LABELLED DEDUCTION

LABELLED DEDUCTION

Edited by
DAVID BASIN
University of Freiburg

MARCELLO D'AGOSTINO
University of Ferrara

DOV M. GABBAY
King's College

SEÁN MATTHEWS
Max-Planck-Insitut für Informatik

LUCA VIGANÒ
University of Freiburg

Kluwer Academic Publishers
Boston/Dordrecht/London

Contributing Authors

Author 1 is an Associate Professor ...

Author 2 is University Professor of ...

Introduction

David Basin, Marcello D'Agostino, Dov M. Gabbay, Seán Matthews, Luca Viganò

TO BE WRITTEN

Chapter 1

LABELLED PROOF SYSTEMS FOR INTUITIONISTIC PROVABILITY

Vincent Balat
Ecole Normale Supérieure
45, rue d'Ulm
75005, Paris, France
Vincent.Balat@ens.fr

Didier Galmiche
LORIA UMR 7503
Université Henri Poincaré
Campus scientifique, BP 239
54506 Vandœuvre-lès-Nancy, France
Didier.Galmiche@loria.fr

Abstract In this paper, we propose new labelled proof systems to analyse the intuitionistic provability in classical and linear logics. An important point is to understand how search in a non-classical logic can be viewed as a perturbation of search in classical logic. Therefore, suitable characterizations of intuitionistic provability and related labelled sequent calculi are defined for linear logic. An alternative approach, based on the notion of proof-net and on the definition of suitable labelled classical proof-nets, allows to directly study the intuitionistic provability by constructing intuitionistic proof-nets for sequents of classical linear logic.

Keywords: Proof theory, intuitionistic logic, linear logic, labelled sequent calculus, proof-nets, automated deduction.

1. INTRODUCTION

Many proof-search methods (sequent calculus, tableaux, resolution, connections) have been naturally developed in classical logic (**CL**) with a view to avoiding the possible redundancies, unsuccessful explorations or loops, as illustrated by naive search in sequent calculi. Some of them have been adapted to intuitionistic logic (**IL**) that can be viewed, from a proof-theoretic point of view, as a specialization of classical logic. Let us recall that a sequent calculus for **IL** can be directly obtained from the classical one by restricting the sequent conclusions to a single formula. Another equivalent multi-conclusion sequent calculus for **IL**, that is a specialisation of **CL** multi-conclusion sequent calculus with so-called special rules, has been proposed to more efficiently deal with intuitionistic proof-search from classical proof-search [13]. Then classical logic plays a pivotal role in search calculi and an interesting point consists in understanding how search in a non-classical logic can be viewed as a perturbation on search in classical logic. It has been recently studied in [30] by considering a proof-theoretic approach, based on $\lambda\mu$-terms [29], where the terms permit an axiomatisation of the perturbation which yields the non-classical search and also some semantical explanations about embedding in classical logic. In this setting, some questions arise: can we conclude from a classical proof of a given sequent that there exists an intuitionistic proof of it ? can we extract such a proof from the classical one ? is it possible to consider the intuitionistic search as a perturbation of classical search and to have a completeness result, i.e., to determine the intuitionistic provability of a sequent from any classical proof ? This paper deals with these questions mainly in the context of classical linear logic (**CLL**) that can be seen as a resource-sensitive refinement of **CL** [21, 22]. A central point is the choice of the proof systems to analyse the intuitionistic provability from classical search. We know that natural deduction is a suitable deduction system for **IL** but not for **CL** for which sequent calculus seems better suited. The Curry-Howard correspondence allows to annotate natural deductions with terms (for **IL** it yields to typed λ-calculus) but for sequent calculus, it is not clear what the appropriate notations are. For instance, Parigot has introduced a variant of multi-conclusion natural deduction and the related $\lambda\mu$-calculus, which seem suited for handling **CL** [29]. In the setting of linear logic, similar problems of representation arise, knowing that the specific notion of *proof-net* can moreover be seen as a counterpart of natural deduction. The relationships between classical linear logic (**CLL**) and intuitionistic linear logic (**ILL**) have to be studied from both proof-theoretic and proof-search points of view. Let us recall that **ILL** can be seen as a model of Petri nets [14, 16] and also as a foundation of functional and logic linear programming languages. Moreover recent specification logics are based on fragments of **ILL** (like Lolli [25]) or **CLL** (like Forum [28]), the choice being motivated by specific operational aims, for instance sequentiality in **ILL** and concurrency in **CLL**. Therefore, the study of intuitionistic provability from classical provability in linear logic would have an important impact on proof-search and semantics analysis.

