. We denote this transformation $\tau(.)$. Even though we did not define formally this transformation, one can be convinced easily that our encoding of the original automaton can be seen as the naive encoding for the transformed automaton, that is $[\mathcal{A}] = \lfloor \tau(\mathcal{A}) \rfloor$.

9.2.2 Fischer-Ladner subformulas of the encoding

The $\mu\mathsf{LK}\odot^{\omega}$ proofs involving a formula of the form $[\mathcal{A}]$, that we will deal with later, will not decompose $[\mathcal{A}]$ into its subformulas but will rather decompose it into its FL-suboccurrences. To simplify the manipulation of these FL-suboccurrences, we introduce in this section some handy notation, then we relate the threads of $[\mathcal{A}]$, which are sequences of FL-suboccurrences of $[\mathcal{A}]$, to the runs of the automaton $\mathcal{A}$.

Definition 9.10. Let $\Gamma = ((q_i, t_i))_{0\leq i\leq n}$ be a run-section of the APW $\mathcal{A}$, and q a state enabled by Γ. Let p be the path in the graph of $[\mathcal{A}]$ that visits successively the nodes labelled σX_{q_i}, according to the transitions t_i, and ends with the node labelled σX_q. The ***address of*** (Γ, q), denoted $\alpha_{\Gamma,q}$, is the label of p.

Example 9.6. Let $\mathcal{A}$ be the APW of Example 9.5. The graph of the encoding of $\mathcal{A}$ is the following:



The run-section $\Gamma = (p, t_2), (q, t_3)$ enables the state q, its path is the dashed one. The address $\alpha_{\Gamma,q}$ is $irlriilri$.

Definition 9.11. Let Γ be a run-section of $\mathcal{A}$, and q a state enabled by Γ. We denote by $[\![q]\!]^{\Gamma}$ the FL-suboccurrence of $[\mathcal{A}]$ at the address $\alpha_{\Gamma,q}$, *i.e.*, the occurrence $\varphi_{\alpha_{\Gamma,q}}$ such that: $[\mathcal{A}]_{\varepsilon} \to^{\star} \varphi_{\alpha_{\Gamma,q}}$.

Example 9.7. Let Γ be the run-section of Example 9.6. We have $[\![q]\!]^{\Gamma} = (\mu X_q.[b] \wedge \odot X_q \wedge \odot[\mathcal{A}])_{\alpha_{\Gamma,q}}$.

Remark 9.9. We have chosen the notation $[\![q]\!]^{\Gamma}$ because of the proximity between these FL-suboccurrences and the formulas $[q]^{\Gamma}$. Indeed, one can show that $[\![q]\!]^{\Gamma}$ is obtained by substituting iteratively the free variables of $[q]^{\Gamma}$ by their bindings in $[\mathcal{A}]$.

Remark 9.10. The formulas $[\![q]\!]^{\Gamma}$ are fixed point formulas. Conversely, all the fixed point FL-subformulas of $[\mathcal{A}]$ are of the form $[\![q]\!]^{\Gamma}$ where Γ, q is a run-section.

The following proposition will be very useful for further proofs. It shows that contrarily to $[q]^{\Gamma}$, no case analysis on Γ and q is required to figure out the shape of $[\![q]\!]^{\Gamma}$.

Proposition 9.2. *Let $\mathcal{A}$ be an APW and $[\![q]\!]^{\Gamma}$ a FL-suboccurrence of $[\mathcal{A}]$. The top-level connective of $[\![q]\!]^{\Gamma}$ is σX_q. and we have:*

$$[\![q]\!]^{\Gamma} \to \bigvee_{t=(q,a,E)\in\Delta} [a] \wedge \bigwedge_{p\in E} \odot [\![p]\!]^{\Gamma,(q,t)}$$

Using Proposition 9.2, we can derive the following rule, which mimics automata transitions in the proof system $\mu\mathsf{LK}\odot^{\infty}$.

Proposition 9.3. *Let $\mathcal{A}$ be an APW of transition relation Δ, q be state of $\mathcal{A}$ and Γ a run-section. For every set of occurrences Θ, the following rule is derivable using a meager derivation:*

$$\frac{\{[a], \odot\{[\![p]\!]^{\Gamma,(q,t)}\}_{p\in E} \vdash \Theta\}_{t=(q,a,E)\in\Delta}}{[\![q]\!]^{\Gamma} \vdash \Theta}$$

Proof. Since $[\![q]\!]^{\Gamma} \to \bigvee_{t=(q,a,E)\in\Delta} [a] \wedge \bigwedge_{p\in E} \odot [\![p]\!]^{\Gamma,(q,t)}$, one can derive the following:

$$\dfrac{\left\{\dfrac{[a], \{\odot[\![p]\!]^{\Gamma,(q,t)}\}_{p\in E} \vdash \Theta}{[a] \wedge \bigwedge_{p\in E} \odot [\![p]\!]^{\Gamma,(q,t)} \vdash \Theta}\ (\wedge_{\mathsf{l}})\right\}_{t=(q,a,E)\in\Delta}}{\bigvee_{t=(q,a,E)\in\Delta} [a] \wedge \bigwedge_{p\in E} \odot [\![p]\!]^{\Gamma,(q,t)} \vdash \Theta}\ (\vee_{\mathsf{l}})$$
$$\frac{\bigvee_{t=(q,a,E)\in\Delta} [a] \wedge \bigwedge_{p\in E} \odot [\![p]\!]^{\Gamma,(q,t)} \vdash \Theta}{[\![q]\!]^{\Gamma} \vdash \Theta}\ (\sigma_{\mathsf{l}})$$

Notice that this is a meager derivation. □

Now we relate the threads of the formula encoding an automaton $\mathcal{A}$ to the run branches of $\mathcal{A}$. For that, we define the run-branch corresponding to a thread starting from $[A]$ as follows:

Definition 9.12. Let $\mathcal{A}$ be an APW and t be a thread starting from $[\mathcal{A}]$. Let $([\![q_i]\!]^{\Gamma_i})_{i\in\omega}$ be the sequence of the fixed-point occurrences appearing in t. The ***run-branch*** of t is $\rho(t) := (q_i)_{i\in\omega}$.

It is easy to see that the run-branch of a thread is indeed a branch in a run of $\mathcal{A}$.

Proposition 9.4. *Let $\mathcal{A}$ be an APW, c be its priority function. A thread t of $[\mathcal{A}]$ is a ν-thread iff $\rho(t)$ is accepting.*

To prove that result, we first show the following lemma:

Lemma 9.1. *Let $\mathcal{A}$ be an APW automaton and let $\{[\![q_i]\!]^{\Gamma_i}\}_{i\in I}$ be a collection of FL-suboccurrences of $[\mathcal{A}]$. For every $i \in I$, let n_i be the node of the occurrence $[\![q_i]\!]^{\Gamma_i}$ in the graph $\mathcal{G}([\mathcal{A}])$ of $[\mathcal{A}]$.*

Suppose that there is an elementary cycle C of $\mathcal{G}([\mathcal{A}])$ whose fixed-point nodes are the nodes $\{n_i\}_{i\in I}$. Let $k \in I$ be the index such that n_k is the nearest node of C to the root of $\mathcal{G}([\mathcal{A}])$. We have that for every $i \in I$, $c(q_0) \leq c(q_i)$.

Proof. Notice first that for every $i \in I$, the node n_i has a label of the form $\sigma_i X_{q_i}$. Notice also that all the nodes of the cycle C are also nodes of the tree $\tau([\mathcal{A}])$ of the $[\mathcal{A}]$, but in $\tau([\mathcal{A}])$, instead of the back-edge, we have a node n labelled X_{q_k}.

There exist two environments Γ and Δ, such that:

- The subformula of $[\mathcal{A}]$ corresponding to the node n_k is of the form $[q_k]^{\Gamma}$.

- The subformula of $[\mathcal{A}]$ corresponding to the node n is of the form $[q_k]^{\Gamma,\Delta}$.

Due to the side condition of the first case of the encoding, Δ is of the form $(p_1, t_1), \ldots, (p_l, t_l)$ where $p_1 = q_k$, $\{p_j\}_{j\in[1,l]} = \{q_i\}_{i\in I}$ and for every $j \in [1, l]$ we have $c(q_k) \leq c(p_j)$, which concludes the proof. □

Lemma 9.1 generalizes easily to the case where C is an arbitrary cycle. We now prove Proposition 9.4.

Proof of Proposition 9.4. Let $(\llbracket q_i \rrbracket^{\Gamma_i})_{i\in\omega}$ be the sequence of the fixed-point occurrences appearing in t, and let $\rho = \rho(t)$ be the run-branch of t. There is a set of indices I such that $\mathsf{Inf}(t) = \{\overline{\llbracket q_i \rrbracket^{\Gamma_i}}\}_{i\in I}$. Note that $\mathsf{Inf}(\rho) = \{q_i\}_{i\in I}$. Let n_i be the node of $\llbracket q_i \rrbracket^{\Gamma_i}$ in $\mathcal{G}([\mathcal{A}])$. There is a cycle C of $\mathcal{G}([\mathcal{A}])$ such that:

- The cycle C contains all the nodes $\{n_i\}_{i\in I}$.
- There is an index k such that the nearest node of C to the root of $\mathcal{G}([\mathcal{A}])$ is n_k.

By Proposition 2.7, one has: $\mathsf{min}(\mathsf{Inf}(\bar{t})) = \overline{\llbracket q_k \rrbracket^{\Gamma_k}}$. By Lemma 9.1, one has that for every $i \in I$ we have $c(q_k) \leq c(q_i)$. Thus, one has: $\mathsf{min}(c(\mathsf{Inf}(\rho))) = c(q_k)$. Since $c(q_k)$ is even if and only if $\overline{\llbracket q_k \rrbracket^{\Gamma_k}}$ is a ν-formula, we conclude that t is a ν-thread if and only if ρ is accepting. □

Now, we show that the language of an automaton and the models of its encoding are equal.

Proposition 9.5. *For any APW $\mathcal{A}$, $\mathcal{M}([\mathcal{A}]) = \mathcal{L}(\mathcal{A})$.*

Proof. We show that the automaton $\mathcal{A}$ and the formula $[\mathcal{A}]$ have the same language that is $\mathcal{L}(\mathcal{A}) = \mathcal{L}([\mathcal{A}])$. We show first that $\mathcal{L}([\mathcal{A}]) \subseteq \mathcal{L}(\mathcal{A})$.

Let $u \in \mathcal{L}([\mathcal{A}])$ and let β be an accepting branch of $\Pi([\mathcal{A}])$ that induces u. Let $(s_i)_{i\in\omega}$ be the sequence of conclusions of $\odot$ rules in β. Every s_i is of the form $C_i, \odot\{\llbracket q_k \rrbracket^{\Gamma_k}\}_{k\in I_i}$ where $\overline{C_i} = C(u_i)$. This branch induces easily a run ρ of $\mathcal{A}$ over u, where at every level i of ρ, the set of nodes labels is $\{q_k\}_{k\in I_i}$. Every branch of ρ is a run-branch of a thread t of β. Since β is accepting, every thread t of β is a ν-thread, thus by Proposition 9.4 every branch of ρ is accepting, hence ρ is accepting.

We show the other direction $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}([\mathcal{A}])$ by using a similar argument: to every accepting run of $\mathcal{A}$ over a word u, one can find an accepting branch of $\Pi([\mathcal{A}])$ that induces the word u.

By Proposition 9.1, $\mathcal{L}([\mathcal{A}]) = \mathcal{M}([\mathcal{A}])$, which concludes the proof. □

9.3 From formulas to APW automata

In this section, we exploit the intuitions and tools developed in Section 9.1 to extract from every μ-calculus formula φ an APW $\mathcal{A}_\varphi$.

Let us give an outline of the construction of the APW $\mathcal{A}_\varphi$ corresponding to a formula φ. Definition 9.5 suggests that we can see the FL-subformulas of φ as the states of an APW, whose transitions are given by the F-derivations. Indeed, if we consider an F-derivation, as the one shown below:

$$\frac{C_1, \odot\Delta_1 \vdash \quad \ldots \quad C_n, \odot\Delta_n \vdash}{F \vdash} \ (F)$$

we can think of the branching of this F-derivation as a form of non-determinism, meaning that every premise yields a possible transition of the automaton when it is in the state F; and we can think of the commas in each premise as conjunction or universal non-determinism, meaning that if the automaton is in the state F and reads a letter a which satisfies some constraint C_i, then the automaton goes to the states corresponding to the formulas of Δ_i. For example, the derivation $\Pi(\delta)$ of Example 9.3 suggests that the set of states of $\mathcal{A}_\delta$ is $\{\delta\}$ and its transition relation is $\{(\delta, a, \delta), (\delta, b, \delta)\}$, where we recall that $a = \emptyset$ and $b = \{\mathfrak{p}\}$.

To assign priorities to the states of the automaton of a formula in a way that preserves its semantics, one has to look at the formulas hidden in the F-derivations. For instance, if we consider again the derivation $\Pi(\delta)$ of Example 9.3, what makes the leftmost branch of $\Pi(\delta)$ accepting is the formula ψ hidden in the δ-derivation. Even if we want to consider δ as the state of $\mathcal{A}_\varphi$, we want the priority of δ to be that of ψ, at least in the leftmost run. One solution would be, for every F-derivation as above, to set the priority of a formula G appearing in some Δ_i to be the priority of the minimum of the thread linking G to F. The problem is that we may have two different occurrences of G in the F-derivation, linked to F with two threads having different minimal formulas. For instance, in Example 9.3, the minimum of the thread linking δ_ε to δ_{riii} is φ, while the minimum of the thread linking δ_ε to δ_{lii} is ψ. Thus, in $\mathcal{A}_\delta$, one should have two copies of the state δ, one with priority corresponding to φ and the other with priority corresponding to ψ.

More generally, the states of the alternating automaton corresponding to a formula will be pairs of formulas, written as δ^ψ, where δ is morally the state and ψ is a formula containing the priority information.

To make this construction work, we have to ensure that in an F-derivation, every thread linking F to a formula occurrence appearing in a premise of this F-derivation admits a minimal formula. This is what the following proposition shows.

Proposition 9.6. *Let s be a successor of F and $G \in s$. The thread of the F-derivation, linking F to G, has a minimal formula ψ.*

We write $F \overset{s,\psi}{\to} G$ as a shortcut for: "The minimum of the thread linking F to G in the F-derivation is ψ".

Proof. It suffices to notice that the thread t linking F to G is a straight thread. By Proposition 2.5 it admits a minimal formula. □

The following proposition is straightforward. It shows that the minimum of the thread linking F to an occurrence G appearing in one of its successors does not change if we relocate F.

Proposition 9.7. *Let φ_α be an occurrence and $s \in S(\varphi_\alpha)$. We have:*

- *The sequent s is of the form $s = \{(\varphi_i)_{\alpha.\alpha_i}\}_{0\leq i\leq n}$.*
- *For every address β, $s' := \{(\varphi_i)_{\beta.\alpha_i}\}_{0\leq i\leq n} \in S(\varphi_\beta)$,*
- *and $\forall i \leq n$ if $\varphi_\alpha \overset{s,\psi}{\to} (\varphi_i)_{\alpha.\alpha_i}$ then $\varphi_\beta \overset{s,\psi}{\to} (\varphi_i)_{\beta.\alpha_i}$.*

We define in the following, for every formula φ, a function p_φ that assigns to every FL-subformula of φ an integer, in a way that respects the subformula ordering and that is compatible with the nature of fixed-point formulas.

Definition 9.13. Let φ be a formula and let $p_\varphi : \mathsf{FL}(\varphi) \to \omega$ be a function such that:

- For all $\psi \in \mathsf{FL}(\varphi)$, ψ is a ν-formula if and only if $p_\varphi(\psi)$ is even.
- If $\psi \leq \delta$ then $p_\varphi(\psi) \leq p_\varphi(\delta)$.

The function p_φ is not unique but we can fix one arbitrarily.

The function p_φ will be used to assign priorities to the states of the APW corresponding to the formula φ.

Definition 9.14. The ***APW associated to φ, denoted $\mathcal{A}_\varphi$***, is the tuple $(\Sigma, Q, q_I, \Delta_\varphi, c)$ where the set of states is $Q = \{\delta^\gamma \mid \delta, \gamma \in \mathsf{FL}(\varphi)\}$, the initial state is $q_I = \varphi^\varphi$, the priority function is defined by $c(\delta^\gamma) = p_\varphi(\gamma)$ and the transition relation Δ_φ is defined as follows: $(\psi^\delta, a, E) \in \Delta_\varphi$ iff there is $s = (C, \odot\Delta) \in S(\psi_\varepsilon)$ such that $a \models \overline{C}$ and:

$$\gamma^\sigma \in E \qquad \leftrightarrow \qquad \exists(\odot\gamma_\beta) \in s \text{ such that } \psi_\varepsilon \overset{s,\sigma}{\to} \odot\gamma_\beta$$

Remark 9.11. The choice of q_I as φ^φ is arbitrary, one could choose any state of the form φ^δ.

Example 9.8. We consider the formula δ of Example 9.3 and we construct its corresponding automaton $\mathcal{A}_\delta$. We recall that $a = \emptyset$ and $b = \{\mathfrak{p}\}$. The Fischer-Ladner closure of δ is $\mathsf{FL}(\delta) = \{\delta, \varphi, \psi, \odot\delta, \mathfrak{p} \wedge \psi\}$. We fix p_δ to be $p_\delta(\varphi) = 1$, $p_\delta(\psi) = 2$ and $p_\delta(\gamma) = 3$ for every other formula γ of $\mathsf{FL}(\delta)$. The accessible states of $\mathcal{A}_\delta$ are $\{\delta^\delta, \delta^\varphi, \delta^\psi\}$ and its initial state is δ^δ. The coloring function c of $\mathcal{A}_\delta$ is $c(\delta^\varphi) = 1$, $c(\delta^\psi) = 2$ and $c(\delta^\delta) = 3$. Now we construct the transition structure of $\mathcal{A}_\delta$. The δ_ε derivation has two premises: $s_1 = \mathfrak{p}, (\odot\delta)_{lri}$ and $s_2 = (\odot\delta)_{rii}$. We have $\delta_\varepsilon \overset{s_1,\psi}{\to} (\odot\delta)_{lri}$. Since only the letter b satisfies the constraint $\{\mathfrak{p}\}$ of s_1, the transition relation of $\mathcal{A}_\delta$ contains the following transitions: $\{(\delta^\gamma, b, \delta^\psi) \mid \gamma \in \mathsf{FL}(\delta)\}$. We have also that $\delta_\varepsilon \overset{s_2,\varphi}{\to} (\odot\delta)_{rii}$. Since both a and b satisfy the (empty) constraint of s_2, the transition relation of $\mathcal{A}_\delta$ contains also the following transitions: $\{(\delta^\gamma, a, \delta^\varphi), (\delta^\gamma, b, \delta^\varphi) \mid \gamma \in \mathsf{FL}(\delta)\}$.

Proposition 9.8. *We have $\mathcal{L}(\mathcal{A}_\varphi) = \mathcal{M}(\varphi)$.*

Proof. We show that the automaton $\mathcal{A}_\varphi$ and the formula φ have the same language ie. $\mathcal{L}(\mathcal{A}_\varphi) = \mathcal{L}(\varphi)$. We first show that $\mathcal{L}(\varphi) \subseteq \mathcal{L}(\mathcal{A}_\varphi)$. Let $u \in \mathcal{L}(\varphi)$ and let β be an accepting branch of $\Pi(\varphi)$ that induces u. Let $(s_i)_{i\in\omega}$ be the sequence of conclusions of $\odot$ rules in β. For every $i \in \omega$, s_i is of the form $C_i, \odot\Delta_i$ where $\overline{C_i}$ is a constraint. We construct in the following a run ρ of $\mathcal{A}_\varphi$ over u, level by level, starting from the root. In this construction, we keep the following invariant: for every $i \in \omega$, if $\Delta_i = \{(\delta_j)_{\alpha_j}\}_{1\leq j\leq n}$, then there exists a set of formulas $\{\psi_j\}_{1\leq j\leq n}$ such that the set of nodes labels at level i in ρ is $\{\delta_j^{\psi_j}\}_{1\leq j\leq n}$.

Level 1: We label the root of ρ by φ^φ. The sequent Δ_1 is of the form $\{(\delta_j)_{\alpha_j}\}_{1\leq j\leq n}$. For every $j \leq n$, there exists ψ_j such that $\varphi_\varepsilon \overset{s_1,\psi_j}{\to} (\odot\delta_j)_{\alpha_j}$, we add then to the root a son labelled $\delta_j^{\psi_j}$. Our invariant is satisfied for level 1.

Level i $\to$ Level i+1: If $\Delta_i = ((\delta_k)_{\alpha_k})_{1\leq k\leq n}$ then s_{i+1} is of the form $s_{i+1} = \bigcup_{1\leq k\leq n} v_k$, where $v_k \in S((\delta_k)_{\alpha_k})$, for every $k \leq n$. Let w be a node of level i, labelled $\delta_k^{\psi_k}$. For every

occurrence $(\odot\gamma)_\alpha$ in v_k, there exists a formula ψ such that $(\delta_k)_{\alpha_k} \overset{v_k,\psi}{\to} (\odot\gamma)_\alpha$, we add to w a son labelled γ^ψ. Here again, our invariant is satisfied for level $i+1$.

The run ρ is valid. Indeed, let $b = (\delta_i^{\psi_i})_{i\in\omega}$ be a branch of ρ. We will show that b is valid using the validity of the branch β of $\Pi(\varphi)$. By construction one has that $\delta_0^{\psi_0} = \varphi^\varphi$ and there exists a sequence of addresses $(\alpha_i)_{i\in\omega}$ such that for every $i \in \omega$ we have $(\delta_i)_{\alpha_i} \in \Delta_i$. Moreover, the thread linking $(\delta_i)_{\alpha_i}$ to $(\delta_{i+1})_{\alpha_{i+1}}$ in $\Pi(\varphi)$, that we denote t_i, satisfies the following property: $\mathsf{min}(\overline{t_i}) = \psi_{i+1}$. Let $t = t_0t_1\ldots$ be the thread of $\Pi(\varphi)$ obtained by the concatenation of the threads $\{t_i\}_{i\in\omega}$. Let $k, l \in \omega$ such that the sub-thread $t_k \ldots t_l$ of t contains all the elements of $\mathsf{Inf}(\overline{t})$. One has also that the sub-sequence $\delta_k^{\psi_k}, \ldots, \delta_{l+1}^{\psi_{l+1}}$ of b contains all the elements of $\mathsf{Inf}(b)$. One has:

$$\begin{array}{rcl} \mathsf{min}(\mathsf{Inf}(\overline{t})) & = & \mathsf{min}(\overline{t_k}\ldots\overline{t_l}) \\ & = & \mathsf{min}(\mathsf{min}(\overline{t_i}))_{k\le i\le l} \\ & = & \mathsf{min}(\psi_i)_{k\le i\le l} \end{array}$$

Since the branch β is accepting, $\mathsf{min}(\psi_i)_{k\le i\le l}$ is a ν-formula. Moreover, if c is the priority function of $\mathcal{A}_\varphi$ and p_φ the auxiliary function used to define it (Definition 9.13), then we have:

$$\begin{array}{rcl} \mathsf{min}(c(\mathsf{Inf}(b))) & = & \mathsf{min}(c(\delta_k^{\psi_k}), \ldots, c(\delta_{l+1}^{\psi_{l+1}})) \\ & = & \mathsf{min}(p_\varphi(\psi_k), \ldots, p_\varphi(\psi_{l+1})) \\ & \overset{\dagger}{=} & p_\varphi(\mathsf{min}(\psi_i)_{k\le i\le l}) \end{array}$$

(†) Since the function p_φ is monotonic.

Since $\mathsf{min}(\psi_i)_{k\le i\le l}$ is a ν-formula, one has that $p_\varphi(\mathsf{min}(\psi_i)_{k\le i\le l})$ is even, hence the branch b is valid. We conclude that ρ is valid and $u \in \mathcal{L}(\mathcal{A}_\varphi)$.

We show the other direction $\mathcal{L}(\mathcal{A}_\varphi) \subseteq \mathcal{L}(\varphi)$ using a similar argument: to every valid run of $\mathcal{A}_\varphi$ over a word u, we can find an accepting branch of $\Pi(\varphi)$ that induces u.

By Proposition 9.1, $\mathcal{L}(\varphi) = \mathcal{M}(\varphi)$, which concludes the proof. □

9.4 A formula and the encoding of its automaton are equivalent in $\mu\mathsf{LK}\odot$

We show that there is a proof of $[\mathcal{A}_\varphi] \vdash \varphi$ in $\mu\mathsf{LK}\odot$. The idea is to construct a meager proof of this sequent in $\mu\mathsf{LK}\odot^\omega$, and using Proposition 7.2, to transform it into a proof in $\mu\mathsf{LK}\odot$. Since φ^φ is the initial state of $\mathcal{A}_\varphi$, we have that $[\mathcal{A}_\varphi]_\varepsilon = [\![\varphi^\varphi]\!]^\emptyset$, hence our goal is to show that the sequent $[\![\varphi^\varphi]\!]^\emptyset \vdash \varphi_\varepsilon$ has a meager $\mu\mathsf{LK}\odot^\omega$ proof. To get this result, we need to generalize our statement and to look for proofs of sequents having the form $[\![\psi^\delta]\!]^\Theta \vdash \psi_\alpha$, where ψ^δ is a state of $\mathcal{A}_\varphi$, Θ a run-section and α an address. Using Proposition 9.3, we decompose the left-hand side of such sequents and get the following derivation:

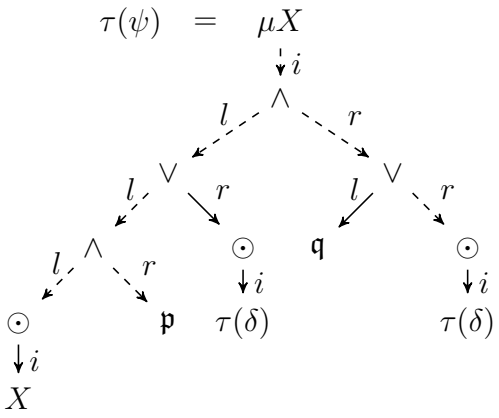
$$\frac{\{[a], \{\odot[\![\gamma^\sigma]\!]^{\Theta,(\psi^\delta,t)}\}_{\gamma^\sigma\in E} \vdash \psi_\alpha\}_{t=(\psi^\delta,a,E)\in\Delta_\varphi}}{[\![\psi^\delta]\!]^\Theta \vdash \psi_\alpha}$$

Each premise of this derivation corresponds to a transition in Δ_φ, and by construction of Δ_φ, every such transition corresponds to a successor $s \in S(\psi_\varepsilon)$. To get a proof of such premise which is meager, we have to apply weakenings at some points, so that when unfolding the

fixed points, their context will be empty. The choice of these weakenings will be guided by this successor s in a sense that we will clarify in the following. But first, let us look closely into the notion of the successor of a formula.

If s is the successor of ψ_ε, then for every $F \in s$, the address of F induces a path in the tree of ψ. When we collect all the paths induced by the formula occurrences of s, we get a sub-tree $\tau(s)$ of $\tau(\psi)$, as illustrated by the following example.

Example 9.9. Let $\psi = \mu X.((\odot X \wedge \mathfrak{p}) \vee \odot\delta) \wedge (\mathfrak{q} \vee \odot\delta)$ where $\delta = \nu Y.\, \odot\, Y$. Let $s = \mathfrak{p}, (\odot\psi)_\alpha, (\odot\delta)_\beta$ be the successor of ψ_ε where $\alpha = illl$ and $\beta = irr$. The paths of $\tau(\psi)$ induced by the formula occurrences of s form a tree $\tau(s)$ indicated by the dashed lines below:



Any node n of $\tau(s)$ satisfies the following properties:

- If n is labelled $\odot$, the son of n does not belong to $\tau(s)$.
- If n is labelled $\wedge$, the two sons of n belong also to $\tau(s)$.
- If n is labelled $\vee$, exactly one son of n belongs to $\tau(s)$.

A successor $s \in S(F)$ induces a function that chooses, for every suboccurrence of F which is a disjunction, one of its disjuncts; we call this function C_s.

Definition 9.15. Let $s = \{(\delta_i)_{\alpha_i}\}_{1 \le i \le n}$ be the successor of an occurrence $F = \varphi_\alpha$. For all α_i, there is p_i st. $\alpha_i = \alpha.p_i$.

The ***choice*** of s is a partial function on occurrences defined as follows. For all i, and for every address $a \sqsubseteq p_i$ such that the node of $\tau(\varphi)$ at the address a is labelled $\vee$, we set:

$$\begin{aligned} C_s((\varphi \vee \psi)_a) &= \varphi_{a.l} \quad \text{if } a.l \sqsubseteq p_i, \\ &= \psi_{a.r} \quad \text{otherwise.} \end{aligned}$$

Example 9.10. In Example 9.9, we have: $C_s((\odot\psi \wedge \mathfrak{p}) \vee \odot\delta)_{il}) = (\odot X \wedge \mathfrak{p})_{ill}$ and $C_s((\mathfrak{q} \vee \odot\delta)_{ir}) = (\odot\delta)_{irr}$.

Using this function, we define the derivation $\Pi_s(\Gamma \vdash F)$. In this derivation of conclusion $\Gamma \vdash F$, we apply only right rules, and whenever we meet a right formula which is a disjunction, we apply $(\vee_\mathsf{r})$ rule followed immediately by a weakening, that keeps only the formula chosen by C_s. Doing so, we obtain a derivation which is meager, since we keep the invariant that we have exactly one right formula along the derivation.

Definition 9.16. Let F be an occurrence, $s \in S(F)$ and Γ be a set of occurrences. The derivation $\Pi_s(\Gamma \vdash F)$ is the finite $\mu\mathsf{LK}\odot^{\infty}$ derivation of conclusion $\Gamma \vdash F$ obtained by applying only the following rules:

$$\frac{\Gamma \vdash F \qquad \Gamma \vdash G}{\Gamma \vdash F \wedge G}\ (\wedge_{\mathsf{r}}) \qquad \frac{\Gamma \vdash F[\sigma X.F/X]}{\Gamma \vdash \sigma X.F}\ (\sigma_{\mathsf{r}}) \qquad \frac{\dfrac{\Gamma \vdash C_s(F \vee G)}{\Gamma \vdash F, G}\ (\mathsf{W}_{\mathsf{r}})}{\Gamma \vdash F \vee G}\ (\vee_{\mathsf{r}})$$

Example 9.11. We show in the following the derivation $\Pi_s(\vdash \psi_\varepsilon)$, where s and ψ are defined in Example 9.9.

$$\dfrac{\dfrac{\dfrac{\dfrac{\vdash (\odot\psi)_{ill} \qquad \vdash \mathfrak{p}}{\vdash (\odot\psi \wedge \mathfrak{p})_{ill}}\ (\wedge_{\mathsf{r}})}{\vdash ((\odot\psi \wedge \mathfrak{p}) \vee \odot\delta))_{il}}\ (\vee_{\mathsf{r}}),(\mathsf{W}_{\mathsf{r}}) \qquad \dfrac{\vdash (\odot\delta)_{irr}}{\vdash (\mathfrak{q} \vee \odot\delta)_{ir}}\ (\vee_{\mathsf{r}}),(\mathsf{W}_{\mathsf{r}})}{\vdash ((\odot\psi \wedge \mathfrak{p}) \vee \odot\delta) \wedge (\mathfrak{q} \vee \odot\delta))_i}\ (\wedge_{\mathsf{r}})}{\vdash \psi_\varepsilon}\ (\mu_{\mathsf{r}})$$

The following proposition is straightforward.

Proposition 9.9. *Let F be an occurrence, $s \in S(F)$ and Γ be a set of occurrences.*

- *$\Pi_s(\Gamma \vdash F)$ is a meager derivation.*
- *The premises of $\Pi_s(\Gamma \vdash F)$ are $\{\Gamma \vdash G \mid G \in s\}$.*
- *If $\Gamma \vdash G$ is a premise of $\Pi_s(\Gamma \vdash F)$ and if $F \overset{s,\psi}{\to} G$ then the minimum of the thread linking F to G in $\Pi_s(\Gamma \vdash F)$ is ψ.*

Proposition 9.10. *Let φ be a formula and ψ^δ be a state of $\mathcal{A}_\varphi$. The following rule is derivable in $\mu\mathsf{LK}\odot^{\omega}$ using a meager derivation:*

$$\frac{\left\{[\![\gamma^\sigma]\!]^{\Theta,(\psi^\delta,t)} \vdash \gamma_{\alpha.\beta}\right\}_{s\in S(\psi_\varepsilon),\gamma_\beta\in s,\psi_\varepsilon \overset{s,\sigma}{\to} \odot(\gamma)_\beta}}{[\![\psi^\delta]\!]^\Theta \vdash \psi_\alpha}\ (\circ)$$

And the minimum of the thread linking ψ_α to $\gamma_{\alpha.\beta}$ is σ.

Proof. We already noticed that the following rule is derivable:

$$\frac{\left\{[a], \{\odot[\![\gamma^\sigma]\!]^{\Theta,(\psi^\delta,t)}\}_{\gamma^\sigma\in E} \vdash \psi_\alpha\right\}_{t=(\psi^\delta,a,E)\in\Delta_\varphi}}{[\![\psi^\delta]\!]^\Theta \vdash \psi_\alpha}$$

By definition of Δ_φ, we have that $(\psi^\delta, a, E) \in \Delta_\varphi$ iff there is $s = (C, \odot\Delta) \in S(\psi_\varepsilon)$ s.t. $a \models \overline{C}$ and if the following holds:

$$\gamma^\sigma \in E \quad \leftrightarrow \quad \exists(\odot\gamma_\beta) \in s \text{ such that } \psi_\varepsilon \overset{s,\sigma}{\to} \odot\gamma_\beta$$

To justify a premise corresponding to a transition $t = (\psi^\delta, a, E)$, we set $\Gamma = [a], \{\odot[\![\gamma^\sigma]\!]^{\Theta,(\psi^\delta,t)}\}_{\gamma^\sigma\in E}$ and apply the derivation $\Pi_s(\Gamma \vdash \psi_\alpha)$:

$$\frac{\{\Gamma \vdash \mathfrak{p}_\eta\}_{\mathfrak{p}_\eta\in C} \qquad \{\Gamma \vdash \odot(\gamma)_{\alpha.\beta}\}_{\odot(\gamma)_\beta\in s}}{\Gamma \vdash \psi_\alpha}\ (\Pi_s(\Gamma \vdash \psi_\alpha))$$

To justify a premise $\Gamma \vdash \mathfrak{p}_\eta$, we notice that since $a \models \overline{C}$, we have $\overline{C} \subseteq a$, thus $[a]$ is of the form $[a] = \mathfrak{p} \wedge G$. We then apply the following derivation:

$$\dfrac{\dfrac{}{[a] \vdash F}\ (\wedge_{\mathsf{l}}),\ (\mathsf{Ax})}{\Gamma \vdash F}\ (\mathsf{W}_{\mathsf{l}})$$

Now we want to justify a premise $\Gamma \vdash \odot(\gamma)_{\alpha.\beta}$. If $\psi_\varepsilon \xrightarrow{s,\sigma} \gamma_\beta$, then $\gamma^\sigma \in E$, we can thus apply the following:

$$\dfrac{\dfrac{[\![\gamma^\sigma]\!]^{\Theta,(\psi^\delta,t)} \vdash \gamma_{\alpha.\beta}}{\odot[\![\gamma^\sigma]\!]^{\Theta,(\psi^\delta,t)} \vdash \odot(\gamma)_{\alpha.\beta}}\ (\odot)}{\Gamma \vdash \odot(\gamma)_{\alpha.\beta}}\ (\mathsf{W}_{\mathsf{l}})$$

The obtained derivation is meager and by Proposition 9.9, the minimum of the thread linking ψ_α to $\gamma_{\alpha.\beta}$ is σ. □

The premises of the derived rule $(\circ)$ have the same shape as its conclusion. If we iterate this derivation starting from $[\mathcal{A}_\varphi] \vdash \varphi$ (which is $[\![\varphi^\varphi]\!]^\emptyset \vdash \varphi_\varepsilon$) we obtain a meager $\mu\mathsf{LK}\odot^\infty$ pre-proof. We show in the following that this pre-proof is actually a proof. Moroever, this derivation is trivially regular, thus it can be represented by a circular $\mu\mathsf{LK}\odot^\omega$ pre-proof. Using Proposition 7.2, we can transform it into a $\mu\mathsf{LK}\odot$ proof of $[\mathcal{A}_\varphi] \vdash \varphi$.

Theorem 9.1. *We can construct a $\mu\mathsf{LK}\odot$ proof of $[\mathcal{A}_\varphi] \vdash \varphi$.*

Proof. Let c be the priority function of $\mathcal{A}_\varphi$ and p_φ (Definition 9.13) the function used to define c. Recall that $[\mathcal{A}_\varphi]_\varepsilon = [\![\varphi^\varphi]\!]^\emptyset$. Let π be the $\mu\mathsf{LK}\odot^\infty$ pre-proof obtained by applying coinductively the derivation $(\circ)$ of Proposition 9.10. It is not difficult to see that π is a meager. We show now that π is $\mu\mathsf{LK}\odot^\infty$ proof, *i.e.*, that is satisfies the validity condition.

Let γ be an infinite branch of π and $([\![\psi_i^{\delta_i}]\!]^{\Theta_i} \vdash (\psi_i)_{\alpha_i})_{i\in\omega}$ be the sequence made of the conclusions of the derivation $\circ$. In γ, there is exactly one right thread t_r and one left thread t_l. The thread t_l is a thread of $[\mathcal{A}]$. Let $\rho(t_l)$ be the run-branch of t_l (Definition 9.12). Let $v = (\delta_i)_{i\in\omega}$, we have that $\psi^\delta \in \mathsf{Inf}(\rho(t_l)) \Leftrightarrow \delta \in \mathsf{Inf}(v)$. Hence we have that:

$$\begin{array}{rcl} & & \mathsf{min}(\{c(\psi^\delta) \mid \psi^\delta \in \mathsf{Inf}(\rho(t_l))\}) \\ (\star) & = & \mathsf{min}(\{p_\varphi(\delta) \mid \psi^\delta \in \mathsf{Inf}(\rho(t_l))\}) \\ & = & \mathsf{min}(\{p_\varphi(\delta) \mid \delta \in \mathsf{Inf}(v)\}) \end{array}$$

On the other hand, since the minimum of the thread linking ψ_i to ψ_{i+1} in γ is δ_i, we have:

$$\begin{array}{rcl} & & \mathsf{min}(\{\psi \mid \psi \in \mathsf{Inf}(\overline{t_r})\}) \\ (\dagger) & = & \mathsf{min}(\{\delta \mid \delta \in \mathsf{Inf}(v)\}) \end{array}$$

There are two possible cases: either t_l is a μ-thread, thus γ is valid; or t_l is a ν-thread then by Proposition 9.4 the run-branch $\rho(t_l)$ is accepting, then, by $(\star)$, $\mathsf{min}(\{p_\varphi(\delta) \mid \delta \in \mathsf{Inf}(v)\})$ is even. By definition of p_φ, $\mathsf{min}(\{\delta \mid \delta \in \mathsf{Inf}(v)\})$ is a ν-formula, thus by $(\dagger)$, t_r is a ν-thread, which means that γ is valid.

It is not difficult to see that π is regular, thus it can be represented by a circular $\mu\mathsf{LK}\odot^\omega$ proof θ. Using Proposition 7.2, we can transform θ into a $\mu\mathsf{LK}\odot$ proof of $[\mathcal{A}_\varphi] \vdash \varphi$. □

Theorem 9.1 is the step I of our proof of completeness. Actually, we can show also that $\varphi \vdash_{\mu\mathsf{LK}\odot} [\mathcal{A}_\varphi]$ in the same way.

Chapter 10

Alternation elimination and parity simplification

We have seen in Chapter 8 that APW, NPW and NBW are three equivalent models of automata. This result is usually shown by exhibiting automata transformations that convert an automaton of a given model into an automaton in the other models, by preserving the language. In this chapter, we recall the automata transformations that turn an APW into a NPW, and a NPW into a NBW, then we show that the transformations provide, via the encoding, provably equivalent formulas. Doing so wo prove Steps II and III of our proof of completeness.

10.1 From APW to NPW

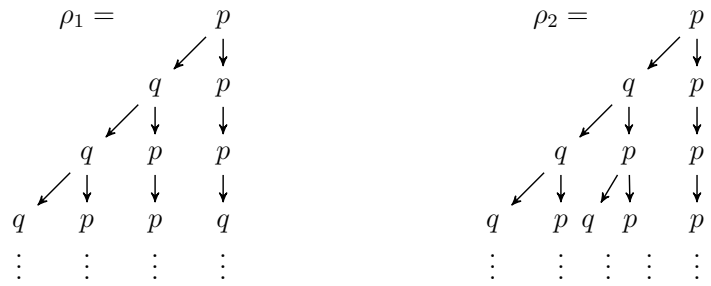
We recall the construction that transforms an APW $\mathcal{A}$ into an NPW $\mathcal{P}$ such that $\mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{A})$. In a second step, we show that the sequent $[\mathcal{P}] \vdash [\mathcal{A}]$ in provable $\mu\mathsf{LK}\odot$.

10.1.1 Non-determinization

We recall in this section the operation of *non-determinization*, that transforms an APW into a NPW with the same language. In [Tho97] this construction is presented for automata over trees, we specialize it here for automata over words. We fix an APW $\mathcal{A} = (\Sigma, Q, \Delta, q_I, c)$.

The key ingredient to non-determinize APW is to simplify their runs. Runs of APW may be *non-uniform* in the sense that two nodes at the same level and labelled with the same state, may have different sets of successors. Let us consider the following example.

Example 10.1. Let $\mathcal{A} = (\{a\}, Q, \Delta, p, c)$ be the APW where $Q = \{p, q\}$, $\Delta = \{t_1, t_2, t_3\}$ where $t_1 = (p, a, \{p\})$, $t_2 = (p, a, \{p, q\})$, $t_3 = (q, a, \{p, q\})$ and $c(p) = 1, c(q) = 2$. ρ_1 and ρ_2 are the beginning of two runs of $\mathcal{A}$ over a^ω.



For instance, in the run ρ_2 of Example 10.1, the two nodes at level 2 and labelled p have as sets of successors $\{p\}$ and $\{p, q\}$ respectively. On the contrary, we call *uniform* a run such that: all nodes at the same level and with the same labels have the same set of successors. For example, the beginning of the run ρ_1 is uniform. Uniform runs are sufficient to compute the languages of APW [Tho97]: $u \in \mathcal{L}(A)$ iff there is a valid uniform run of $\mathcal{A}$ over u.

In a uniform run, every level can be described by a function that maps a state to the set of its successors: uniform runs can be described by sequences of such functions. For instance, the run ρ_1 of Example 10.1 can be seen as a sequence $f_0 f_1 f_2 \dots$ where $f_0(p) = \{p, q\}$, $f_1(p) = \{p, q\}, f_1(q) = q$, $f_2(p) = \{p, q\}, f_2(q) = q$, etc. We call these functions ***choice functions***. The interest of using uniform runs instead of arbitrary runs is that we went from runs having a tree structure to runs with a structure of words (sequences of choice functions), that we will be able to synchronize with Σ-words.