In section 2, we study the intuitionistic provability from classical search by defining a new labelled proof system. This labelled sequent calculus for **CL** is an alternative to the based-on $\lambda\mu$-terms approach given in [30], being not type-theoretic but proof-search oriented. In both cases, there is no completeness result, i.e one cannot conclude the intuitionistic provability from any given classical proof. The main part of this paper deals with characterizations of the intuitionistic provability in the setting of linear logic. Therefore, in section 3, we summarize the main characteristics of classical and intuitionistic linear logics and focus on a particular intuitionistic system called (multiplicative) **FILL** (Full Intuitionistic Linear Logic) that simultaneously embodies features of concurrent logical computations, induced by the par connective and the sequential properties of intuitionistic linear implication [26]. In section 4, we then propose two labelled sequent calculi for (multiplicative) **CLL** with related characterizations of intuitionistic provability. The first one is a direct extension of the labelled **CL** sequent calculus to **CLL** but the provability characterization does not fit well from a logical point of view. The second one has a more suitable and useful definition of labels w.r.t. linear logic specificities. Moreover it can be independently considered as a new proof system for **FILL** without terms or pattern calculus [7, 26] that is more adapted to proof-search. As in the classical case, these systems do not give a positive answer to the above mentioned completeness problem. In section 5, we consider a new labelled proof system, based on the notion of *proof-net* and its possible use for proof-search. It allows to directly construct an intuitionistic proof, if there exits, from a classical search. For that, we define (multiplicative) **FILL** proof-nets that are in fact classical proof-nets with labels that represent some so-called dependency paths. Then we design an algorithm that, for a given **CLL** initial sequent, builds a **FILL** proof-net and also a sequent proof, if they exist. Therefore, the intuitionistic provability question can be successfully studied from labelled proof-nets and the intrinsic completeness problem arising with the labelled sequent calculi is solved. Finally, section 6 presents some concluding remarks.

2. INTUITIONISTIC PROVABILITY IN CLASSICAL LOGIC

The relationships between **CL** and **IL** are strong and then one can try to use the classical proof-search to characterize the intuitionistic provability. The intuitionistic logic is studied because of its underlying constructivism and its connections with the λ-calculus and programming (Curry-Howard isomorphism). A sequent calculus for **IL** can be obtained from classical sequent calculus of **CL** (see appendix 1.) by restricting sequents to a single conclusion (see appendix 2.). It is important to recall that there exists a multi-conclusion sequent calculus for **IL** (see appendix 3.), that is well adapted to study efficient proof-search [13, 31, 32]. Such a calculus is in fact a restriction of classical sequent calculus including some particular rules called *special rules*. It allows to naturally consider the intuitionistic provability like a perturbation of classical provability. Thus, we can consider the following question: is it possible to conclude to the existence of an intuitionistic proof from the existence of a classical proof ? An analysis of this problem has been developed in [30] where classical proofs are encoded in the $\lambda\mu$-calculus [29] that is an extension of λ-calculus related to the classical logic. Therefore, in a type-theoretic approach, a classical proof is represented with a $\lambda\mu$-term and a characterization of terms representing classical proofs that include an intuitionistic proof is given. But such a characterization is not complete, i.e., a sequent can be provable in **IL** but one cannot observe it from any classical proof. Another approach is the one of [11] with a tableau system that distinguishes classical and intuitionistic logics by comparing semantic labels after a proof is constructed. Is the analysis easier using $\lambda\mu$-terms, semantics labels or initial classical proofs ? Because we are more interested on proof-search than on provability, we propose to directly analyse the classical proofs to detect the existence of an underlying intuitionistic proof. Then we define a sequent calculus with labels that allows to memorize the necessary information to analyse intuitionistic provability.

2.1 A LABELLED SEQUENT CALCULUS FOR CL

We now present a new labelled system for classical proofs that is used to detect their intuitionistic character, with a very simple criterion[1]. The principle consists in annotating each formula with a specific label (or word), to keep a trace of the construction of the classical sequent proof. From a given labelled classical proof, the labels of the sequent axioms allow to characterize the intuitionistic provability.