Definition 10.1. A ***choice function*** is a function $\sigma : Q \to 2^Q$. The set of choice functions is denoted $\mathcal{S}$. Notice that $\mathcal{S}$ is finite. The ***auxiliary alphabet*** of $\mathcal{A}$ is $\Sigma^{aux} = \{(a, \sigma) \in \Sigma \times \mathcal{S} \mid (p, a, \sigma(p)) \in \Delta\}$.

Two sequences can be extracted from a word U over Σ^{aux}: the first is a word u over Σ obtained by projection on the first components of U, and the second is a sequence of choice functions obtained by projection on the second components of U. This latter sequence describes a uniform run of $\mathcal{A}$ over u.

Definition 10.2. Let $U = ((a_i, \sigma_i))_{i \in \omega} \in (\Sigma^{aux})^\omega$. The ***run associated*** to U is a labelled tree such that: the root is labelled q_I; and for every node v at level n, if q is the label of v then each son of v is labelled with one of the sates of $\sigma_n(q)$. Notice that τ is a uniform run of U over the word $(a_i)_{i \in \omega}$.

Definition 10.3. The language $L \subseteq (\Sigma^{aux})^\omega$ is the ***auxiliary language*** of $\mathcal{A}$ iff $\forall U \in L$, the run associated to U is accepting (*w.r.t.* the acceptance condition of $\mathcal{A}$). We denoted it by $\mathcal{L}^{aux}(\mathcal{A})$.

The idea of the non-determinization is to exhibit a deterministic parity automaton (DPW) recognizing the auxiliary language of $\mathcal{A}$. Intuitively, this is possible since all the information of alternation is explicit in the Σ^{aux}-words, hence we do not need an alternating automaton to recognize it. Once this is done, we erase the choice functions from the transition structure of this automaton, to get an NPW recognizing the language of $\mathcal{A}$.

Proposition 10.1. *For every APW we can construct effectively a NPW recognizing the same language.*

Proof. Let $\mathcal{A} = (\Sigma, Q, \Delta, q_I, c)$ be an APW. We construct a NPW recognizing the language of $\mathcal{A}$ in two steps.

I. Construct a DPW $\mathcal{D} = (\Sigma^{aux}, Q^d, q_I^d, \Delta^d, c^d)$ such that $\mathcal{L}(\mathcal{D}) = \mathcal{L}^{aux}(\mathcal{A})$. Even though the details of the construction of the automaton $\mathcal{D}$ are not needed for the proof of completeness, we recall them below:

a) **Construct an APW recognizing $\mathcal{L}^{aux}(\mathcal{A})$.**
Let $\mathcal{A}_1 = (\Gamma, Q, \Delta_1, q_I, c)$ be the APW where:

$$\Delta_1 = \{(p, (a, \sigma), E) \mid (p, a, E) \in \Delta \text{ and } \sigma(p) = E\}$$

The language of $\mathcal{A}_1$ is $\mathcal{L}^{aux}(\mathcal{A})$. Notice that for every Σ^{aux}-word, there is only one possible run over $\mathcal{A}_1$.

b) **Construct a NPW recognizing the complement of $\mathcal{L}^{aux}(\mathcal{A})$.**
Let $\mathcal{A}_2 = (\Gamma, Q, \Delta_2, q_I, c_2)$ be the APW where $c_2(q) = c(q) + 1$ and:

$$\Delta_2 = \{(p, A, q) \mid \exists E, (p, A, E \cup \{q\}) \in \Delta_1\}.$$

The language of $\mathcal{A}_2$ is the complement of $\mathcal{L}^{aux}(\mathcal{A})$. Indeed, R is a run of $\mathcal{A}_2$ over U iff R is a path in the unique run of $\mathcal{A}_1$ over U. Since we shifted the priorities in $\mathcal{A}_2$, R is a valid run of $\mathcal{A}_2$ over U iff R seen as a path in the run of $\mathcal{A}_1$ over U is not valid. Thus, $U \in \mathcal{L}(\mathcal{A}_2)$ iff $U \notin \mathcal{L}(\mathcal{A}_1)$.

c) **Construct a DPW recognizing the complement of $\mathcal{L}^{aux}(\mathcal{A})$.**
Determinize $\mathcal{A}_2$ to get a *DPW* $\mathcal{A}_3 = (\Gamma, Q_3, \Delta_3, p_I, c_3)$, using a determinization technique (Safra's construction for example).

d) **Construct a DPW recognizing $\mathcal{L}^{aux}(\mathcal{A})$.**
Let $\mathcal{A}_4 = (\Gamma, Q_3, \Delta_3, p_I, c_4)$ where $c_4(q) = c_3(q) + 1$. One has $\mathcal{L}(\mathcal{A}_4) = \mathcal{L}^{aux}(\mathcal{A})$.

II. Let $\mathcal{P} = (\Sigma, Q^d, q_I^d, \Delta', c^d)$ where $\Delta' = \{(d, a, d') \mid \exists \sigma, (d, (a, \sigma), d') \in \Delta^d\}$. One has $\mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{A})$.

□

10.1.2 Non-determinization in the logic

Let $\mathcal{A}$ be an APW automaton and $\mathcal{P}$ be the NPW obtained in the proof of Proposition 10.1. In this section we show that $[\mathcal{P}] \vdash [\mathcal{A}]$ has a proof in $\mu\mathsf{LK}\odot$. For that purpose, we build a meager $\mu\mathsf{LK}\odot^{\omega}$ proof of this sequent, which gives us a $\mu\mathsf{LK}\odot$ proof by Proposition 7.2. We use in this section the notations of the proof of Proposition 10.1. Since q_I and q_I^d are the initial states of $\mathcal{A}$ and $\mathcal{P}$ respectively, we have that $[\mathcal{A}]_\varepsilon = [\![q_I]\!]^\emptyset$ and $[\mathcal{P}]_\varepsilon = [\![q_I^d]\!]^\emptyset$. Our goal is then to show that $[\![q_I^d]\!]^\emptyset \vdash [\![q_I]\!]^\emptyset$ is provable with a meager $\mu\mathsf{LK}\odot^{\omega}$ proof. To do so, we need to generalize this statement, and to look for proofs of sequents having the form $[\![d]\!]^\Theta \vdash [\![q]\!]^\Upsilon$, where d and Θ (resp. q and Υ) are a state and a run-section for $\mathcal{P}$ (resp. $\mathcal{A}$). This is what we do in the following lemma:

Lemma 10.1. *The following rule is derivable in $\mu\mathsf{LK}\odot^{\infty}$ using a finite meager derivation:*

$$\frac{\{[\![d']\!]^{\Theta,(d,t)} \vdash [\![q']\!]^{\Upsilon,(q,T)}\}_{(d,(a,\sigma),d')\in\Delta^d, q'\in\sigma(q)}}{[\![d]\!]^{\Theta} \vdash [\![q]\!]^{\Upsilon}} \ (\rightarrow)$$

Where $t = (d, a, d')$ and $T = (q, a, \sigma(q))$.

Proof. We have $t = (d, a, d') \in \Delta'$ iff $\exists (a, \sigma) \in \Sigma^{aux}$ such that $(d, (a, \sigma), d') \in \Delta^d$. Using Proposition 9.3, we can derive the following, where $t = (d, a, d')$:

$$\frac{\{[a], \odot[\![d']\!]^{\Theta,(d,t)} \vdash [\![q]\!]^{\Upsilon}\}_{(q,(a,\sigma),d')\in\Delta^d}}{[\![d]\!]^{\Theta} \vdash [\![q]\!]^{\Upsilon}}$$

Now we have to justify every premise of this derivation corresponding to a transition $(q, (a, \sigma), d') \in \Delta^d$. By Proposition 9.2:

$$[\![q]\!]^{\Upsilon} \quad \rightarrow \quad \bigvee_{a\in\Sigma,(q,a,E)\in\Delta} [a] \wedge \bigwedge_{q'\in E} \odot [\![q']\!]^{\Upsilon,q}$$

Since $(a, \sigma) \in \Sigma^{aux}$, then $T := (q, a, \sigma(q)) \in \Delta$. If we set $\Gamma := [a], \odot[\![d']\!]^{\Theta,(d,t)}$, we can the derive the following:

$$\dfrac{\left\{ \dfrac{[\![d']\!]^{\Theta,(d,t)} \vdash [\![q']\!]^{\Upsilon,(q,T)}}{\Gamma \vdash \odot[\![q']\!]^{\Upsilon,(q,T)}} \ (\odot) \right\}_{q'\in\sigma(q)} \qquad \dfrac{}{\Gamma \vdash [a]} \ (\mathsf{Ax})}{\Gamma \vdash [a] \wedge \bigwedge_{q'\in\sigma(q)} \odot [\![q']\!]^{\Upsilon,(q,T)}} \ (\wedge_{\mathsf{r}})$$
$$\dfrac{\Gamma \vdash [a] \wedge \bigwedge_{q'\in\sigma(q)} \odot [\![q']\!]^{\Upsilon,(q,T)}}{\Gamma \vdash [\![q]\!]^{\Upsilon}} \ (\sigma_{\mathsf{r}}),\ (\vee_{\mathsf{r}}),(\mathsf{W}_{\mathsf{r}})$$

□

The premises of the derivation $(\rightarrow)$ have the same shape as its conclusion. If we iterate coinductively $(\rightarrow)$ starting from $[\mathcal{P}] \vdash [\mathcal{A}]$, which is $[\![q_I^d]\!]^{\emptyset} \vdash [\![q_I]\!]^{\emptyset}$, we get a meager $\mu\mathsf{LK}\odot^{\omega}$ pre-proof. We show that this pre-proof is actually a $\mu\mathsf{LK}\odot^{\omega}$ proof.

Proposition 10.2. *Let π be the $\mu\mathsf{LK}\odot^{\infty}$ pre-proof of conclusion $[\mathcal{P}] \vdash [\mathcal{A}]$ obtained by applying coinductively the rule $(\rightarrow)$. We have that π is a meager and regular $\mu\mathsf{LK}\odot^{\infty}$ proof.*

Proof. It is easy to see that π is regular. Let us show that it is valid. Let β be a branch of π and $([\![d_i]\!]^{\Theta_i} \vdash [\![q_i]\!]^{\Upsilon_i})_{i\in\omega}$ be the sequence made of the conclusions of the derivation $(\rightarrow)$ in β. The branch β has only one infinite right thread t_r and one infinite left thread t_l. The run-branch of t_l (Definition 9.12) is $\rho_l = (d_i)_{i\in\omega}$ and the run-branch of t_r is $\rho_r = (q_i)_{i\in\omega}$. For all $i \in \omega$, $\exists (a_i, \sigma_i) \in \Sigma^{aux}$ such that $(d_i, (a_i, \sigma_i), d_{i+1}) \in \Delta^d$, we set $U = ((a_i, \sigma_i))_{i\in\omega}$ and $u = (a_i)_{i\in\omega}$. The sequence ρ_l is a run of $\mathcal{P}$ over u; and it is also the run of the DPW $\mathcal{D}$ (proof of Proposition 10.1) over U, since $\mathcal{P}$ and $\mathcal{D}$ have the same set of states. Let R be the run of $\mathcal{A}$ associated to the Σ^{aux} word U. The sequence ρ_r is a branch of R. There are two possible cases:

- If $U \notin \mathcal{L}^{aux}(\mathcal{A})$. Then the run ρ_l of $\mathcal{D}$ is not accepting. Hence ρ_l seen as a run of $\mathcal{P}$ is also not accepting, since $\mathcal{P}$ and $\mathcal{D}$ have the same priority function. By Proposition 9.4, t_l is a μ-thread, hence β is valid.
- If $U \in \mathcal{L}^{aux}(\mathcal{A})$. Then the run R associated to U is accepting. In particular, its branch ρ_r is accepting. By Proposition 9.4, t_r is a ν-thread, hence β is valid.

□

Using Proposition 7.2, we get the following theorem, which is step II of our completeness proof. $[\mathcal{A}] \vdash_{\mu\mathsf{LK}\odot} [\mathcal{P}]$ can be shown in the same way.

Theorem 10.1. *One can construct a $\mu\mathsf{LK}\odot$ proof of $[\mathcal{P}] \vdash [\mathcal{A}]$.*

10.2 From NPW to NBW

We recall the construction that transforms an NPW $\mathcal{P}$ into an NBW $\mathcal{B}$ such that $\mathcal{L}(\mathcal{B}) = \mathcal{L}(\mathcal{P})$. In a second step, we show that the sequent $[\mathcal{B}] \vdash [\mathcal{P}]$ is provable in $\mu\mathsf{LK}\odot$.

10.2.1 Parity simplification

Proposition 10.3. *For very NPW we can construct effectively a NBW recognizing the same language.*

Proof. Let $\mathcal{P} = (\Sigma, Q, \Delta, q_I, c)$ be a NPW and let Q_{ev} be the set of its even states. The idea is to create for every even state p, a copy of the automaton $\mathcal{P}$ where the state p will be accepting and where all the states with smaller priority will be dropped. We keep also a copy of $\mathcal{P}$ where no state is accepting. A run will stay in this copy for some time and will choose one of the copies where an even state is accepting. Formally, let $\mathcal{B} = (\Sigma, Q_\bot \cup \bigcup_{p\in Q_{ev}} Q_p, (q_I, \bot), \Delta_\bot \cup \Delta_t \cup \bigcup_{p\in Q_{ev}} \Delta_p, F)$ be the NBW such that $\forall i \in \{\bot\} \cup Q_{ev}$, $Q_i = \{(q, i) \mid q \in Q\}$. The relations $\Delta_\bot, \Delta_t$ and Δ_p where $p \in Q_{ev}$ are defined by:

$$\begin{aligned}
\Delta_\bot &= \{((q,\bot), a, (r,\bot)) \mid (q,a,r) \in \Delta\} \\
\Delta_t &= \{((q,\bot), a, (r,r)) \mid (q,a,r) \in \Delta \text{ and } r \in Q_{ev}\} \\
\Delta_p &= \{((q,p), a, (r,p)) \mid (q,a,r) \in \Delta \text{ and } c(q), c(r) \geq c(p)\}
\end{aligned}$$

The set of accepting states is $F = \{(p,p) \mid p \in Q_{ev}\}$

The states $Q_\bot$ and the relation $\Delta_\bot$ correspond to the copy of $\mathcal{P}$ without any accepting state, and for every $p \in Q_{ev}$, the states Q_p and the relation Δ_p correspond to the copy where p is accepting. The relation Δ_t serves as a transition between the non-accepting copy and the other copies. It is easy to show that $\mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{B})$. □

10.2.2 Parity simplification in the logic

Let $\mathcal{P}$ be an NPW and $\mathcal{B}$ be the NBW obtained in the proof of Proposition 10.3. We show that $[\mathcal{B}] \vdash [\mathcal{P}]$ has a proof in $\mu\mathsf{LK}\odot$. For that purpose, we build a meager $\mu\mathsf{LK}\odot^\omega$ proof of this sequent, which gives us a $\mu\mathsf{LK}\odot$ proof by Proposition 7.2. We use the notations

introduced in the proof of Proposition 10.3. Since q_I and $(q_I, \bot)$ are the initial states of $\mathcal{P}$ and $\mathcal{B}$ respectively, we have that $[\mathcal{P}] = [\![q_I]\!]^{\emptyset}$ and $[\mathcal{B}] = [\![(q_I, \bot)]\!]^{\emptyset}$. Our goal is then to show that $[\![(q_I, \bot)]\!]^{\emptyset} \vdash [\![q_I]\!]^{\emptyset}$ has a meager $\mu\mathsf{LK}\odot^{\omega}$ proof. To do so, we need to generalize this statement, and to look for proofs of sequents having the form $[\![(q, r)^{\Theta}]\!] \vdash [\![q]\!]^{\Upsilon}$ where $q \in Q$, $r \in Q_{ev} \cup \{\bot\}$ and Θ, Υ are environments of $\mathcal{B}$ and $\mathcal{P}$ respectively. We show in the following lemma.

Lemma 10.2. *The following rule is derivable in $\mu\mathsf{LK}\odot^{\infty}$ using a finite meager derivation, where $t = (q, a, q')$:*

$$\frac{\{[\![(q', r')]\!]^{\Theta,((q,r),T)} \vdash [\![q']\!]^{\Upsilon,(q,t)}\}_{T=((q,r),a,(q',r'))\in\Delta_{\mathcal{B}}}}{[\![(q, r)]\!]^{\Theta} \vdash [\![q]\!]^{\Upsilon}} \ (\hookrightarrow)$$

Proof. We start by decomposing the left-hand side of the sequent using Proposition 9.3:

$$\frac{\{[a], \odot[\![(q', r')]\!]^{\Theta,((q,r),T)} \vdash [\![q]\!]^{\Upsilon}\}_{T=((q,r),a,(q',r'))\in\Delta_{\mathcal{B}}}}{[\![(q, r)]\!]^{\Theta} \vdash [\![q]\!]^{\Upsilon}}$$

Now we have to justify every premise of this derivation. Let $T = ((q, r), a, (q', r')) \in \Delta_{\mathcal{B}}$. By construction of $\Delta_{\mathcal{B}}$, we have that $t = (q, a, q') \in \Delta$. We have also that:

$$[\![q]\!]^{\Upsilon} \to \bigvee_{t=(q,a,q')\in\Delta} [a] \wedge \odot[\![q']\!]^{\Upsilon,(q,t)}$$

We set $\Gamma = [a], \odot[\![(q', r')]\!]^{\Theta,((q,r),T)}$.

$$\dfrac{\dfrac{\dfrac{}{\Gamma \vdash [a]}\ (\mathsf{Ax}) \qquad \dfrac{[\![(q', r')]\!]^{\Theta,((q,r),T)} \vdash [\![q']\!]^{\Upsilon,(q,t)}}{\Gamma \vdash \odot[\![q']\!]^{\Upsilon,(q,t)}}\ (\odot)}{\Gamma \vdash [a] \wedge \odot[\![q']\!]^{\Upsilon,(q,t)}}\ (\wedge_{\mathsf{r}})}{\Gamma \vdash [\![q]\!]^{\Upsilon}}\ (\sigma_{\mathsf{r}}),\ (\vee_{\mathsf{r}}),\ (\mathsf{W}_{\mathsf{r}})$$

□

The premises of the derivation $\hookrightarrow$ have the same shape as its conclusion. If we iterate coinductively $\hookrightarrow$ starting from $[\mathcal{B}] \vdash [\mathcal{P}]$, which is $[\![(q_I, \bot)]\!]^{\emptyset} \vdash [\![q_I]\!]^{\emptyset}$, we get a meager $\mu\mathsf{LK}\odot^{\infty}$ pre-proof. We show that this pre-proof is a $\mu\mathsf{LK}\odot^{\infty}$ proof.

Proposition 10.4. *Let θ be the $\mu\mathsf{LK}\odot^{\infty}$ pre-proof obtained by applying coinductively $\hookrightarrow$. One has that θ is a meager and regular $\mu\mathsf{LK}\odot^{\infty}$ proof.*

Proof. It is easy to see that θ is regular. Let us show that it is valid. Let β be a branch of θ and $([\![(q_i, r_i)]\!]^{\Theta_i} \vdash [\![q_i]\!]^{\Upsilon_i})_{i\in\omega}$ be the sequence of conclusions of the derivation $\hookrightarrow$ in β. We have that $\forall i \in \omega$, $\exists a_i$ such that $(q_i, a_i, q_{i+1}) \in \Delta$. Let $u = (a_i)_{i\in\omega}$. The sequence $\rho_{\mathcal{B}} = ((q_i, r_i))_{i\in\omega}$ is a run of $\mathcal{B}$ over u and $\rho_{\mathcal{P}} = (q_i)_{i\in\omega}$ is a run of $\mathcal{P}$ over u. There are two possible cases:

- The run $\rho_{\mathcal{B}}$ is not accepting. Then by Proposition 9.4 the left thread of β is a μ-thread, hence β is a valid branch.

- The run $\rho_{\mathcal{B}}$ is accepting. This means that there is $r \in Q_{ev}$ and $j \in \omega$ such that $\forall i \geq j$ we have $r_i = r$ and $c(q_i) \geq c(r)$ and $q_i = r$ infinitely many times. Hence the run $\rho_{\mathcal{P}}$ is also accepting. By Proposition 9.4, the right thread of β is a ν-thread, hence β is a valid branch.

□

Using Proposition 7.2, we get the following theorem, which is step III of our completeness proof. The other implication can be show in the same way.

Theorem 10.2. *One can construct a* $\mu\mathsf{LK}\odot$ *proof of* $[\mathcal{B}] \vdash [\mathcal{P}]$.

Chapter 11

Büchi inclusions in $\mu\mathsf{LK}\odot$

In this chapter, we show that we can reflect language inclusions of Büchi automata in the logic. More precisely, given two NBW $\mathcal{B}_1$ and $\mathcal{B}_2$ such that $\mathcal{L}(\mathcal{B}_1) \subseteq \mathcal{L}(\mathcal{B}_2)$, we show that we can construct effectively a $\mu\mathsf{LK}\odot$ proof of $[\mathcal{B}_1] \vdash [\mathcal{B}_2]$. As we did in Chapter 10, we will go through the circular proof system $\mu\mathsf{LK}\odot^\omega$, but in a different way. Indeed, during our investigations, we were able to construct a $\mu\mathsf{LK}\odot^\omega$ proof of $[\mathcal{B}_1] \vdash [\mathcal{B}_2]$, unfortunately the proof we obtained does not satisfy the translatability criterion. We had to think about a new approach. Our idea was then to construct a $\mu\mathsf{LK}\odot$ proof of $[\mathcal{B}_1] \vdash [\mathcal{B}_2]$ by induction on a measure that we call the *level of non-determinism* of $\mathcal{B}_2$. During this induction, we step out of the class of Büchi automata, and the base case of the induction consists on constructing a $\mu\mathsf{LK}\odot$ proof of $[\mathcal{A}_1] \vdash [\mathcal{A}_2]$, for every NPW $\mathcal{A}_1$ and $\mathcal{A}_2$ such that the level of non-determinism of $\mathcal{A}_2$ is zero, and such that $\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$. To construct such a proof, we search for a $\mu\mathsf{LK}\odot^\omega$ proof of $[\mathcal{A}_1] \vdash [\mathcal{A}_2]$ satisfying the translatability criterion. In fact, we will show a more general result: for every NPW $\mathcal{A}_1$ and $\mathcal{A}_2$ such that $\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$, we can construct a $\mu\mathsf{LK}\odot^\omega$ proof of $[\mathcal{A}_1] \vdash [\mathcal{A}_2]$. In the particular case where the level of non determinism of $\mathcal{A}_2$ is zero, we show that the obtained proof satisfies the translatability criterion, which gives us the base case of the induction.

For presentational purposes, we will not work directly with automata but rather with two classes of formulas, slightly more general then the encoding of automata: the class of *disjunctive formulas*, roughly corresponding to the encoding of NPW, and the class of *Büchi formulas* corresponding to the encoding of NBW. We show then the following results:

1. If φ is a Büchi formula and ψ a disjunctive one such that $\mathcal{L}(\varphi) \subseteq \mathcal{L}(\psi)$ then we can construct a $\mu\mathsf{LK}\odot^\omega$ proof of $\varphi \vdash \psi$.

2. If in addition, the level of non-determinism of ψ is zero, then the obtained proof satisfies the translatability criterion.

3. We finally show that for every Büchi formulas φ and ψ such that $\mathcal{L}(\varphi) \subseteq \mathcal{L}(\psi)$, we can construct effectively a $\mu\mathsf{LK}\odot$ proof of $\varphi \vdash \psi$.

To show Point 1, we use Safra's construction, a construction commonly used to determinize parity automata (but also Rabin, Streett and Büchi automata), as a proof search algorithm in $\mu\mathsf{LK}\odot^\omega$, in a way that we make clear in Section 11.2. To obtain Point 3, we genralize it as follows: for every disjunctive formulas φ and ψ such that the level of non-determinism of ψ is n, we can construct a $\mu\mathsf{LK}\odot$ proof of $\varphi \vdash \psi$. We show this result by induction on the

level of non-determinism of ψ. Finally, let us note that the level of non-determinism is not well defined for every disjunctive formula. Thus, the last result does not easily genralize to every disjunctive formula ψ.

Organization of the chapter. We define in Section 11.1, the class of disjunctive and Büchi formulas. In Section 11.2, we show that the circular proof system $\mu\mathsf{LK}\odot^{\omega}$ is complete for inclusions of disjunctive formulas, using Safra's determinization construction. We show in Section 11.3 that the circular proof constructed in Section 11.2 satisfies the translatability criterion when the formulas are "deterministic enough". This yields completeness of $\mu\mathsf{LK}\odot$ for inclusions of sufficiently deterministic disjunctive formulas. We finally establish completeness for inclusions of NBW, by gradually diminishing their level of non-determinism.

11.1 Power-set construction in the logic

We now transpose automata-theoretic notions to an appropriate class of formulas that contains the image of our encoding. This will be useful since most of our work is done directly on formulas, and it allows us to state simpler and slightly more general results.

Definition 11.1. ***Disjunctive formulas*** are defined as follows:

$$\varphi ::= X \mid \sigma X.\, \vee_{i\in I}\, (a_i \wedge \odot\varphi_i) \qquad \text{with} \qquad X \in \mathcal{X},\ a_i \in \Sigma.$$

The set of disjunctive formulas is denoted by $\mathcal{F}_{\vee}$. A ***Büchi formula*** is any $\varphi \in \mathcal{F}_{\vee}$ such that, for all $\mu X.\psi \leq \varphi$, there is no $\nu Y.\xi \leq \psi$ with $X \in \mathsf{fv}(\nu Y.\xi)$.

We call ***disjunctive occurrences*** the occurrences of disjunctive formulas, and ***Büchi occurrences*** the occurrences of Büchi formulas.

Remark 11.1. Notice that the set of closed disjunctive (resp. Büchi) formulas is the image of NPW (resp. NBW) by our encoding.

Definition 11.2. Let F be a disjunctive occurrence. The set of ***states*** of F, denoted $Q(F)$, is the set of disjunctive occurrences which are FL-suboccurrences of F, that is $Q(F) := \mathsf{FL}_o(F) \cap \mathcal{F}_{\vee}$. When we forget the addresses of the occurrences of $Q(F)$, we get the finite set denoted $\overline{Q}(F)$.

For $G, H \in Q(F)$ we write $G \xrightarrow{a} H$ if $G = \sigma X.\, \vee_{i\in I} F_i$ and $F_i = (a \wedge \odot H')$ for some $i \in I$ such that $H = H'[G/X]$. We set $a^{-1}G := \{H : G \xrightarrow{a} H\}$. The ***transition relation*** of F is the function $\delta(F)$ defined by:

$$\begin{array}{rccc} \delta(F): & Q(F)\times\Sigma & \to & 2^{Q(F)} \\ & (G,a) & \mapsto & a^{-1}G \end{array}$$

A ***priority assignment*** of F is any function $c : Q(F) \to \omega$ whose co-domain is finite and satisfying the following conditions:

- G is a ν-formula iff $c(G)$ is even.
- If $G \leq H$ then $c(G) \leq c(H)$.
- If $G \equiv H$ then $c(G) = c(H)$.

Let $\alpha := (a_i)_{i\in\omega} \in \Sigma^\omega$. A ***run*** of F over α is a sequence $(G_i)_{i\in\omega}$ of states of F such that $G_0 = F$ and for all $i \geq 0$, $G_i \xrightarrow{a_i} G_{i+1}$. We say that a run ρ of F is ***accepting*** if $\mathsf{min}(\mathsf{Inf}(c(\rho)))$ is even.

For every infinite thread starting from a disjunctive occurrence F, we can associate a run of F in a unique way, and conversely. We show in Proposition 11.1 that a thread and its associated run have the same behaviour, this will allow us to work with the two notions of thread and run indifferently later on.

Definition 11.3. Let F be a disjunctive occurrence and let ρ be a run of F. We define $t(\rho)$ to be the unique thread of F containing ρ as a sub-sequence. Conversely, If t is a thread of F, we define $\rho(t)$ to be the sub-sequence of t collecting the elements of $Q(F)$.

Proposition 11.1. *Let F be a disjunctive occurrence, t be an infinite thread of F and ρ be a run of F. The thread t is a ν-thread if and only if $\rho(t)$ is accepting. The run ρ is accepting if and only if $t(\rho)$ is a ν-thread.*

Proof. We show that t is a ν-thread if and only if $\rho(t)$ is accepting. The other assertion can be show in the same way. We set $t = (F_i)_{i\in\omega}$ and $\rho := \rho(t) = (G_k)_{k\in\omega}$. Let c be a priority assignment for F. First observe that if i, j and k are such that $F_i = G_k$ and $F_j = G_{k+1}$, then for every l between i and j, F_l cannot be smaller than F_i, nor than F_j, thus $\mathsf{min}(\mathsf{Inf}(\overline{t})) = \mathsf{min}(\mathsf{Inf}(\overline{\rho}))$. Moreover, since c respects the sub-formula order, we have that if $c(\mathsf{min}(\mathsf{Inf}(\overline{\rho}))) = \mathsf{min}(\mathsf{Inf}(c(\rho)))$. In the left-hand side of the last equation, we used c as a function on disjunctive formulas not on occurrences, this is possible since by definition $c(F) = c(G)$ iff $F \equiv G$, thus c also be seen as a function on formulas. Finally, since $c(G)$ is even iff G is a ν-formula, we conclude that t is a ν-thread iff ρ is accepting. □

Proposition 11.2. *$\alpha \in \mathcal{L}(F)$ iff there is an accepting run of F over α.*

Proof. We show that if $\alpha \in \mathcal{L}(F)$ then there is an accepting run of F over α. Since $\alpha \in \mathcal{L}(F)$, there is an accepting branch b of $\Pi(F)$ that induces α. This means in particular that all the threads of b a ν-threads. Let t be one of them and let ρ be the run associated to t. By Proposition 11.1, the run ρ is accepting. Now it suffices to see that ρ is a run over α. This is true since the premises of any G-derivation where G is disjunctive, are of the form $a, \odot H$ where $G \xrightarrow{a} H$. □

The following propositions shows that the powerset construction can be reflected in the proof system $\mu\mathsf{LK}\odot^\omega$. Indeed, if we neglect the left occurrence in the following rule, we can think of sequents as set of states, and whenever we read a letter a starting from the conclusion, we go to a premise of this rule that contains all the states accessible form the states of the conclusion by reading a.

Proposition 11.3. *For any closed disjunctive occurrences F, G_1, ..., G_n, the following rule is derivable in $\mu\mathsf{LK}\odot^\infty$ using a finite derivation:*

$$\frac{\{F_a \vdash a^{-1}G_1, \ldots, a^{-1}G_n\}_{a\in\Sigma, F\xrightarrow{a}F_a}}{F \vdash G_1, \ldots, G_n}\ (\mathsf{PwrSet})$$

Proof. We restrict to $n = 1$ for clarity, but the general case is similar by applying the right-hand side rules successively on all G_i formulas. In that case, assuming $F = \sigma X.\bigvee_{i\in I}(a_i \wedge \odot F_i)$, we derive $(\to)$ as follows:

$$\dfrac{\dfrac{\pi_1}{a_1, \odot F_1[F/X] \vdash G_1} \quad \ldots \quad \dfrac{\pi_n}{a_n, \odot F_n[F/X] \vdash G_1}}{\sigma X.\ \bigvee_{i\in I}(a_i \wedge \odot F_i) \vdash G_1}\ (\sigma_\mathsf{l}),(\vee_\mathsf{l}),(\wedge_\mathsf{l})$$

with π_i being defined as:

$$\dfrac{\dfrac{\dfrac{F_i[F/X] \vdash a_i^{-1}G_1}{a_i, \odot F_i[F/X] \vdash \{\ \odot G_k^i\ \}_{G_1 \xrightarrow{a_i} G_k^i}}\ (\odot)}{a_i, \odot F_i[F/X] \vdash \{\ a_i \wedge \odot G_k^i\ \}_{G_1 \xrightarrow{a_i} G_k}}\ (\wedge_\mathsf{r}),(\mathsf{Ax}),(\mathsf{W}_\mathsf{l})}{a_i, \odot F_i[F/X] \vdash G_1}\ (\sigma_\mathsf{r}),(\vee_\mathsf{r}),(\mathsf{W}_\mathsf{r})$$

Note that threads of this derived inference rule do not encounter fixed-point formulas except the ones visible at the premise and conclusion of the rule. Therefore, one can ignore the internal construction of the rule when checking the validity of pre-proofs. □

11.2 Parity automata inclusions in $\mu\mathsf{LK}\odot^\omega$

We shall establish the completeness of $\mu\mathsf{LK}\odot^\omega$ for automata inclusions. Here we can actually work with the full class of parity automata, which we exploit later.

Theorem 11.1. *For any disjunctive occurrences F and G one has:*

$$\textit{If } \mathcal{L}(F) \subseteq \mathcal{L}(G) \quad \textit{then} \quad F \vdash G \textit{ is derivable in } \mu\mathsf{LK}\odot^\omega.$$

Proposition 11.3 indicates that we can embed the powerset construction in the logic, suggesting a natural strategy for building proofs of language inclusions. But we face a problem here due to the fact that our sequents are not made of sets of formulas, but keep track of distinct occurrences of formulas. This means that we are in fact considering a "powermultiset" construction, accounting for all possible runs rather than reachable states only. If we apply this naive construction, keeping all the copies of a state coming from different states, we are certainly going to get a $\mu\mathsf{LK}\odot^\infty$ proof, but this proof has no chance of being regular in general, since sequents will become larger and larger. We illustrate this in the following example.

Example 11.1. Consider the automata $\mathcal{B}_1$ and $\mathcal{B}_2$ depicted in Figures 11.1 and 11.2. We have obviously $\mathcal{L}(\mathcal{B}_1) \subseteq \mathcal{L}(\mathcal{B}_2)$ since $\mathcal{B}_1$ has an empty language.

In this example, we use the notations of Section 9.2.2. We simplify the notation $[\![q]\!]^{(q_1,t_1),\ldots,(q_n,t_n)}$ by forgetting the transitions and simply writing $[\![q]\!]^{q_1,\ldots,q_n}$. If we apply coinductively the rule (PwrSet) to the sequent $[\![e]\!]^\emptyset \vdash [\![i]\!]^\emptyset$, we get the following $\mu\mathsf{LK}\odot^\infty$

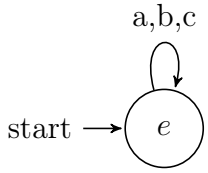


Figure 11.1: Automaton $\mathcal{B}_1$

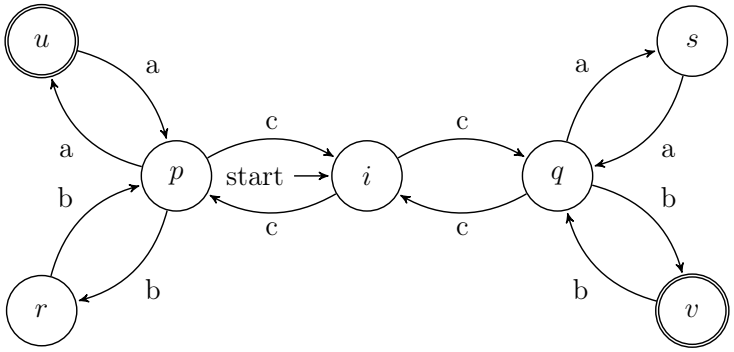


Figure 11.2: Automaton $\mathcal{B}_2$

proof.

$$\dfrac{\dfrac{\dfrac{(\star)}{[\![e]\!]^{ee} \vdash [\![p]\!]^{ipu}, [\![q]\!]^{ips}}\,(\mathsf{PwrSet})}{[\![e]\!]^{ee} \vdash [\![u]\!]^{ip}, [\![s]\!]^{ip}}\,(\mathsf{PwrSet}) \qquad \dfrac{\dfrac{(\star)}{[\![e]\!]^{ee} \vdash [\![p]\!]^{ipr}, [\![q]\!]^{iqv}}\,(\mathsf{PwrSet})}{[\![e]\!]^{ee} \vdash [\![r]\!]^{ip}, [\![v]\!]^{iq}}\,(\mathsf{PwrSet}) \qquad \dfrac{\vdots}{[\![e]\!]^{ee} \vdash [\![i]\!]^{ip}, [\![i]\!]^{iq}}\,(\mathsf{PwrSet})}{\dfrac{[\![e]\!]^{e} \vdash [\![p]\!]^{i}, [\![q]\!]^{i} \quad (\star)}{[\![e]\!]^{\emptyset} \vdash [\![i]\!]^{\emptyset}}\,(\mathsf{PwrSet})}\,(\mathsf{PwrSet})$$

This proof is not regular since the sequents of its rightmost branch get larger and larger.

Note that the infinite branches of this derivation correspond one to one to the infinite words over the alphabet $\{a, b, c\}$. For instance, the rightmost branch of this proof correspond to the word c^{ω}.

In order to recover regularity, we have to weaken some occurrences. The challenge is to do so whilst preserving validity. But this is not easy to perform in general as illustrated by the following Example.

Example 11.2. The non regularity of the proof of Example 11.1 comes from the sequent $[\![e]\!]^{ee} \vdash [\![i]\!]^{ip}, [\![i]\!]^{iq}$ where the two right occurrences, coming both from the right occurrence of the conclusion, will generate each in turn two other copies, etc. To get a regular proof starting from this proof, one idea would be to weaken one of these two occurrences, $[\![i]\!]^{iq}$ for

instance, and close the cycle on $[\![e]\!] \vdash [\![i]\!]^{ip}$, which would give the following derivation.

$$\dfrac{\dfrac{\dfrac{(\star)}{[\![e]\!]^{ee} \vdash [\![p]\!]^{ipu}, [\![q]\!]^{ips}}\,(\mathsf{PwrSet})}{[\![e]\!]^{ee} \vdash [\![u]\!]^{ip}, [\![s]\!]^{ip}}\,(\mathsf{PwrSet}) \qquad \dfrac{\dfrac{(\star)}{[\![e]\!]^{ee} \vdash [\![p]\!]^{ipr}, [\![q]\!]^{iqv}}\,(\mathsf{PwrSet})}{[\![e]\!]^{ee} \vdash [\![r]\!]^{ip}, [\![v]\!]^{iq}}\,(\mathsf{PwrSet}) \qquad \dfrac{\dfrac{(\dagger)}{[\![e]\!]^{ee} \vdash [\![i]\!]^{ip}}}{[\![e]\!]^{ee} \vdash [\![i]\!]^{ip}, [\![i]\!]^{iq}}\,(\mathsf{W})}{\dfrac{[\![e]\!]^{e} \vdash [\![p]\!]^{i}, [\![q]\!]^{i} \quad (\star)}{[\![e]\!]^{\emptyset} \vdash [\![i]\!]^{\emptyset} \quad (\dagger)}\,(\mathsf{PwrSet})}\,(\mathsf{PwrSet})$$

But this regular derivation is not a proof. Indeed, it is easy to check that the branch corresponding to the word $c.(b.b.c.c)^\omega$ is not valid. The same thing holds if we weaken the occurrence $[\![i]\!]^{ip}$, since the branch corresponding to the word $c.(a.a.c.c)^\omega$ will not be valid.

We denote by $\mathsf{PS}(F \vdash G)$ the infinitary proof obtained by coinductively iterating the (PwrSet) starting from $F \vdash G$. As illustrated by the previous example, using weakenings in a naive way to turn $\mathsf{PS}(F \vdash G)$ into a regular proof does not work. To solve this problem, we will adopt an other approach.

Since the left-hand sides of the sequents $\mathsf{PS}(F \vdash G)$ contain at most one formula, its the right threads (which are the runs of G) who causes the non regularity of $\mathsf{PS}(F \vdash G)$. Our goal then is to select a bounded subset of the runs of G in a way that brings a valid and regular proof. This problematic looks very much like the problematic of determinization of ω-automata. To determinize these automata, the powerset constructions does not work, or it works by keeping multiple copies of the same state, but then we fall out the scope of finite automata. An efficient recipe to determinize ω-automata is the well-known ***Safra's construction***. This construction is a refinement of the powerset construction, in the sens that the macro-states are not simply sets of states, but they are equipped with a structure of tree which indicates a hierarchy between these states, those trees are called Safra-trees (or S-trees for simplicity). We will show that this is exactly what we need: our construction (PwrSet) was too coarse, and thus led to a non-regular proof. We need to refine it into a finer rule, inspired by Safra's construction.

We will adapt Safra's construction for Streett automata, introduced in [Saf92, GTW02b], on two dimensions: first, as our disjunctive formulas correspond to parity automata, we should specialize this construction designed for Streett automata to parity ones. Second, this construction determinizes automata, while we are dealing with occurrences. As we are morally determinizing the occurrence G, the labels of Safra-trees will be the states of G.

In the rest of this section, we assume a fixed occurrence G. We set $Q := Q(G)$, $\delta := \delta(G)$ and let c be a priority assignment for G. We consider without loss of generality that the co-domain of c is a subset of $P = \{0, 1, \ldots, 2k, 2k+1\}$.

Definition 11.4. An ***S-tree*** is a finitely branching tree whose leaves are labelled with non-empty subsets of Q, whose edges are labelled by priorities from P, and satisfying the following conditions:

- Descendants of a node are ordered, from left (younger) to right (older).
- Moreover, if p, q are two leaves' labels, then the underlying formulas of p and q are distinct, that is $p \not\equiv q$.

- Every internal node has at least one outgoing edge with an odd label.
- On every branch, edge labels appear in increasing order starting from the root.
- Further, only even labels can appear more than once on a branch and if two labels occur on a branch then odd labels in between must occur too.

Each node is identified by a name from the set $N := \{1, \dots, 2 \times |Q(G)| \times (2k+2)\}$. We say that $v \in N$ is a node in T if T contains a node of name v. We denote by $e_T(v)$ the union of the labels of all leaves of the subtree of T rooted in node v.

Remark 11.2. If we forget the addresses of the states of an S-tree T, we get a tree that we denote by $\overline{T}$, whose leaves are labelled by FL-subformulas of G. We say that two S-trees T_1 and T_2 are equal up-to-renaming if $\overline{T_1} = \overline{T_2}$, and we write $T_1 \equiv T_2$. Note that only finitely many S-trees exist up-to-renaming.

The role of node names is to be able to track nodes through the modifications of the trees corresponding to automata transitions.

Given an S-tree T and a letter a, we now describe the transition function $\Delta(T, a)$ by the following procedure.

Definition 11.5. Let T be an S-tree and a a letter. The ***S-tree*** $\Delta(T, a)$ is the tree obtained after the following steps. Whenever we create a new node in this procedure, we assume that it is given a fresh name, *i.e.*, one that does not yet occur in the whole tree.

(1) For any leaf node v_0 of T, let $2m+1$ be the least odd label that does not occur from the root to v_0 (skip this step if all odd labels occur). Create nodes $v_1, \dots, v_{k-m+1}$ such that for each $i = 0, \dots, k-m$, v_{i+1} is the son of v_i, with an edge (v_i, v_{i+1}) labelled $2(m+i)+1$. The leaf v_{k-m+1} takes the label of v_0.

(2) Replace each leaf label S by $a^{-1}S$.

(3) For each leaf v and state G appearing in it, let (w, w') be the edge appearing in the path from the root to v, that is labelled by $c(F)$ if $c(F)$ is odd, $c(F)+1$ otherwise. Remove q from the label of v and append to w a new child leaf v', to the right, labelled $\{F\}$, with the edge (w, v') labelled $c(F)$.