Considering its multi-conclusion (propositional) version **IL** differs from **CL** only because of the so-called *special rules* $\rightarrow_R$ and $\neg_R$ [13, 32]. Let us analyse the $\rightarrow_R$ rule, respectively in multi-conclusion **CL** and **IL** versions :

$$\frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \rightarrow B, \Delta} \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B, \Delta}$$

For each application of the $\rightarrow_R$ rule, we mark all formulae of Δ with the 0 symbol and the A and B formulae with the x symbol. The positions of the symbols in the labels memorize the different applications of such a special rule. In fact, we forbid a (sub)formula of A and a (sub)formula of one formula of Δ to meet in an axiom sequent. We apply the same principle to the $\neg_R$ rule. We thus consider the alphabet $\Sigma = \{1, x, 0\}$ and the sequents have the following

[1]The initial version of this system was discussed and developed with G. Delzanno, when he visited LORIA in Nancy during the year 1996/97.

$$\frac{}{\Gamma, A^v \vdash A^w, \Delta}\ ax \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ cut$$

$$\frac{\Gamma, A^v, A^v \vdash \Delta}{\Gamma, A^v \vdash \Delta}\ c_L \qquad \frac{\Gamma \vdash \Delta, A^v, A^v}{\Gamma \vdash \Delta, A^v}\ c_R$$

$$\frac{\Gamma \vdash A^v, \Delta \quad \Gamma, B^v \vdash \Delta}{\Gamma, (A \to B)^v \vdash \Delta}\ \to_L \qquad \frac{\Gamma^{1u}, A^{xv} \vdash B^{xv}, \Delta^{0w}}{\Gamma^u \vdash (A \to B)^v, \Delta^w}\ \to_R$$

$$\frac{\Gamma, A^v \vdash \Delta \quad \Gamma, B^v \vdash \Delta}{\Gamma, (A \vee B)^v \vdash \Delta}\ \vee_L \qquad \frac{\Gamma \vdash A^v, \Delta}{\Gamma \vdash (A \vee B)^v, \Delta}\ \vee_{R1} \qquad \frac{\Gamma \vdash B^v, \Delta}{\Gamma \vdash (A \vee B)^v, \Delta}\ \vee_{R2}$$

$$\frac{\Gamma, A^v, B^v \vdash \Delta}{\Gamma, (A \wedge B)^v \vdash \Delta}\ \wedge_L \qquad \frac{\Gamma \vdash A^v, \Delta \quad \Gamma \vdash B^v, \Delta}{\Gamma \vdash (A \wedge B)^v, \Delta}\ \wedge_R$$

$$\frac{\Gamma \vdash A^v, \Delta}{\Gamma, \neg A^v \vdash \Delta}\ \neg_L \qquad \frac{\Gamma^{1u}, A^{xv} \vdash \Delta^{0w}}{\Gamma^u \vdash \neg A^v, \Delta^w}\ \neg_R$$

Figure 1.1 Labelled sequent calculus for **CL**

form $A_1^{v_1}, ...A_k^{v_k} \vdash B_1^{w_1}, ...B_m^{w_m}$ where $v_1, ...v_k, w_1, ..., w_m$ are words of same length on Σ.

Notation 1 *$u[p]$ denotes the letter of the word u at position p, the letters being annotated from 1, from right-hand side to left-hand side. $|u|$ denotes the length of the word u and ε denotes the empty word. If $a \in \Sigma$ and $v = v[n]...v[1] \in \Sigma^*$, then av denotes the word $av[n]...v[1]$. If $\Gamma = A_1, ..., A_k$ is a list of formulae and $v = v_1, ..., v_k$ is a list of words, Γ^v denotes the set of labelled formulae $A_1^{v_1}, ...A_k^{v_k}$, and Γ^{av} denotes $A_1^{av_1}, ...A_k^{av_k}$.*

At the beginning, the formulae of the sequent to prove are labelled with the empty word. The labelled sequent calculus for **CL** is given in Figure 1.1. When labels are omitted it means that they are unchanged.