(4) If two states F_1 and F_2 such that $F_1 \equiv F_2$ belong respectively to two distinct leaves w_1, and w_2, let b_1, b_2 be two branches starting from the root and ending in w_1, w_2 respectively. Let v be the node on which these branches fork, and v_i be the child of v in b_i, $i \in \{1, 2\}$. If the label of the edge (v, v_1) is strictly greater than that of (v, v_2) then remove F_1 from w_1, otherwise remove F_2 from w_2. If they have the same label, remove q_i from w_i, where w_i is the rightmost leave.

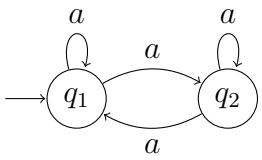
(5) Remove any subtree whose leaves all have empty labels.

(6) If after the previous steps all edges going from a node v to its children are labelled by the same even priority, make v a leaf and label it $e_T(v)$.

Example 11.3. We develop an example of Safra's construction illustrating the different steps of the construction of Definition 11.5. Let us consider the disjunctive formula:

$$\varphi = \mu X.(a \wedge \odot X) \vee (a \wedge \odot(\nu Y.(a \wedge \odot X) \vee (a \wedge \odot Y)))$$

We set $\psi = \nu Y.(a \wedge \odot\varphi) \vee (a \wedge \odot Y)$ and $G = \varphi_\varepsilon$. Let c be the priority assignment for G defined by: if a state F of G is of the form φ_α then $c(F) = 1$, otherwise $c(F) = 2$. We thus consider the following set of priorities $P = \{0, 1, 2, 3\}$. The formula φ is the encoding of the parity automaton $\mathcal{A} = (\{q_1, q_2\}, q_1, \delta, c)$ depicted below, where priorities are such that $c(q_i) = i$.



Let T_1 be the S-tree with a single node of name n_0, labelled by G. We show Figure 11.3 the S-trees $T_2 := \Delta(T_1, a)$ and $T_3 := \Delta(T_2, a)$. We denote node names by using brackets (*e.g.* $[n_0]$) and we indicate the leaves' labels to the right of their names (*e.g.* $[n_0]\ \{F\}$). In Figure 11.3, $G_1 = \varphi_\alpha$, $H_1 = \psi_\beta$ where $\alpha = ilri$, $\beta = irri$, and $G_2 = \varphi_{\alpha.\alpha}$, $H_2 = \psi_{\alpha.\beta}$.

We also detail the construction of $T_3 = \Delta(T_2, a)$ in Figure 11.4, where $\overset{i}{\Rightarrow}$ denotes the application of Step (i) of Safra's construction. The occurrences G_3 and H_3 are defined by $G_3 = \varphi_{\beta.\alpha}$ and $H_3 = \psi_{\beta.\beta}$.

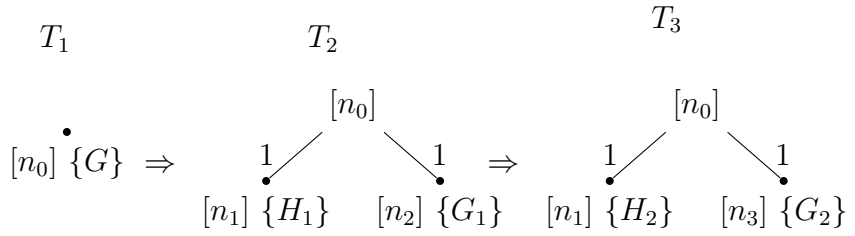


Figure 11.3: Example of S-trees obtained by Safra's construction.

Remark 11.3. The following properties are invariants of our Safra construction, that is to say, they are satisfied by the initial singleton trees and are preserved by Δ:

- If w is a leaf of an S-tree T and F appears in the label of w, then $c(F) \geq p$, where p is any edge label from the root to w.
- If v is an internal node in an S-tree T, then there is an even priority p such that all the outgoing edges from v in T are either labelled p or $p + 1$. In the following, this even priority will be denoted by $p_T(v)$, or simply $p(v)$ when T is obvious.
- Moreover, if $T' = \Delta(T, a)$ and v appears in both T and T' as an internal node, then $p_T(v) = p_{T'}(v)$.

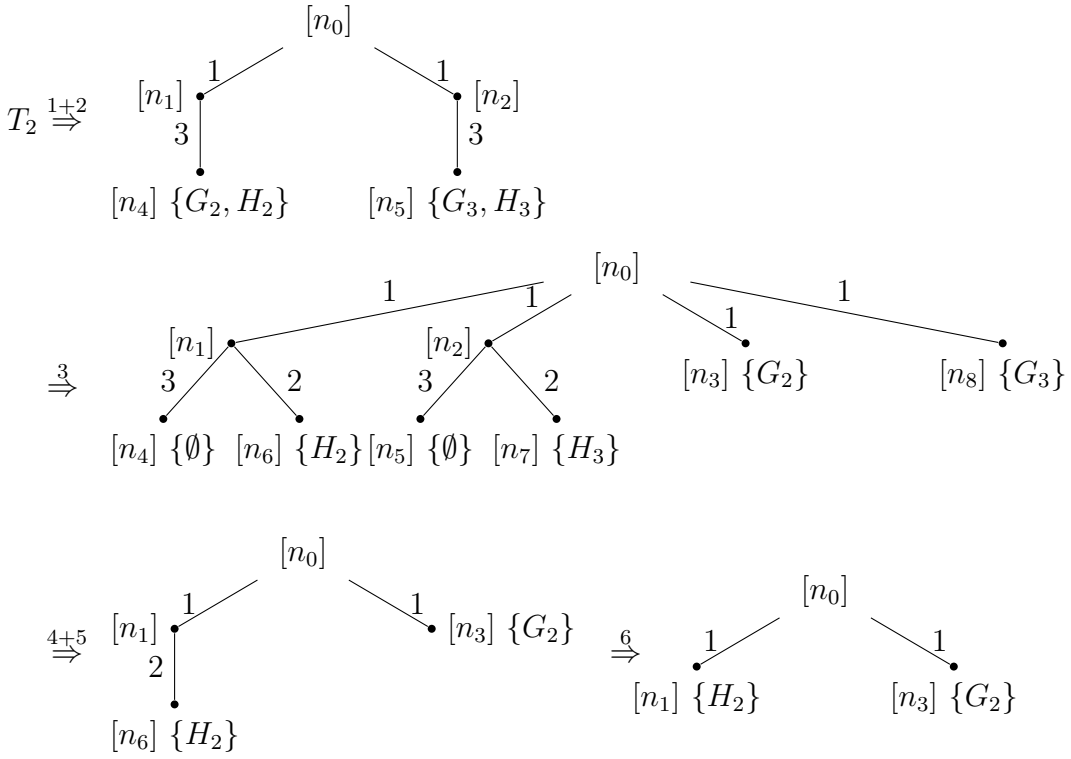


Figure 11.4: Internal steps of $\Delta(T_2, a) = T_3$

Proposition 11.4. *Let $\alpha := (a_i)_{i\in\omega} \in \Sigma^{\omega}$ and $\rho := (G_i)_{i\in\omega}$ be a run of G over α. Let T_0 be the S-tree with a single leaf node labelled $\{G_0\}$, and $T_{i+1} := \Delta(T_i, a_i)$ for all $i \in \omega$. If ρ is accepting then there is a node $v \in N$ and an index j such that for all $i > j$, v is a node in T_i and v becomes a leaf infinitely often.*

Proof. Let us first prove the following assertion:

Assertion: Let $v \in N$. If there is $i_v \in \omega$ such that for all $i > i_v$, v is a node of T_i and $G_i \in e_{T_i}(v)$, then there are two possibilities:

1. The node v becomes a leaf infinitely often.
2. There is $w \in N$ and $i_w \in \omega$ such that for all $i > i_w$, w is a son of v in T_i and $G_i \in e_{T_i}(w)$.

Proof of the assertion: Let m be the minimal priority that appears infinitely often in ρ. Suppose that v does not become a leaf infinitely often: let k be an index such that for all $i > k$, the node v is never a leaf in T_i and $c(G_i) \geq m$. By Remark 11.3, there is an even priority p such that $p = p_{T_i}(v)$ for all $i > i_v$. Since $G_i \in e_{T_i}(v)$ for all $i > k$, and again by Remark 11.3, we have that $p \leq m$. For all $i > k$, $G_i \in e_{T_i}(w_i)$ for some child w_i of v; we seek to establish that $(w_i)_{i>k}$ is eventually constant. We distinguish two cases:

(i) If $\mathbf{p < m}$. We actually have $p+1 < m$. Let T'_i be the tree obtained after steps (1–3) of the construction of $T_{i+1} = \Delta(T_i, a_i)$. Because $c(G_{i+1}) > p+1$ we still have $G_{i+1} \in e_{T'_i}(w_i)$. We now consider the rest of the construction, from T'_i to T_{i+1}. Step (4) may remove the occurrence of $G_{i+1} \in e_{T'_i}(w_i)$ in favor of another one in $eT'_i(w_{i+1})$ for $w_{i+1} \neq w_i$. However,

this can only happen if (a) the edge (v, w_{i+1}) is labelled p while (v, w_i) is labelled $p+1$, or (b) the edge labels are the same but the rank of w_{i+1} among the children of v is less than that of w_i. Thus, this can happen only finitely many times, and the sequence $(w_i)_i$ is eventually constant. (Obviously, Step (5) is irrelevant in this argument, and Step (6) cannot happen because v never becomes a leaf by assumption.)

(ii) Otherwise, $\mathbf{p = m}$. We first show that there is some i such that (v, w_i) is labelled p. Consider any position $j > k$ such that $c(G_i) = m$. If $(v, w_i) = p + 1$ then G_{i+1} will be moved by step (3) of the construction of T_{i+1} to a new child w' of v, with an edge (v, w') labelled p. Then step (4) may not keep the occurrence of G_{i+1} in w', but it will preserve an occurrence in a child w'' with (v, w'') labelled p too. Next we observe that, once (v, w_i) is labelled p, we have (v, w_j) labelled p as well for all $j \geq i$. This follows from a simple inspection of the procedure, arguing as above. From this point on, we can only have $w_j \neq w_{j+1}$ because of step (4) but, as observed before, this can only decrease the rank of w_{j+1} among the children of v. This can only happen finitely often, which concludes the proof.

To conclude, we iterate the assertion starting from the root node and iterate it as many times as necessary to obtain the result. The iteration is bounded by the maximal height of S-trees. □

For an S-tree T, we define $\Phi(T)$ to be the union of the labels of the leaves of T, forgetting all the tree structure. In the following, when unambiguous, we will sometimes write T instead of $\Phi(T)$. The S-tree structure corresponds to a refined powerset construction, in the sense that, if T is an S-tree and $a \in \Sigma$, $\Phi(\Delta(T, a)) \subseteq a^{-1}(\Phi(T))$. We now observe that Safra's construction can be immediately adapted to proof theory, in a way that yields proofs of inclusions for disjunctive formulas.

Proposition 11.5. *Given a disjunctive occurrence F and an S-tree T, the following rule is derivable:*

$$\frac{\{\ F_a \vdash \Phi(\Delta(T, a))\ \}_{a\in\Sigma, F \xrightarrow{a} F_a}}{F \vdash \Phi(T)}\ (\mathsf{S})$$

Proof. Since $\Phi(\Delta(T, a)) \subseteq a^{-1}(\Phi(T))$, rule (S) can be obtained by applying rule (PwrSet) of Proposition 11.3 and some weakenings. □

Definition 11.6. Given a disjunctive occurrence F and an S-tree T, we coinductively define $\Pi(F, T)$ to be the following $\mu\mathsf{LK}\odot^\infty$ pre-proof:

$$\frac{\left\{\ \dfrac{\Pi(F_a, \Delta(T, a))}{F_a \vdash \Phi(\Delta(T, a))}\ \right\}_{a\in\Sigma, F \xrightarrow{a} F_a}}{F \vdash \Phi(T)}\ (\mathsf{S})$$

In other words, it is the infinite derivation obtained by repeatedly applying rule (S). If G is a disjunctive formula, we set $\Pi(F, G) := \Pi(F, T)$ where T is the single-node S-tree with label $\{G\}$.

Note that $\Pi(F, T)$ is a standard $\mu\mathsf{LK}\odot^\infty$ pre-proof, whose sequents are sets of occurrences, but those sequents have been obtained by forgetting the structure of S-trees which form the blueprint of the construction.

Theorem 11.2. *Let $F, G \in \mathcal{F}_{\vee}$. If $\mathcal{L}(F) \subseteq \mathcal{L}(G)$, then $\Pi(F, G)$ is a $\mu\mathsf{LK}\odot^{\infty}$ proof.*

Proof. Let us establish the validity of each infinite branch $\gamma = (s_i)_{i\in\omega}$ of $\Pi(F, G)$. For each such branch, there is an ω-word $\alpha = (a_i)_{i\in\omega}$ such that, for all i, $s_i = F_i \vdash \Phi(T_i)$ where $F_0 = F$, T_0 is the tree with a single node labeled by $\{G\}$, $F_i \xrightarrow{a_i} F_{i+1}$ and $\Delta(T_i, a_i) = T_{i+1}$. Hence $\rho := (F_i)_{i\in\omega}$ corresponds to a run of F over α.

If the run ρ is not accepting, we have by Proposition 11.1 that its corresponding thread $t(\rho)$ is a μ-formula and thus validates the branch γ.

Otherwise, the run ρ is accepting, thus $\alpha \in \mathcal{L}(F)$ by Proposition 11.2, and $\alpha \in \mathcal{L}(G)$ by hypothesis. By Proposition 11.4, there is a node v and an index j such that $\forall i \geq j$, v is a node of T_i and v appears as a leaf infinitely often. Now, let us extract a valid thread from this node. To do so, we prove the following assertion:

Assertion 1: Let $m > n \geq j$ such that the node v is a leaf in T_n and T_m, and it is not a leaf in T_i for all $n < i < m$. For every $K \in e_{T_m}(v)$, there is a sequence of occurrences $G_n, \ldots, G_m$ satisfying the following conditions:

- $G_m = K$;
- for all $i = n, \ldots, m-1$, $G_i \xrightarrow{a_i} G_{i+1}$
- $\mathsf{min}(\overline{G_n}, \ldots, \overline{G_m})$ is a ν-formula.

We first show that the assertion allows us to conclude the main proof. Let $j < i_0 < i_1 < \ldots$ such that v is a leaf in T_i where $i > j$ iff $i = i_n$ for some $n \in \omega$. We define a tree $\mathcal{T}$ whose set of nodes N is

$$N := \{r\} \cup \{\ (K, n) \mid n \in \omega, K \in e_{T_{i_n}}(v)\ \},$$

where r is a distinguished root node, with an edge from r to any node of the form $(K, 0)$, and an edge from (H, n) to $(K, n+1)$ iff there is a thread $G_{i_n} G_{i_n+1} \ldots G_{i_{n+1}}$ such that $H = G_{i_n}$, $K = G_{i_{n+1}}$ and $\mathsf{min}(\overline{G_{i_n}}, \overline{G_{i_n+1}}, \ldots, \overline{G_{i_{n+1}}})$ is a ν-formula. By the previous assertion, every node except r have at least a parent. The tree $\mathcal{T}$ is thus an infinite tree which is finitely branching, hence by König's lemma it has an infinite branch. It is easy to see that this branch corresponds to a valid thread.

Before proving Assertion 1, we will establish a more precise result. By Remark 11.3, there exists an even priority p such that for all $i \geq j$, the outgoing edges from v are either labelled p or $p+1$. We shall now prove the following, by induction on $i = n, \ldots, m-1$:

Assertion 2: Let w be a child of v in T_i and $K \in e_{T_i}(w)$, then there are occurrences $F_k, \ldots, F_i$ satisfying the following conditions:

- $F_i = K$;
- For all $l = n, \ldots, i-1$, $F_l \xrightarrow{a_l} F_{l+1}$.
- For all $l = n, \ldots, i$, $F_l \in e_{T_l}(v)$.
- If l is the label of the edge (v, w), then $\mathsf{min}(c(F_{n+1}), \ldots, c(F_i)) = l$.

Let us prove Assertion 2. When $i = n$, the result is obvious, notably because v is a leaf in T_n. Now, suppose that the result is true for i and let us prove it for $i+1$. Let w be a child of v in T_{i+1} and $K \in e_{T_{i+1}}(w)$. The edge (v, w) is labelled p. Notice first that $c(K) \geq l$, otherwise it would be absorbed by a parent of v.

There exists $L \in e_{T_i}(v)$ such that $L \xrightarrow{a_i} K$. Actually, L should belong to one of the children of v in T_i; call it w' and let l' be the label of the edge connecting v and w' in T_i. We distinguish two cases:

1. If $w = w'$ we conclude by applying the induction hypothesis.

2. If $w \neq w'$, that is w is a new child in T_{i+1}, created by Step 3, then we distinguish two cases: if $l = l'$, here again we apply directly the induction hypothesis. Otherwise, the node w has been created because $c(F_{i+1}) = p$ and $l' = p+1$. We then have that $l = p$. Here again we apply induction hypothesis to conclude.

We finally prove Assertion 1. Let $K \in e_{T_m}(v)$ and let w be a child of v in T_{m-1} and L an occurrence such that $L \in e_{T_{m-1}}(w)$ and $L \xrightarrow{a_{m-1}} K$. By the previous result instantiated for $i = m-1$, we have a thread $F_n, \dots F_{m-1}$ such that $F_{m-1} = L$ and $\mathsf{min}(c(F_n), \dots, c(F_{m-1})) = l$ where $l \in \{p, p+1\}$ is the label of the edge (v, w). As v is a leaf in T_m, it has been obtained after Step 6, which means that before Step 6, the tree T_{m-1} has only ongoing edges labelled p. Let w' be the child of T_{m-1} just before Step 6 such that K is in the label of this subtree rooted in w'. There are two possibilities:

- If $w' = w$, we have $l = p$ and, since $c(F_m) \geq p$, then $\mathsf{min}(c(F_n), \dots, c(F_{m-1})) = p$.

- Otherwise, w' has been created in Step 3, and $c(F_m) = p$. Hence we also have that:

$$\mathsf{min}(c(F_n), \dots, c(F_m)) = p$$

Anyway, the thread $F_n, \dots F_m$ satisfies $\mathsf{min}(c(F_n), \dots, c(F_m)) = p$. Since p is even, we have that $\mathsf{min}(F_n, \dots, F_m)$ is a ν-formula. □

Definition 11.7. The pre-proof $\Pi(F, T)$ is regular. Indeed, if T' and T'' are two S-trees equal up-to-renaming via the bijection b, then $\Pi(F, T')$ and $\Pi(F, T'')$ are also equal up-to-renaming via the same bijection b. We denote by $\Theta(F, T)$ the minimal circular representation of $\Pi(F, T)$.

11.3 Büchi automata inclusions in $\mu\mathsf{LK}\odot$

Now, given two disjunctive formulas F and G such that $\mathcal{L}(F) \subseteq \mathcal{L}(G)$, we want to construct a $\mu\mathsf{LK}\odot$ proof of the sequent $F \vdash G$. A natural idea is to build the circular proof $\Theta(F, G)$ introduced in the previous section, then to translate it into a $\mu\mathsf{LK}\odot$ proof using the translation algorithm described in Chapter 2, Section 2.4.5. Unfortunately, we are limited by the condition of application of this algorithm, which requires the input circular proof to be *translatable* (Definition 2.39), since the proof $\Theta(F, G)$ does not satisfy this condition in general. The difficulty to translate circular proofs into finitary ones is the "non-determinism" of the right hand-side formulas of the conclusion sequent. For instance, if G is deterministic (that is, the encoding of a deterministic automaton), then $\Theta(F, G)$ meets easily our

translation condition. We will push this observation further, and show that if G contains a "weak form of non-determinism" (we call this class of formulas weakly deterministic), then $\Theta(F, G)$ can be translated into a $\mu\mathsf{LK}\odot$ proof.

To establish our final result (Theorem 11.3), we will not use a direct translation of $\Theta(F, G)$ into a $\mu\mathsf{LK}\odot$ proof, but rather an induction on a measure that we call the "level of non-determinism of G". More precisely, we will define a hierarchy of formulas, whose lowest level is the set of weakly deterministic formulas, then the highest we go in this hierarchy, the most formulas contain "complex" form of non-determinism. The level of non-determinism of a formula, if it exists, is the lowest index of the hierarchy to which this formula belongs. We show, by induction on this level, that if G belongs to the hierarchy, then $F \vdash G$ is provable in $\mu\mathsf{LK}\odot$, the base case being that of weakly deterministic formulas which we will show by a direct translation as we have discussed above. We finally show that Büchi automata belong to our hierarchy, and this concludes the proof of Theorem 11.3. Unfortunately, not all disjunctive formulas belong to the hierarchy, that is why our result is stated only for Büchi formulas.

11.3.1 Weakly deterministic formulas

We define a particular class of disjunctive formulas called *weakly deterministic*. We show that if G is weakly determinstic, then all the valid traces of $\Theta(F, G)$ are strongly valid (Definition 2.39).

Definition 11.8. A disjunctive formula is said to be ***deterministic*** if and only if, in all its sub-formulas of the form $\vee_{i\in I}(a_i \wedge \odot\varphi_i)$, the letters a_i are pairwise distinct. We define the class of formulas $\mathcal{I}$ to be the set of disjunctive formulas, not containing the ν-connective. That is:

$$\varphi \in \mathcal{I} := X \;\mid\; \mu X.\, \vee_{i\in I}\,(a_i \wedge \odot\varphi_i)$$

We define the class of formulas $\mathcal{E}^\mu$ as follows:

$$\mathcal{E}^\mu = \mathcal{I} \cup \{\vee_{i\in I}(a_i \wedge \odot\varphi_i) \mid \varphi_i \in \mathcal{I}\}$$

The set of ***weakly deterministic formulas*** $\mathcal{H}_0$, is the set of formulas of the form $\varphi[\mathcal{D}_i/X_i]_{i=1}^n$, where $\varphi \in \mathcal{E}^\mu$ and $[\mathcal{D}_i]_{i=1}^n$ are deterministic disjunctive formulas.

Remark 11.4. Since we consider capture-avoiding substitution, in every weakly deterministic formula $\varphi[\mathcal{D}_i/X_i]_{i=1}^n$, the free variables of $\mathcal{D}_i$ cannot be bound in φ.

The elements of $\mathcal{E}^\mu$ are essentially those of $\mathcal{I}$, but we allow the top-level connective to not to be the μ-connective. Weakly deterministic formulas are obtained by putting deterministic formulas into an environment of $\mathcal{E}^\mu$ formulas, such that no free variable in these deterministic formulas is bound in the environment. In a weakly deterministic formula, the non-determinism can come only from the $\mathcal{E}^\mu$ part. A valid run of $\mathcal{H}_0$ cannot stay always in this $\mathcal{E}^\mu$ part, since it contains only μ-formulas, thus such a run should eventually visit one of the deterministic components of the formula. Since the $\mathcal{E}^\mu$ part is not accessible from the deterministic ones, then every valid run stays eventually in one of the deterministic components. Thus the infinitary part of every valid run comes from a deterministic automaton, that is why we say that $\mathcal{H}_0$ formulas have a weak form of non-determinism.

Now we will show that if G is weakly deterministic, then all the valid tarces $\Theta(F, G)$ are strongly valid.

Proposition 11.6. *Let F be a disjunctive occurrence and G be a weakly deterministic disjunctive occurrence. If $\mathcal{L}(F) \subseteq \mathcal{L}(G)$ then very valid trace of $\Theta(F, G)$ is strongly valid.*

Proof. Since G is weakly deterministic, it has the form $G = N[D_i/X_i]_{i=1}^n$ where D_k are deterministic and N is an occurrence that does not contain the ν connective. Let us denote the set of states of D_k by Q_k and those of N by Q_N.

The construction of $\Theta(F, G)$ depends on the priority assignment c associated to G. Here we choose a priority assignment c such that $c(K) = 1$ for all $K \in Q_N$, and $c(K) > 1$ for all $K \in Q_i$, $1 \leq i \leq n$. Such an assignment exists since Q_N contains only μ-formulas and, for all $1 \leq i \leq n$, the states in Q_N and Q_i are not co-accessible.

Let us first make some observation about Safra's construction for G, with the chosen priority assignment c. Consider the sequence $(T_i)_{i\in\omega}$ on some word $(a_i)_{i\in\omega}$, as in the proof of Theorem 11.2.

Assertion: Let $i \in \omega$ and v be a child of the root r in T_i. One has:

- The edge (r, v) is labelled by 1.
- Either $e_{T_i}(v) = \{K\}$ where $K \in Q_N$ and v is a leaf, in this case we call v a child ***of Type 1***. Otherwise, $e_{T_i}(v) = \{K_1, \ldots, K_p\}$ such that for all $k \in [1, p]$ there exists $j_k \in [1, n]$ such that $K_k \in Q_{j_k}$, and $k \neq k'$ implies $j_k \neq j_{k'}$, in this case, we call v a child ***of Type 2***.
- If v is also a node in T_{i+1}, then v is of Type 2 in T_{i+1}.

We prove this assertion by induction on i. When $i = 0$ it is trivial since the root has no child. Suppose that the assertion is true for some i and let us prove it for $i + 1$. Let v be a child of the root in T_{i+1}. The edge (r, v) is labelled by 1: indeed the outgoing edges from the root can only be 1 or 0. But by definition of the priority assignment we gave above, no state has priority 0. On the other hand, the only way to create an edge of label 0 is by Step 3 when a formula has a priority 0. Hence the outgoing edges from r are labelled by 1. To prove the second item, we separate two cases:

- If v does not exist in T_i this means that w has been created in Step 3, hence it is a leaf labelled by a singleton $\{K\}$. We have that $K \in Q_N$ since only occurrences from Q_N have priority 1.
- Otherwise, v is a child of r in T_i. By induction hypothesis, the node v is of Type 2 in T_{i+1}.

Suppose that v is also a child of r in T_{i+2}. There are two cases:

- If v is of Type 1 in T_{i+1}, let $e_{T_{i+1}}(v) = \{K\}$ and $\delta(K, a_{i+1}) = \{\vec{M}, \vec{K}\}$, $\vec{M} \subseteq Q_N$ and $\vec{K} \subseteq \cup_{1\leq i\leq n} Q_i$. The tree structure of G implies that $\vec{K}$ is of the form $\vec{K} := \{K_1, \ldots, K_n\}$ with at most one K_k per Q_j, and none in Q_N. After Step 3, all occurrences of $\vec{M}$ will be in new child of the root in T_{i+2}. Hence, v will be of Type 2 in T_{i+2}.

- If v is of Type 2 in T_{i+1}, it is also of Type 2 in T_{i+2} since D_k are deterministic.

Having proved our assertion, we now establish the result. Let v be a node and j an index such that for all $i \geq j$, v is a node in T_i and v is a leaf infinitely often. As the outgoing edges of the root all have label 1, v cannot be the root. Indeed, to become a leaf infinitely often, a node has to collapse in Step 6, and this is possible only when the outgoing edges are of even priority. Hence v appears in the sub-tree of a child w of the root in T_i, for all $i > j$. Since v is persistent, w is also persistent, and by the above assertion, the node w is of Type 2 in T_i, for all $i > j$. Hence v is also of Type 2, that is, $e_{T_i}(v)$ has at most one occurrence from each Q_k and no occurrence from Q_N.

We finally come back to the proof of Theorem 11.2, and show that each time we exhibited a valid thread t in $\Pi(F, G)$, then the corresponding trace (which we call also t for simplicity) in $\Theta(F, G)$ is strongly valid. This follows from the following remarks:

(i) When the sequents of a branch contain only one occurrence in their left-hand side, any valid thrace visiting only left-hand side occurrences is strongly valid. Since all the sequents in $\Theta(F, G)$ have only one formula in their left-hand side, the left-hand side thread exhibited in the first case of the proof of Theorem 11.2 is actually strongly valid.

(ii) Consider the right-hand side trace $t = (G_i)_{i\in\omega}$ that we extracted from the persistent node v of the path $p = (n_i)_{i\in\omega}$ in the second case of the proof of Theorem 11.2. Let us show that eventually, whenever t meets a node of $\Theta(F, G)$, it meets it at the level of the same occurrence. Since t is valid, there is $l, k \in \omega$ such that for every $i \geq l$, $G_i \in \mathcal{D}_k$. By contradition, suppose that there is $j, m \geq l$ such that $n_j = n_m$ but $G_j \neq G_m$. Since the trace t has been extraxted from the persistent node v, we have that both G_j and G_m belong to $e_{T_j}(v)$, which means that $e_{T_j}(v)$ contains two distinct elements from Q_k and this contradicts the last observation above.

□

11.3.2 Büchi inclusions

We define a hierarchy of formulas such that whenever we go high in its levels, we find more "complex" forms of non-determinism.

Definition 11.9. We define the ***hierarchy*** $(\mathcal{H}_i)_{i\in\omega}$ of sets of formulas as follows:

- $\mathcal{H}_0$ is the set of weakly deterministic formulas.
- $\mathcal{H}_{i+1} = \mathcal{H}_i \cup \{\varphi[\nu X_j.\psi_j/Y_j]_{j=1}^n \mid n \in \omega, \varphi \in \mathcal{H}_0, \psi_j \in \mathcal{H}_i\}$

We call ***level of non-determinism*** of a formula φ, if it exists, the least i such that $\varphi \in \mathcal{H}_i$.

We show the following stability properties of our hierarchy, that will be useful in the proofs.

Proposition 11.7. *Let $\psi \in \mathcal{H}_i$. We have:*

- *If ψ is a disjunctive formula and $\varphi \in \mathcal{E}^\mu$ then $\varphi[\psi/X] \in \mathcal{H}_i$.*
- *If $\mathcal{D}$ is deterministic, then $\psi[\mathcal{D}/X] \in \mathcal{H}_i$.*

Proof. Suppose *w.l.o.g.* that the level of non-determinism of ψ is i. Then ψ is of the form $\gamma[\nu X_j.\psi_j/Y_j]_{j=1}^n$ where $\gamma \in \mathcal{H}_0$ and $\psi_j \in \mathcal{H}_i$. Since ψ is disjunctive, the formula γ is also disjunctive, thus it is of the form $\delta[\mathcal{D}_k/X_k]_k$, where $\delta \in \mathcal{I}$ and $\mathcal{D}_k$ are deterministic formulas. The first item follows from the fact that if we pre-compose a $\mathcal{I}$ formula with a $\mathcal{E}^\mu$ one, the result is in $\mathcal{E}^\mu$, in particular $\varphi[\delta/X] \in \mathcal{E}^\mu$.

We show the second item by induction on the level of non-determinism of ψ. The base case is trivial. We suppose that the result is true for the level i and show it for $i+1$. Since the level of non-determinism of ψ is $i+1$, ψ is of the form $\varphi[\nu X_j.\psi_j/Y_j]_{j=1}^n$ where $\varphi \in \mathcal{H}_0$ and $\psi_j \in \mathcal{H}_i$. By the base case, $\varphi[\mathcal{D}/X] \in \mathcal{H}_0$ and by IH, $\psi_j[\mathcal{D}/X] \in \mathcal{H}_i$, thus $\psi \in \mathcal{H}_{i+1}$. □

As established by the following proposition, every Büchi formula belongs to the hierarchy.

Proposition 11.8. *For every Büchi formula φ, there is an index k such that $\varphi \in \mathcal{H}_k$.*

Proof. We show the result by induction on φ. If φ is a variable, then it belongs to $\mathcal{H}_0$. Otherwise, let ψ be the formula not containing any ν-connective such that φ is of the form $\varphi = \psi[\nu X_i.\psi_i/Y_i]_{i\in I}$. It is always possible to write a Büchi formula under this form since no μ connective can bind a variable which is free in a ν-formula. Every formula ψ_i is of the form $\psi_i = \vee_{j\in I_i}(a_j \wedge \odot\varphi_j)$, where φ_j are Büchi formulas. By induction hypothesis, the formulas φ_j belong to a level $\mathcal{H}_k$ (the induction hypothesis may give different indices to each φ_j, we take k to be their max). We can write ψ_i under the form $\delta_i[\varphi_j/Z_j]_{j\in I_j}$, where $\delta_j = \vee_{j\in I_i}(a_j \wedge \odot Z_j)$. The formula δ_j obviously belongs to $\mathcal{E}^\mu$, thus by Proposition 11.7 ψ_i is in $\mathcal{H}_k$. Hence, φ is in $\mathcal{H}_{k+1}$. □

Remark 11.5. Disjunctive formulas do not all belong to the hierarchy. For instance, the formula $\mu X.a \wedge \odot(\nu Y.(a \wedge \odot X) \vee (a \wedge Y))$ does not belong to any $\mathcal{H}_k$.

Theorem 11.3. *Let F and G be two Büchi occurrences. If $\mathcal{L}(F) \subseteq \mathcal{L}(G)$ then we can construct effectively a $\mu\mathsf{LK}\odot$ proof of $F \vdash G$.*

Proof. Let us consider the following more general result:

> *Let F be a disjunctive occurrence and G be disjunctive occurrence of level of non-determinism k. If $\mathcal{L}(F) \subseteq \mathcal{L}(G)$ then we can construct effectively a $\mu\mathsf{LK}\odot$ proof of $F \vdash G$.*

We prove the latter by induction on the level of non-determinism of G.

Base case: When the level of non-determinism of G is 0, that is when G is weakly deterministic, we apply Proposition 11.6 and Proposition 2.17 to get a $\mu\mathsf{LK}\odot$ proof of $F \vdash G$.

Inductive case: Suppose that the level of non-determinism of G is $k+1$. Thus, G can be written in the form $G = H[\nu X_i.G_i/Y_i]_{i\in I}$ where $H \in \mathcal{H}_0$ and $G_i \in \mathcal{H}_k$. Since G is closed, every $\nu X_i.G_i$ is also closed, thus by Remark 11.1, it is the encoding of a NPW, that we denote by $\mathcal{A}_i$. Let $\mathcal{D}_i$ be a deterministic automaton accepting the same language as $\mathcal{A}_i$, obtained by an effective construction, Safra's construction for instance [SV14]. Let D_i be

the encoding of $\mathcal{D}_i$. Notice that $\mathcal{L}(G) = \mathcal{L}(H[D_i/Y_i]_{i\in I})$. Consider the following derivation:

$$\dfrac{\dfrac{\theta}{F \vdash H[D_i/Y_i]_{i\in I}} \qquad \dfrac{\left\{ \dfrac{\pi_i}{D_i \vdash \nu X_i.G_i} \right\}_{i\in I}}{H[D_i/Y_i]_{i\in I} \vdash H[\nu X_i.G_i/Y_i]_{i\in I}}\ (\mathsf{F_H})}{F \vdash G}\ (\mathsf{Cut})$$

The derivation θ can be obtained by the base case since $H[D_i/Y_i]_{i\in I}$ is a $\mathcal{H}_0$ formula. For each $i \in I$, the sub-proof π_i is obtained by applying the coinduction rule (ν_r) with co-invariant D_i for $\nu X_i.G_i$ as follows.

$$\dfrac{\dfrac{}{D_i \vdash D_i}\ (\mathsf{Ax}) \qquad \dfrac{\theta_i}{D_i \vdash G_i[D_i/X_i]}}{D_i \vdash \nu X_i.G_i}\ (\nu)$$

The proof θ_i is obtained by IH. Indeed, since D_i is deterministic and since $G_i \in \mathcal{H}_k$, then by Proposition 11.7, $G_i[D_i/X_i] \in \mathcal{H}_k$. Thus we can construct a proof θ_i of $D_i \vdash \nu X_i.G_i$. This concludes the inductive case. □

Remark 11.6. We can adapt the algorithm described in the proof of Theorem 11.3 to get one that takes two arbitrary Büchi occurrences F and G, and outputs either a proof of $F \vdash G$ in case of $\mathcal{L}(F) \subseteq \mathcal{L}(G)$, or a counter-example otherwise, that is a word u such that $u \in \mathcal{L}(F)$ but $u \notin \mathcal{L}(G)$. The algorithm of the proof of Theorem 11.3 can run on arbitrary formulas independently from the hypothesis of language inclusion, the only place where the algorithm may get stuck is the base case. In the base case, corresponding to weakly deterministic formulas, we start by building the derivation $\Theta(F, G)$ then we run the algorithm of Theorem 2.5 to construct a $\mu\mathsf{LK}\odot$ proof of $F \vdash G$. In case $\mathcal{L}(F) \subseteq \mathcal{L}(G)$, this algorithm produces a $\mu\mathsf{LK}\odot$ proof, otherwise, it gets stuck in a "Strongly connected" phase. This strongly connected component of $\Theta(F, G)$ induces a word of the form v^ω which is in $\mathcal{L}(F)$ but not in $\mathcal{L}(G)$, thus can be returned as a counter-example.

Chapter 12

Constructive completeness

We have shown in Chapters 9 and 10 the steps I-III of our proof of completeness. In Chapter 11, we have shown Step IV'. Now we are left with the steps IV and V, which we shall establish in this Chapter. Once this is done, we show how to get the constructive completeness from these five building blocks.

12.1 Picking up all the pieces

Step IV is an easy consequence of Theorem 11.3:

Theorem 12.1. *If $\mathcal{B}$ is a NBW such that $\mathcal{L}(\mathcal{B}) = \Sigma^\omega$ then there is a DBW $\mathcal{D}$ such that $\mathcal{L}(\mathcal{D}) = \Sigma^\omega$ and one can construct effectively a $\mu\mathsf{LK}\odot$ proof of $[\mathcal{D}] \vdash [\mathcal{B}]$.*

Proof. Let $\mathcal{D}$ be the DBW $\{\Sigma, \{q\}, q, \Delta, \{q\}\}$ where $\Delta = \{(q, a, q) | a \in \Sigma\}$. We have obviously that $\mathcal{L}(\mathcal{D}) = \Sigma^\omega$. By Theorem 11.3, we can construct effectively a $\mu\mathsf{LK}\odot$ proof of $[\mathcal{D}] \vdash [\mathcal{B}]$. □

Step V relies on the same idea used before: construct a meager circular proof of $\vdash [\mathcal{D}]$, then translate into a $\mu\mathsf{LK}\odot$ proof. Contrarily to earlier steps, the search of the circular proof is much easier.

Theorem 12.2. *For every DBW $\mathcal{D}$ such that $\mathcal{L}(\mathcal{D}) = \Sigma^\omega$, one can construct effectively a $\mu\mathsf{LK}\odot$ proof of $\vdash [\mathcal{D}]$.*

We first show the following lemma:

Lemma 12.1. *Let Δ be the transition relation of $\mathcal{D}$, p be a state and Θ be an environment for $\mathcal{D}$. The following rule is derivable in $\mu\mathsf{LK}\odot^\infty$ using a finite meager derivation:*

$$\frac{\{\vdash [\![q]\!]^{\Theta,(p,t)}\}_{t=(p,a,q)\in\Delta}}{\vdash [\![p]\!]^\Theta}\ (\to)$$

Proof. Since $\mathcal{D}$ is deterministic, for every $a \in \Sigma$ there is a unique state q_a such that $t_a := (p, a, q_a) \in \Delta$. We have then:

$$[\![p]\!]^\Theta \to \bigvee_{a\in\Sigma} [a] \wedge [\![q_a]\!]^{\Theta,(p,t_a)}$$

hence we have the following derivation:

$$\dfrac{\dfrac{\{\vdash [a] \wedge [\![q_a]\!]^{\Theta,(p,t_a)}\}_{a\in\Sigma}}{\vdash \bigvee\limits_{a\in\Sigma} [a] \wedge [\![q_a]\!]^{\Theta,(p,t_a)}}\ (\vee_{\mathsf{r}})}{\vdash [\![p]\!]^{\Theta}}\ (\sigma_{\mathsf{r}})$$

To justify this premise, we apply the following derivation whenever it applies:

$$\dfrac{\vdash [a], \Gamma \qquad \dfrac{\vdash [\![q_a]\!]^{\Theta,(p,t_a)}}{\vdash [\![q]\!]^{\Theta,(p,t_a)}, \Gamma}\ (\mathsf{W}_{\mathsf{r}})}{\vdash [a] \wedge [\![q_a]\!]^{\Theta,(p,t_a)}, \Gamma}\ (\wedge_{\mathsf{r}})$$

Doing so, we get the following rule:

$$\dfrac{\vdash \{[a]\}_{a\in\Sigma} \qquad \{\vdash [\![q_a]\!]^{\Theta,(q_a,t)}\}_{a\in\Sigma}}{\vdash [\![p]\!]^{\Theta}}\ (\rightarrow)$$

Now we show that $\vdash \{[a]\}_{a\in\Sigma}$ is provable. For that we notice that $\{[a]\}_{a\in\Sigma} = \{p \wedge [a]\}_{a\in\Sigma_p} \cup \{p^{\perp} \wedge [a]\}_{a\in\Sigma_p}$ where $\Sigma_p = 2^{\mathcal{P}\setminus\{p\}}$. We apply the following scheme recursively, to obtain a finite proof of $\vdash \{[a]\}_{a\in\Sigma}$:

$$\dfrac{\dfrac{}{\vdash p, p^{\perp}}\ (\mathsf{Ax}) \qquad \vdash \{[a]\}_{a\in\Sigma_p}}{\vdash \{[a]\}_{a\in\Sigma}}\ (\wedge_{\mathsf{r}}),\ (\mathsf{W}_{\mathsf{r}})$$

Which concludes the proof. □

Proof of Theorem 12.2. Let π be the proof obtained by applying coinductively the derivation of Lemma 12.1. This derivation is a meager $\mu\mathsf{LK}\odot^{\infty}$ pre-proof. Note that π is a regular derivation. To show that it π is actually a proof, notice that every infinite branch β of π contains exactly one right thread t. The run-branch of t, $\rho(t)$ is an accepting run, since every run of $\mathcal{D}$ is accepting. By Proposition 9.4, t is a ν-thread. Thus π is a proof. □

We can show now our completeness result.

Theorem 12.3. *If $\models \varphi$, we can construct a $\mu\mathsf{LK}\odot$ proof of $\vdash \varphi$.*

Proof. Let φ be a valid formula.

I. Let $\mathcal{A}_{\varphi}$ the APW associated to φ (Definition 9.14). By Proposition 9.8, one has $\mathcal{M}(\varphi) = \mathcal{L}(\mathcal{A}_{\varphi})$, thus $\mathcal{L}(\mathcal{A}_{\varphi}) = \Sigma^{\omega}$. By Theorem 9.1, one can construct a $\mu\mathsf{LK}\odot$ proof π_1 of $[\mathcal{A}_{\varphi}] \vdash \varphi$.

II. Let $\mathcal{P}$ be the NPW constructed from $\mathcal{A}_{\varphi}$ using the algorithm of the proof of Proposition 10.1. One has $\mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{A}) = \Sigma^{\omega}$. By Theorem 10.1, one can construct a $\mu\mathsf{LK}\odot$ proof π_2 of $[\mathcal{P}] \vdash [\mathcal{A}_{\varphi}]$.

III. Let $\mathcal{B}$ be the NBW constructed from $\mathcal{P}$ using the algorithm of the proof of Proposition 10.3. One has $\mathcal{L}(\mathcal{B}) = \mathcal{L}(\mathcal{P}) = \Sigma^{\omega}$. By Theorem 10.2, one can construct a $\mu\mathsf{LK}\odot$ proof π_3 of $[\mathcal{B}] \vdash [\mathcal{P}]$.

IV. Since $\mathcal{L}(\mathcal{B}) = \Sigma^\omega$ and using Theorem 12.1, there is DBW $\mathcal{D}$ such that $\mathcal{L}(\mathcal{D}) = \Sigma^\omega$ and such that we can build a proof π_4 of $[\mathcal{D}] \vdash [\mathcal{B}]$.

V. Finally by Proposition 12.2, one can construct a proof π_5 of $\vdash [\mathcal{D}]$.

We gather all these pieces together to build a $\mu\mathsf{LK}\odot$ proof of $\vdash \varphi$ using several cuts:

$$\dfrac{\dfrac{\pi_5}{\vdash [\mathcal{D}]} \quad \dfrac{\pi_4}{[\mathcal{D}] \vdash [\mathcal{B}]} \quad \dfrac{\pi_3}{[\mathcal{B}] \vdash [\mathcal{P}]} \quad \dfrac{\pi_2}{[\mathcal{P}] \vdash [\mathcal{A}_\varphi]} \quad \dfrac{\pi_1}{[\mathcal{A}_\varphi] \vdash \varphi}}{\vdash \varphi}\ (\mathsf{Cut})$$

□

Remark 12.1. The proof of Theorem 12.3 describes an algorithm that outputs a $\mu\mathsf{LK}\odot$ proof for every valid formula. We can adapt it to get an algorithm that takes an arbitrary formula φ and outputs either a proof of φ if φ is valid, or a word $u \notin \mathcal{M}(\varphi)$, as follows:

1. Build the automata $\mathcal{A}_\varphi$, $\mathcal{P}$ and $\mathcal{B}$ as in the proof of Theorem 12.3. By construction, $\mathcal{M}(\varphi) = \mathcal{L}(\mathcal{P}) = \mathcal{L}(\mathcal{B})$.
2. Construct a DBW $\mathcal{D}$ such that $\mathcal{L}(\mathcal{D}) = \Sigma^\omega$.
3. Run the improved algorithm of remark 11.6 on the Büchi automata $\mathcal{B}$ and $\mathcal{D}$. This algorithm has the ability to output either a proof of $[\mathcal{D}] \vdash [\mathcal{B}]$ if $\mathcal{L}(\mathcal{D}) \subseteq \mathcal{L}(\mathcal{B})$; or a word u such that $u \in \mathcal{L}(\mathcal{D})$ and $u \notin \mathcal{L}(\mathcal{B})$ otherwise.
4. If this algorithm outputs a proof of $[\mathcal{D}] \vdash [\mathcal{B}]$, we can carry on with the other steps of the algorithm of the proof of Thereom 12.3.
5. Otherwise, the word u is a counter-example of the validity of φ since $u \in \mathcal{L}(\mathcal{D}) = \Sigma^\omega$ but $u \notin \mathcal{L}(\mathcal{B}) = \mathcal{M}(\varphi)$.

12.2 Discussion

We have given a new completeness argument for the full linear-time μ-calculus with respect to $\mu\mathsf{LK}\odot$. Unlike earlier proofs, our argument is constructive, that is, it exhibits a proof for every valid formula. To achieve this, we have combined techniques and tools coming from automata theory and proof theory, two domains that have been growing apart. On the one hand, we used the fine-grained formalism of occurrences which is widely used in proof theory but far removed from the automata side. On the other hand, we showed that the automata transformations can be reflected in the proof-theoretic side.

Related work. Our proof generalizes a previously used idea, which consists in introducing an intermediate class of formulas to solve the completeness problem [Koz83], [Dam92], [Kai95], [Wal95]. The completeness statement in these proofs is generally of the form: "If φ is consistent (*i.e.*, $\neg\varphi$ is unprovable), then φ is satisfiable", whose converse is: "If $\neg\varphi$ is valid then $\neg\varphi$ is provable". Thus to compare our intermediate classes to the ones introduced in these proofs, one should actually have in mind the negations of these classes. Let us give an overview on these proofs and discuss the intermediate classes they use.

- In his seminal paper [Koz83], Kozen introduced the class of aconjunctive formulas:

Definition 12.1. Let φ be a formula and φ' a subformula of it. We say that a variable X is active in φ' if either there is a free occurrence of X in φ' or if there is a free occurrence Y in φ' such that φ' is a subformula of $\sigma Y.\varphi''$, $\sigma Y.\varphi''$ is a subformula of φ and Y is active in $\sigma Y.\varphi''$. A formula is said to be ***aconjunctive*** iff for every subformula $\mu X.\varphi$ of it, there is no subformula $\psi_1 \wedge \varphi_2$ of φ such that X is active in both φ_1 and φ_2.

Kozen showed that every valid formula which is the negation of an aconjunctive formula is provable. The proof proceeds by contradiction. To do so, he builds a tableau (sort of circular proof) for this formula, then assigns a formula to each node of the tableau. The role of these formulas is to record the fixed points that are regenerated, and in which context, in such a way that if the same fixed point is regenerated in the same context in a future node, then the formula assigned to this node will be easily provable. He also shows that if φ is non provable, then there is an infinite path whose nodes are assigned unprovable formulas. Since in this infinite path, some fixed point formula will be eventually regenerated in the same context, this yields a contradiction. Clearly, this argument is not constructive. Moreover, Kozen was not able to show that for every valid formula φ, there is a valid formula $\neg\psi$ where ψ is aconjunctive, such that $\neg\psi \vdash \varphi$. Although partial, this completeness result restricted to aconjunctive formulas will be crucial in the upcoming proofs of completeness.

- Nine years later, in 1992, Dam attempted another approach by introducing an intermediate class which is the class of Büchi automata's encoding [Dam92]. His encoding of NBW is very different from ours. Indeed, he duplicates a Büchi automaton $\mathcal{B}$ into a collection of Büchi automata $\{\mathcal{B}_i\}_i$ having each exactly one accepting state among the accepting states of $\mathcal{B}$. Then he encodes $\mathcal{B}$ as the disjunction of the encodings of the $\mathcal{B}_i$, each of these formulas having exactly one ν-connective. The consequence of such method is that the image of NBW by this encoding forms a class of formulas less rich than the one obtained by our encoding, since its formulas contain at most one alternation of fixed points. Dam showed that every valid formula which is the negation of his encoding of an NBW is provable. The proof of this result is not constructive. He conjectured that every formula is provably equivalent to the encoding of a NBW, and proposed some steps to show this result.

- The first proof of completeness for the full linear-time μ-calculus was given by Kaivola in 1995 [Kai95]. The intermediate class he uses is the class of *banan (bi-aconjunctive non-alternating normal)* form formulas, which are the aconjunctive formulas whose negations are also aconjunctive, and having no alternation of fixed points. Aconjunctivity here is the same notion as in Definition 12.1. Kaivola shows, using the same non-constructive argument as Kozen and specializing it to the linear-time case, that every valid aconjunctive (thus banan form) formula is provable. He also shows that every formula is provably equivalent to a banan form one, using a structural induction on the formula. Inside this induction, there is a call to the first result (aconjunctive and valid implies provable), thus this second result is not constructive itself.

- Independently from Kaivola, Walukiewicz proposes in 1995 [Wal95] a completeness proof for the whole branching-time μ-calculus. His argument uses the intermediate

class of *disjunctive* formulas. We will not formally define disjunctive formulas, but let us just notice that their negations are very close to deterministic alternating parity automata over trees. Walukiewicz shows in a constructive way that every valid formula which is the negation of a disjunctive formula, is provable. The second part of the completeness, that is, every formula is provably equivalent to the negation of a disjunctive formula, is much more involved and makes a detour by (weakly) aconjunctive formulas. Actually, Walukiewicz noticed that what makes Kozen's completeness proof for aconjunctive formulas work, is the particular shape of the tableau associated aconjunctive formulas. He isolated this condition, which he called *thin*, and showed using the same (non-constructive) argument as Kozen, that every formula having a thin refutation is provable. This is precisely where he loses constructivity.

The original sin. Notice that in all these proofs, except for Dam's one, the non constructivity originates from Kozen's argument of completeness for aconjunctive formulas. If one could find a constructive proof for this result, all the others will be constructive as well. Unfortunately, we do not have a clue on how to proceed to make it constructive.

Possible improvement. There might be room for improvements in our argument, for instance by attempting to build more compact circular proofs in the different steps of the proof. In particular, the place where the constructed circular proof can be exponentially large is Step IV, where we use Safra's determinization algorithm. We can wonder wether all the complexity of the S-trees is needed, and if simpler structures may work. For instance, we can consider Piterman trees [Pit07], a structure more succint than Safra's trees, used to determinize NBW and NSW into DPW, with fewer states that Safra's construction. Even if it happens that such techniques can be leaveraged to construct circular proofs, the most important point and maybe the less obvious, is to show that they yield *translatable* circular proofs.

An other improvement channel is the relaxation of the translatability criterion. Recently, we were able to show that every valid formula is provable in the circular proof system $\mu\mathsf{LK}\odot^{\omega}$, by generalizing Safra's construction. This result, which will not be discussed further is this thesis, is for the moment of no use to get a constructive completeness proof, since we are limited by the translatability criterion, which is not satisfied by the circular proofs we obtained.

Evaluation of the complexity. The complexity of our algorithm is far from being optimal. The main source of the complexity blow up is Step IV, where we call recursively Safra's construction in the induction of the proof of Theorem 11.3. A coarse estimation of complexity yields a tour of exponentials. A clever implementation of this algorithm may help to reduce this complexity, but the algorithm remains fundamentally under-optimal.

Extension to the branching-time case. The branhcing-time μ-calculus has infinite trees as models, and in the same way that the linear-time μ-calculus corersponds to APW, the branching-time μ-calculus enjoys the same relationship with alternating parity automata over trees. It seems however difficult to lift our proof technique to show completeness for the branching-time μ-calculus *w.r.t.* Kozen's axiomatization. Indeed, automata over trees are not as well behaved as automata over words, and they do not enjoy the equivalence

properties that we used in our proof. Except for the non-determinization result, that is, every APT can be transfromed into a NPT, all the other equivalences do not hold. For instance, there is no algorithm to transform NPT into NBT automata, nor un algorithm to determinize NPT. We hope though that the tools developed in the present work may help to solve this problem.

Conclusions

We hope that the reader in now convinced that:

> Logics with fixed points can be equipped with infinitary proof systems having a true proof-theoretical status which can be fruitfully applied to other domains such as formal verification.

Instead of summarizing the work that has been done, let us review some perspectives of interest in the continuation of our work.

Perspectives on Part I

Relating the finitary and the infinitary proof systems. The question of relating the finitary and the infinitary proof systems is a hard question. It has been already set as a conjecture in similar settings, for instance by Brotherston and Simpson [BS07] in the framework of classical first-order logic with definitions.

We can ask this question at two levels. The first one is the **provability**: For a given proof system S, does the proof systems $\mu\mathsf{S}$ and $\mu\mathsf{S}^\omega$ prove the same theorems?

> **Problem 1:**
> Do $\mu\mathsf{S}$ and $\mu\mathsf{S}^\omega$ have the same expressive power?

We have partially answered this question by showing that every provable sequent in $\mu\mathsf{S}$ is also provable in $\mu\mathsf{S}^\omega$. The converse in more difficult, and we could only provide a sufficient condition on the proof system $\mu\mathsf{S}$ and $\mu\mathsf{S}^\omega$ (the invariant property Definition 2.40) and a condition on the circular proofs to be translated (the translatability criterion Definition 2.39) which guarantees that they can be indeed transformed into finitary proofs.

If the answer to Problem 1 happens to be positive, we see two ways to proceed:

- Either by finding a semantics of formulas, common to both $\mu\mathsf{S}$ and $\mu\mathsf{S}^\omega$. The idea is to show soundness for $\mu\mathsf{S}^\omega$ (that is, if a formula is provable in $\mu\mathsf{S}$ then it is valid *w.r.t.* this semantics) and completeness for $\mu\mathsf{S}$ (that is, if a formula is valid then it is provable in $\mu\mathsf{S}^\omega$). This is the approach taken by Brotherstone and Simpson [BS07], who could only show soundness of their circular proof system but failed showing completeness.

- Or by using combinatorial techniques, that is by showing a translation procedure that transforms $\mu\mathsf{S}^\omega$ proofs into $\mu\mathsf{S}$ ones. This is the approach adopted in this thesis. It has the advantage of being constructive by essence.

Recently, the conjecture of Brotherston and Simpson has been been answered by Berard and Tatsuta [BT17] and by Alex Simpson [Sim17].

Finitary and circular proof systems can be related at a second level, which is that of the **computational behaviour**. By the Curry-Howard correspondence, proofs correspond to programs. A natural question then is to know if the proofs of $\mu\mathsf{S}$ and $\mu\mathsf{S}^\omega$ respectively denote the same set of programs.

Problem 2:
Do $\mu\mathsf{S}$ and $\mu\mathsf{S}^\omega$ have the same computational power?

If the answer happens to be positive, a way to proceed is by using **proof semantics**. For that, we have to find a domain of interpretation for proofs, common to both $\mu\mathsf{S}$ and $\mu\mathsf{S}^\omega$. If we denote by $[\![\pi]\!]$, the interpretation of a $\mu\mathsf{S}$ or a $\mu\mathsf{S}^\omega$ proof π, the question 2) can be reformulated as follows: For every proof Π of $\mu\mathsf{S}^\omega$ (resp. $\mu\mathsf{S}$), is there a $\mu\mathsf{S}$ (resp. $\mu\mathsf{S}^\omega$) proof π such that $[\![\Pi]\!] = [\![\pi]\!]$?

An example of such proof semantics is computational ludics, where we interpreted the proofs of $\mu\mathsf{MALLP}$ and $\mu\mathsf{MALLP}^\omega$ respectively (Chapter 5).

Extending the cut-elimination to richer logics. We have seen how to extend the cut-elimination rules of a given proof system S to get cut-elimination rules for the proof systems $\mu\mathsf{S}^\infty$ and $\mu\mathsf{S}^\omega$. The question of cut-eliminability remains open in its generality. We have given a positive answer to this question it in the case of $\mu\mathsf{MALL}^\infty$ (Chapter 3) and we hinted the extension to other systems such as $\mu\mathsf{LL}^\infty$, $\mu\mathsf{LK}^\infty$ and $\mu\mathsf{LK}\odot^\infty$. As discussed earlier, it seems difficult to keep circularity after cut-elimination (this is the case for $\mu\mathsf{ALL}^\infty$, $\mu\mathsf{MALL}^\infty$, $\mu\mathsf{LL}^\infty$, $\mu\mathsf{LK}^\infty$ for instance), it is natural then to ask the question of cut-eliminability directly in the infinitary setting rather than in the circular one. In the same vein, another interesting question would be to characterize those circular proofs that remain circular after cut elimination.

Problem 3:
Does the proof system $\mu\mathsf{S}^\infty$ admit the cut-elimination property?

Problem 4:
Characterize those $\mu\mathsf{S}^\omega$ proofs that remain circular after cut-elimination.

Capturing productivity by validity condition. The cut elimination result for infinitary proofs says (in particular) that if a pre-proof satisfies the validity condition, then it is productive (that is, the cut-elimination process eventually produces a part of the result). This is interesting, since the static property of validity guarantees the dynamic property of productivity. We can go further, and ask whether the validity condition **characterizes** productivity. The answer is no since there are pre-proofs having a productive cut-elimination but which fall out of the scope of our validity criterion. This is the case of the following pre-proof for instance, which reduces to a (valid!) pre-proof:

$$\dfrac{\dfrac{(\star)}{\vdash \nu Y.Y} \qquad \dfrac{\dfrac{}{\vdash \mu Y.Y, \nu X.X}\,(\mathsf{Ax})}{\vdash \mu Y.Y, \nu X.X}\,(\nu)}{(\star) \quad \vdash \nu X.X}\,(\mathsf{Cut})$$

We can wonder whether there is a static validity criterion that captures productivity:

> **Problem 5:**
> Find a validity criterion that characterizes productivity of infinitary proofs.

In general, the problem of checking the productivity of a circular proof is undecidable (this is the case for $\mu\mathsf{MALL}^\omega$ for instance). Thus, if we find a validity criterion characterizing productivity (that is answering question 5), it would be necessarily undecidable in turn. But in practice we need a validity criterion which is decidable, an alternative question is then to **approximate** productivity by a decidable correctness criterion.

> **Problem 6:**
> Find decidable validity criteria that guarantee the productivity of infinitary proofs.

Infinitary proofs in natural deduction. The translation between sequent calculus and natural deduction is usually immediate in the finitary case. In the infinitary and the circular setting, this is not the case anymore. Designing infinitary and circular proof systems in natural deduction is crucial, since real-life programs are usually in correspondence with this formalism of proofs rather than with sequent calculus.

> **Problem 7:**
> Develop infinitary proof theory in the natural deduction setting.

The following goal is not a precise scientific question, but rather a more general perspective. Infinitary and circular proofs have received much less attention than proofs using induction from the scientific community. This situation is changing thanks to recent fundamental and applied work **??**, we hope that infinitary and circular proofs will become standard in the proof-theoretical literature.

Perspective

Integrate circular reasoning to theorem provers. and circular programs to programming languages.

Perspectives on Part II

Improving the complexity of the algorithm. In Part II, we have given a constructive completeness proof for the linear-time μ-calculus *w.r.t.* Kozen's axiomatization. One of our motivations was the ability to provide proof certificates. Indeed, we can extract from our constructive proof an algorithm that outputs a proof for every valid formula. This proof can be used as a **certificate**. Unfortunately, the complexity of this underlying argument is so big that we cannot imagine to use it in real-life applications. A natural question is to find algorithms producing proof certificates with **reasonable** complexity.

Problem 8:

Find algorithms, with **reasonable complexity**, that output proofs for valid formulas and counter-examples for non-valid ones.

A constructive completeness proof for the branching-time μ-calculus. This is a natural question once constructive completeness has being established for the linear-time case.

Problem 9:

Show, in a constructive way, completeness for the branching-time μ-calculus *w.r.t.* Kozen's axiomatization.

The model of automata corresponding to the branching-time μ-calculus is that of alternating parity automata over trees. Automata over trees are not as well-behaved as automata over words. And most of the automata equivalences that we have used to solve the linear case do not hold in the branching one. It is necessary to develop other proof techniques. For instance, if general translation procedures from circular to finitary proofs are found, we could imagine to establish this result by first searching for a circular proof for every valid formula, then translating this circular proof into a finitary one.

Bibliography

[AKSW03] Jürgen Avenhaus, Ulrich Kühler, Tobias Schmidt-Samoa, and Claus-Peter Wirth. How to prove inductive theorems? quodlibet! In Franz Baader, editor, *Automated Deduction - CADE-19, 19th International Conference on Automated Deduction Miami Beach, FL, USA, July 28 - August 2, 2003, Proceedings*, volume 2741 of *Lecture Notes in Computer Science*, pages 328–333. Springer, 2003. URL: `http://dx.doi.org/10.1007/978-3-540-45085-6_29`. (Cited on page 19.)

[And92] Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. 2(3):297–347, 1992. (Cited on pages 13, 109, and 111.)

[AP91] Jean-Marc Andreoli and Remo Pareschi. Linear ojects: Logical processes with built-in inheritance. *New Generation Comput.*, 9(3/4):445–474, 1991. URL: `http://dx.doi.org/10.1007/BF03037173`. (Cited on page 109.)

[Bae12a] David Baelde. Least and greatest fixed points in linear logic. *ACM Trans. Comput. Log.*, 13(1):2, 2012. URL: `http://doi.acm.org/10.1145/2071368.2071370`. (Cited on page 18.)

[Bae12b] David Baelde. Least and greatest fixed points in linear logic. *ACM Trans. Comput. Log.*, 13(1):2, 2012. URL: `http://doi.acm.org/10.1145/2071368.2071370`. (Cited on pages 60, 122, 123, and 126.)

[Bar84] H.P. Barendregt. *The lambda calculus: its syntax and semantics*. Studies in logic and the foundations of mathematics. North-Holland, 1984. (Cited on page 11.)

[BDS15] David Baelde, Amina Doumane, and Alexis Saurin. Least and greatest fixed points in ludics. In *24th EACSL Annual Conference on Computer Science Logic, CSL 2015, September 7-10, 2015, Berlin, Germany*, pages 549–566, 2015. URL: `http://dx.doi.org/10.4230/LIPIcs.CSL.2015.549`. (Cited on pages 136 and 143.)

[BF11] Michele Basaldella and Claudia Faggian. Ludics with repetitions (exponentials, interactive types and completeness). *Logical Methods in Computer Science*, 7(2), 2011. URL: `http://dx.doi.org/10.2168/LMCS-7(2:13)2011`. (Cited on pages 22, 122, and 130.)

[BKP] Howard Barringer, Ruurd Kuiper, and Amir Pnueli. A really abstract concurrent model and its temporal logic. In *POPL Florida, USA, January 1986*. (Cited on pages 147 and 153.)

[BKP84] Howard Barringer, Ruurd Kuiper, and Amir Pnueli. Now you may compose temporal logic specifications. In *Proceedings of the 16th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1984, Washington, DC, USA*, pages 51–63, 1984. URL: http://doi.acm.org/10.1145/800057.808665. (Cited on page 153.)

[BL08] Kai Brünnler and Martin Lange. Cut-free sequent systems for temporal logic. *J. Log. Algebr. Program.*, 76(2):216–225, 2008. (Cited on page 37.)

[Bor09] Bianca Boretti. *Proof Analysis in Temporal Logic*. PhD thesis, Universita degli studi di Milano, 2009. (Cited on page 37.)

[Bra98] Julian C. Bradfield. The modal μ-calculus alternation hierarchy is strict. *Theor. Comput. Sci.*, 195(2):133–153, 1998. URL: http://dx.doi.org/10.1016/S0304-3975(97)00217-X. (Cited on pages 16 and 159.)

[BS07] James Brotherston and Alex Simpson. Complete sequent calculi for induction and infinite descent. In *22nd IEEE Symposium on Logic in Computer Science (LICS 2007), 10-12 July 2007, Wroclaw, Poland, Proceedings*, pages 51–62. IEEE Computer Society, 2007. URL: http://dx.doi.org/10.1109/LICS.2007.16. (Cited on pages 18, 21, and 217.)

[BST11] Michele Basaldella, Alexis Saurin, and Kazushige Terui. On the meaning of focalization. In Alain Lecomte and Samuel Tronçon, editors, *Ludics, Dialogue and Interaction - PRELUDE Project - 2006-2009. Revised Selected Papers*, volume 6505 of *Lecture Notes in Computer Science*, pages 78–87. Springer, 2011. (Cited on page 110.)

[BT09] Michele Basaldella and Kazushige Terui. On the meaning of logical completeness. In *Typed Lambda Calculi and Applications, 9th International Conference, TLCA 2009, Brasilia, Brazil*, pages 50–64, 2009. (Cited on page 132.)

[BT17] Stefano Berardi and Makoto Tatsuta. Classical system of martin-löf's inductive definitions is not equivalent to cyclic proof system. In Javier Esparza and Andrzej S. Murawski, editors, *Foundations of Software Science and Computation Structures - 20th International Conference, FOSSACS 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings*, volume 10203 of *Lecture Notes in Computer Science*, pages 301–317, 2017. URL: http://dx.doi.org/10.1007/978-3-662-54458-7_18. (Cited on page 218.)

[CC79] Patrick Cousot and Radhia Cousot. Constructive versions of tarski's fixed point theorems., 1979. (Cited on page 45.)

[CE81] Edmund M. Clarke and E. Allen Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In Dexter Kozen, editor, *Logics of Programs, Workshop*, volume 131 of *LNCS*, pages 52–71. Springer, 1981. URL: http://dx.doi.org/10.1007/BFb0025774. (Cited on page 14.)

[CH00] Pierre-Louis Curien and Hugo Herbelin. The duality of computation. In Martin Odersky and Philip Wadler, editors, *Proceedings of the Fifth ACM SIGPLAN International Conference on Functional Programming (ICFP '00), Montreal, Canada, September 18-21, 2000.*, pages 233–243. ACM, 2000. URL: `http://doi.acm.org/10.1145/351240.351262`. (Cited on page 13.)

[Chu32] A. Church. *A Set of Postulates for the Foundation of Logic*, volume 33 of *Annals of Mathematics*. 1932. (Cited on page 11.)

[Chu36] Alonzo Church. A note on the entscheidungsproblem. *Journal of Symbolic Logic*, 1:40–44, 1936. (Cited on page 11.)

[Cla09] Pierre Clairambault. Least and greatest fixpoints in game semantics. In Luca de Alfaro, editor, *FOSSACS '09*, volume 5504 of *LNCS*, pages 16–31. Springer, 2009. URL: `http://dx.doi.org/10.1007/978-3-642-00596-1_3`. (Cited on page 130.)

[CM10] Pierre-Louis Curien and Guillaume Munch-Maccagnoni. The duality of computation under focus. In Cristian S. Calude and Vladimiro Sassone, editors, *Theoretical Computer Science - 6th IFIP TC 1/WG 2.2 International Conference, TCS 2010, Held as Part of WCC 2010, Brisbane, Australia, September 20-23, 2010. Proceedings*, volume 323 of *IFIP Advances in Information and Communication Technology*, pages 165–181. Springer, 2010. URL: `http://dx.doi.org/10.1007/978-3-642-15240-5_13`. (Cited on pages 13 and 110.)

[Dam92] Mads Dam. Fixed points of büchi automata. In *Foundations of Software Technology and Theoretical Computer Science, 12th Conference, New Delhi, India, December 18-20, 1992, Proceedings*, pages 39–50, 1992. URL: `http://dx.doi.org/10.1007/3-540-56287-7_93`. (Cited on pages 213 and 214.)

[DG02] Mads Dam and Dilian Gurov. μ-calculus with explicit points and approximations. *J. Log. Comput.*, 12(2):255–269, 2002. URL: `http://dx.doi.org/10.1093/logcom/12.2.255`. (Cited on page 19.)

[DHL06] Christian Dax, Martin Hofmann, and Martin Lange. A proof system for the linear time μ-calculus. In *FSTTCS 2006, Kolkata, India, December 13-15, 2006, Proceedings*, pages 273–284, 2006. (Cited on pages 19, 61, 68, 150, 154, and 157.)

[DJS97] Vincent Danos, Jean-Baptiste Joinet, and Harold Schellinx. A new deconstructive logic: Linear logic. *J. Symb. Log.*, 62(3):755–807, 1997. URL: `http://dx.doi.org/10.2307/2275572`. (Cited on pages 13 and 110.)

[EJ91] E. Allen Emerson and Charanjit S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *32nd Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 1-4 October 1991*, pages 368–377, 1991. URL: `http://dx.doi.org/10.1109/SFCS.1991.185392`. (Cited on pages 16 and 159.)

[FFKD87] Matthias Felleisen, Daniel P. Friedman, Eugene E. Kohlbecker, and Bruce F. Duba. A syntactic theory of sequential control. *Theor. Comput. Sci.*, 52:205–237, 1987. URL: http://dx.doi.org/10.1016/0304-3975(87)90109-5. (Cited on page 12.)

[FH02] Claudia Faggian and Martin Hyland. Designs, disputes and strategies. In *Computer Science Logic, 16th International Workshop, CSL 2002, 11th Annual Conference of the EACSL, Edinburgh, Scotland, UK*, volume 2471 of *Lecture Notes in Computer Science*, pages 442–457. Springer, 2002. URL: http://dx.doi.org/10.1007/3-540-45793-3_30. (Cited on page 130.)

[FL12] Oliver Friedmann and Martin Lange. Ramsey-based analysis of parity automata. In Cormac Flanagan and Barbara König, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 18th International Conference, TACAS 2012, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2012, Tallinn, Estonia, March 24 - April 1, 2012. Proceedings*, volume 7214 of *Lecture Notes in Computer Science*, pages 64–78. Springer, 2012. URL: http://dx.doi.org/10.1007/978-3-642-28756-5_6. (Cited on page 81.)

[FS13] Jérôme Fortier and Luigi Santocanale. Cuts for circular proofs: semantics and cut-elimination. In Simona Ronchi Della Rocca, editor, *Computer Science Logic 2013 (CSL 2013), CSL 2013, September 2-5, 2013, Torino, Italy*, volume 23 of *LIPIcs*, pages 248–262. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013. (Cited on pages 20, 70, and 88.)

[Gen35] G. Gentzen. Untersuchungen über das logische schließen ii. *Mathematische Zeitschrift*, 39:405–431, 1935. URL: http://eudml.org/doc/168556. (Cited on page 27.)

[Gir72] Jean-Yves Girard. *Interprétation fonctionnelle et élimination des coupures de l'arithmétique d'ordre supérieur*. Thèse de doctorat d'etat, Université Paris VII, 1972. (Cited on page 12.)

[Gir87] Jean-Yves Girard. Linear logic. *Theor. Comput. Sci.*, 50:1–102, 1987. URL: http://dx.doi.org/10.1016/0304-3975(87)90045-4. (Cited on pages 13 and 35.)

[Gir01] Jean-Yves Girard. Locus solum: From the rules of logic to the logic of rules. *Mathematical Structures in Computer Science*, 11(3):301–506, 2001. URL: http://dx.doi.org/10.1017/S096012950100336X. (Cited on pages 22, 33, 122, 129, and 137.)

[Gri90] Timothy Griffin. A formulae-as-types notion of control. In Frances E. Allen, editor, *Conference Record of the Seventeenth Annual ACM Symposium on Principles of Programming Languages, San Francisco, California, USA, January 1990*, pages 47–58. ACM Press, 1990. URL: http://doi.acm.org/10.1145/96709.96714. (Cited on page 12.)

[GTL89] Jean-Yves Girard, Paul Taylor, and Yves Lafont. *Proofs and Types*. C.U.P., 1989. (Cited on page 16.)

[GTW02a] Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata Logics, and Infinite Games: A Guide to Current Research*. Springer-Verlag New York, Inc., New York, NY, USA, 2002. (Cited on page 166.)

[GTW02b] Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata Logics, and Infinite Games: A Guide to Current Research*. Springer-Verlag New York, Inc., New York, NY, USA, 2002. (Cited on page 198.)

[Gö31] Kurt Gödel. Über formal unentscheidbare sätze der principia mathematica und verwandter systeme i. *Monatshefte für Mathematik*, 38:173–198, 1931. (Cited on page 11.)

[HO00] Martin Hyland and Luke Ong. On full abstraction for PCF. *Information and Computation*, 163(2):285–408, December 2000. (Cited on pages 126 and 130.)

[How80] William A. Howard. The formulae-as-type notion of construction, 1969. In J. P. Seldin and R. Hindley, editors, *To H. B. Curry: Essays in Combinatory Logic, Lambda Calculus, and Formalism*, pages 479–490. Academic Press, New York, 1980. (Cited on page 12.)

[JW95] David Janin and Igor Walukiewicz. Automata for the modal mu-calculus and related results. In *Mathematical Foundations of Computer Science 1995, 20th International Symposium, MFCS'95, Prague, Czech Republic, August 28 - September 1, 1995, Proceedings*, pages 552–562, 1995. URL: `http://dx.doi.org/10.1007/3-540-60246-1_160`. (Cited on pages 169 and 171.)

[Kai95] Roope Kaivola. Axiomatising linear time mu-calculus. In *CONCUR '95: Concurrency Theory, 6th International Conference, Philadelphia, PA, USA, August 21-24, 1995, Proceedings*, pages 423–437, 1995. (Cited on pages 16, 23, 147, 148, 153, 155, 213, and 214.)

[Kle52] Stephen Cole Kleene. *Introduction to metamathematics*. Bibl. Matematica. North-Holland, Amsterdam, 1952. (Cited on pages 44 and 45.)

[Koz83] Dexter Kozen. Results on the propositional mu-calculus. *Theor. Comput. Sci.*, 27:333–354, 1983. URL: `http://dx.doi.org/10.1016/0304-3975(82)90125-6`. (Cited on pages 16, 147, 153, 158, and 213.)

[Kri09] Jean-Louis Krivine. Realizability in classical logic. *Panoramas et synthèses*, 27:197–229, 2009. URL: `https://hal.archives-ouvertes.fr/hal-00154500`. (Cited on page 126.)

[Lan05] Martin Lange. Weak automata for the linear time μ-calculus. In *Verification, Model Checking, and Abstract Interpretation, 6th International Conference, VMCAI 2005, Paris, France, January 17-19, 2005, Proceedings*, pages 267–281, 2005. URL: `http://dx.doi.org/10.1007/978-3-540-30579-8_18`. (Cited on pages 16 and 159.)

[Lau02] Olivier Laurent. *Etude de la polarisation en logique*. Thèse de doctorat, Université Aix-Marseille II, March 2002. (Cited on pages 13, 34, and 110.)

[Lau04a] Olivier Laurent. Polarized games. *Ann. Pure Appl. Logic*, 130(1-3):79–123, 2004. URL: http://dx.doi.org/10.1016/j.apal.2004.04.006. (Cited on page 110.)

[Lau04b] Olivier Laurent. A proof of the focalization property of linear logic. Unpublished note, May 2004. (Cited on page 110.)

[Mad94] Dam Mads. Temporal logics, automata, and classical theories. *Notes for the sixth European Summer School in Logic, Language, and Information*, 1994. (Cited on pages 16 and 159.)

[Mat99] Ralph Matthes. Monotone (co)inductive types and positive fixed-point types. *RAIRO - Theoretical Informatics and Applications*, 33(4-5):309–328, 1999. (Cited on page 16.)

[Men91] N. P. Mendler. Inductive types and type constraints in the second-order lambda calculus. *Ann. Pure Appl. Logic*, 51(1-2):159–172, 1991. URL: http://dx.doi.org/10.1016/0168-0072(91)90069-X. (Cited on page 18.)

[Mil96] Dale Miller. Forum: A multiple-conclusion specification logic. *Theor. Comput. Sci.*, 165(1):201–232, 1996. URL: http://dx.doi.org/10.1016/0304-3975(96)00045-X. (Cited on page 109.)

[Mil11] Dale Miller. A proposal for broad spectrum proof certificates. In Jean-Pierre Jouannaud and Zhong Shao, editors, *Certified Programs and Proofs - First International Conference, CPP 2011, Kenting, Taiwan, December 7-9, 2011. Proceedings*, volume 7086 of *Lecture Notes in Computer Science*, pages 54–69. Springer, 2011. URL: http://dx.doi.org/10.1007/978-3-642-25379-9_6. (Cited on page 14.)

[ML71] Per Martin-Löf. Hauptsatz for the theory of species. *Studies in Logic and the Foundations of Mathematics*, 63:217 – 233, 1971. URL: http://www.sciencedirect.com/science/article/pii/S0049237X08708486. (Cited on page 18.)

[MM00] Raymond McDowell and Dale Miller. Cut-elimination for a logic with definitions and induction. *Theor. Comput. Sci.*, 232(1-2):91–119, 2000. URL: http://dx.doi.org/10.1016/S0304-3975(99)00171-1. (Cited on page 18.)

[MM09] Guillaume Munch-Maccagnoni. Focalisation and Classical Realisability. In Erich Grädel and Reinhard Kahle, editors, *Computer Science Logic '09*, volume 5771 of *Lecture Notes in Computer Science*, pages 409–423. Springer, Heidelberg, 2009. (Cited on page 13.)

[MN12] Dale Miller and Gopalan Nadathur. *Programming with Higher-Order Logic*. Cambridge University Press, 2012. URL: http://www.cambridge.org/de/academic/subjects/

computer-science/programming-languages-and-applied-logic/programming-higher-order-logic?format=HB. (Cited on page 13.)

[MNPS91] Dale Miller, Gopalan Nadathur, Frank Pfenning, and Andre Scedrov. Uniform proofs as a foundation for logic programming. *Ann. Pure Appl. Logic*, 51(1-2):125–157, 1991. URL: http://dx.doi.org/10.1016/0168-0072(91)90068-W. (Cited on page 13.)

[MP81] Zohar Manna and Amir Pnueli. Verification of the concurrent programs: the temporal framework. volume 398, pages 215–273. Academic Press, 1981. (Cited on page 14.)

[MS07] Dale Miller and Alexis Saurin. From proofs to focused proofs: A modular proof of focalization in linear logic. In Jacques Duparc and Thomas A. Henzinger, editors, *Computer Science Logic, 21st International Workshop, CSL 2007, 16th Annual Conference of the EACSL, Lausanne, Switzerland, September 11-15, 2007, Proceedings*, volume 4646 of *Lecture Notes in Computer Science*, pages 405–419. Springer, 2007. URL: http://dx.doi.org/10.1007/978-3-540-74915-8_31. (Cited on pages 110, 114, 115, 116, and 117.)

[MT10] Paul-André Melliès and Nicolas Tabareau. Resource modalities in tensor logic. *Ann. Pure Appl. Logic*, 161(5):632–653, 2010. URL: http://dx.doi.org/10.1016/j.apal.2009.07.018. (Cited on page 110.)

[NW96] Damian Niwinski and Igor Walukiewicz. Games for the mu-calculus. *Theor. Comput. Sci.*, 163(1&2):99–116, 1996. URL: http://dx.doi.org/10.1016/0304-3975(95)00136-0. (Cited on page 19.)

[Par92] Michel Parigot. Lambda-mu-calculus: An algorithmic interpretation of classical natural deduction. In Andrei Voronkov, editor, *Logic Programming and Automated Reasoning,International Conference LPAR'92, St. Petersburg, Russia, July 15-20, 1992, Proceedings*, volume 624 of *Lecture Notes in Computer Science*, pages 190–201. Springer, 1992. URL: http://dx.doi.org/10.1007/BFb0013061. (Cited on page 13.)

[Pit07] Nir Piterman. From nondeterministic büchi and streett automata to deterministic parity automata. *Logical Methods in Computer Science*, 3(3), 2007. URL: http://dx.doi.org/10.2168/LMCS-3(3:5)2007. (Cited on page 215.)

[Pnu77] Amir Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977*, pages 46–57, 1977. URL: http://dx.doi.org/10.1109/SFCS.1977.32. (Cited on pages 14 and 153.)

[Rey74] John C. Reynolds. Towards a theory of type structure. In Bernard Robinet, editor, *Programming Symposium, Proceedings Colloque sur la Programmation, Paris, France, April 9-11, 1974*, volume 19 of *Lecture Notes in Computer Science*, pages 408–423. Springer, 1974. (Cited on page 12.)

[Saf92] Shmuel Safra. Exponential determinization for ω-automata with strong-fairness acceptance condition (extended abstract). In *Proceedings of the Twenty-fourth Annual ACM Symposium on Theory of Computing*, STOC '92, pages 275–282, New York, NY, USA, 1992. ACM. URL: `http://doi.acm.org/10.1145/129712.129739`. (Cited on page 198.)

[San02] Luigi Santocanale. A calculus of circular proofs and its categorical semantics. In Mogens Nielsen and Uffe Engberg, editors, *FOSSACS '02*, volume 2303 of *LNCS*, pages 357–371. Springer, 2002. URL: `http://dx.doi.org/10.1007/3-540-45931-6_25`. (Cited on pages 19, 61, and 77.)

[Sha08] Natarajan Shankar. Trust and automation in verification tools. In Sung Deok Cha, Jin-Young Choi, Moonzoo Kim, Insup Lee, and Mahesh Viswanathan, editors, *Automated Technology for Verification and Analysis, 6th International Symposium, ATVA 2008, Seoul, Korea, October 20-23, 2008. Proceedings*, volume 5311 of *Lecture Notes in Computer Science*, pages 4–17. Springer, 2008. URL: `http://dx.doi.org/10.1007/978-3-540-88387-6_3`. (Cited on page 14.)

[Sim17] Alex Simpson. *Cyclic Arithmetic Is Equivalent to Peano Arithmetic*, pages 283–300. Springer Berlin Heidelberg, Berlin, Heidelberg, 2017. URL: `http://dx.doi.org/10.1007/978-3-662-54458-7_17`. (Cited on page 218.)

[SS86] Leon Sterling and Ehud Shapiro. *The Art of Prolog: Advanced Programming Techniques*. MIT Press, Cambridge, MA, USA, 1986. (Cited on page 13.)

[SV14] Sven Schewe and Thomas Varghese. Determinising parity automata. In *Mathematical Foundations of Computer Science 2014 - 39th International Symposium, MFCS 2014, Budapest, Hungary, August 25-29, 2014. Proceedings, Part I*, pages 486–498, 2014. URL: `http://dx.doi.org/10.1007/978-3-662-44522-8_41`. (Cited on page 208.)

[Tar35] Alfred Tarski. Der wahrheitsbegriff in den formalisierten sprachen. *Studia Philosophica*, (1):261–405, 1935. (Cited on page 11.)

[Tar55] Alfred Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific J. Math.*, 5(2):285–309, 1955. (Cited on page 44.)

[Ter03] Terese. *Term rewriting systems*. Cambridge tracts in theoretical computer science. Cambridge University Press, Cambridge, New York, 2003. URL: `http://opac.inria.fr/record=b1099883`. (Cited on page 93.)

[Ter11] Kazushige Terui. Computational ludics. *Theor. Comput. Sci.*, 412(20):2048–2071, 2011. URL: `http://dx.doi.org/10.1016/j.tcs.2010.12.026`. (Cited on pages 13, 22, 110, 121, 122, 126, 127, 128, 129, 130, and 131.)

[Tho97] Wolfgang Thomas. Handbook of formal languages, vol. 3. chapter Languages, Automata, and Logic, pages 389–455. Springer-Verlag New York, Inc., New York, NY, USA, 1997. URL: `http://dl.acm.org/citation.cfm?id=267871.267878`. (Cited on pages 185 and 186.)

[TM12] Alwen Tiu and Alberto Momigliano. Cut elimination for a logic with induction and co-induction. *J. Applied Logic*, 10(4):330–367, 2012. URL: `http://dx.doi.org/10.1016/j.jal.2012.07.007`. (Cited on page 18.)

[Tur36] Alan M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 2(42):230–265, 1936. (Cited on page 11.)

[Var95] Moshe Y. Vardi. An automata-theoretic approach to linear temporal logic. In *Logics for Concurrency - Structure versus Automata (8th Banff Higher Order Workshop, August 27 - September 3, 1995, Proceedings)*, pages 238–266, 1995. URL: `http://dx.doi.org/10.1007/3-540-60915-6_6`. (Cited on page 165.)

[VEK76] M. H. Van Emden and R. A. Kowalski. The semantics of predicate logic as a programming language. *J. ACM*, 23(4):733–742, October 1976. URL: `http://doi.acm.org/10.1145/321978.321991`. (Cited on page 13.)

[Wal93] Igor Walukiewicz. On completeness of the mu-calculus. In *Proceedings of the Eighth Annual Symposium on Logic in Computer Science (LICS '93), Montreal, Canada, June 19-23, 1993*, pages 136–146, 1993. URL: `http://dx.doi.org/10.1109/LICS.1993.287593`. (Cited on page 148.)

[Wal94] Igor Walukiewicz. *A complete deductive system for the mu-Calculus*. PhD thesis, Warsaw University, 1994. (Cited on pages 14 and 147.)

[Wal95] Igor Walukiewicz. Completeness of kozen's axiomatisation of the propositional mu-calculus. In *LICS 95, San Diego, California, USA, June 26-29, 1995*, pages 14–24, 1995. (Cited on pages 16, 19, 23, 61, 148, 153, 213, and 214.)

[Zei09] Noam Zeilberger. *The logical basis of evaluation order and pattern matching*. PhD thesis, The logical basis of evaluation order and pattern matching, 2009. (Cited on pages 13 and 110.)