Definition 2 *Let $\Gamma \vdash \Delta$ be a labelled sequent, A^u (resp. A^v) a labelled formula of Γ (resp. Δ), and p an integer such that $0 < p \leq min(|u|, |v|)$, the pair $(u[p], v[p])$ is called a* correspondence at position *p on the sequent.*

Definition 3 *A set of axiom sequents is said to be* intuitionistic *if, for any position p, there is no correspondence of the form $(x, 0)$ or $(0, x)$.*

Theorem 4 *If a sequent $\Gamma \vdash \Delta$ has a proof (in the system Figure 1.1) such that the set of axiom sequents is intuitionistic, then there exists an intuitionistic proof of the sequent.*

Proof It is a direct consequence of the relationships between what a correspondence of the form $(x, 0)$ or $(0, x)$ means and the application of so-called special rules in the multi-conclusion sequent calculus of **IL** (see appendix 3.). ∎

Then, this characterization of intuitionistic provability depends on syntactic labels (that are words) and mainly on the notion of position in such labels.

2.2 THE COMPLETENESS PROBLEM

Let us first consider a classical proof that includes an intuitionistic proof.

$$
\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma^1, A^x \vdash A^1, B^x, B^0}\,ax \quad \dfrac{}{\Gamma^1, A^x, B^1 \vdash B^x, B^0}\,ax}{\Gamma^1, A^x, \underline{A \to B}^1 \vdash B^x, B^0}\,\to_L}{\Gamma, A \to B \vdash \underline{A \to B}, B}\,\to_R \qquad \dfrac{}{\Gamma, A \to B, B \vdash B}\,ax}{\Gamma, A \to B, \underline{(A \to B) \to B} \vdash B}\,\to_L
$$

Let us notice that if the inference rules are applied in another order, it is no more possible to claim, by application of the previous criterion, that there exists an intuitionistic proof of such a sequent (because of a $(x, 0)$ correspondence on the formula A of one axiom sequent).

$$
\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma^1, A^x \vdash B^x, A^0, B^0}\,ax}{\Gamma \vdash \underline{A \to B}, A, B}\,\to_R \quad \dfrac{}{\Gamma, B \vdash A, B}\,ax}{\Gamma, \underline{(A \to B) \to B} \vdash A, B}\,\to_L \qquad \dfrac{}{\Gamma, (A \to B) \to B, B \vdash B}\,ax}{\Gamma, \underline{A \to B}, (A \to B) \to B \vdash B}\,\to_L
$$

We have no completeness result, i.e, for a given sequent, we cannot conclude from any classical proof if it is also provable in **IL**. It is well illustrated with both previous proofs. It is also the case in the type-theoretic approach proposed in [30] where classical logic can be viewed as a type theory based on an extension of the Parigot's $\lambda\mu$-calculus [29]. Conditions for intuitionistic provability are expressed on classical $\lambda\mu$-terms and are also equivalent to admissible conditions on prefix unifiers in [32]. The use of such terms for the axiomatisation of the perturbation which yields non-classical search, allows to have semantical explanations of syntactic tricks to embed non-classical search in classical systems. Our equivalent approach, based on syntactic labels in a sequent calculus, is nevertheless simple and readable from both algorithmic and proof-search points of view.

When a classical proof contains an intuitionistic proof, we can use the information given by the labels to extract an intuitionistic proof. The principle consists in detecting the needless formulae in the classical proof by an analysis of the labels and then in erasing them. A simple algorithm can be proposed for this operation. The main advantages and limits of using such labelled systems will be illustrated in the case of linear logic, for which the based-on $\lambda\mu$-calculus approach could be also used at first to devise a natural deduction formulation of classical linear logic [8].

3. LINEAR LOGIC

Linear logic is a resource-sensitive logic that can be seen as a refinement of classical logic sequent calculus where contraction and weakening rules are forbidden [22]. Both classical linear logic (**CLL**) and its intuitionistic fragment, called intuitionistic linear logic (**ILL**), are studied as models of computation [2] and of semantics of parallelism [20].

3.1 CLASSICAL LINEAR LOGIC

The usual presentation of **CLL** consists in starting from the **CL** sequent calculus with suppression of the contraction and weakening rules. Then the equivalence between the additive and multiplicative presentations of the $\vee$ and $\wedge$ connectives disappears and then leads to distinguish a multiplicative $\vee$ ("par", denoted $\parr$), an additive $\vee$ ("with", denoted $\&$), a multiplicative $\wedge$ ("times", denoted $\otimes$), and an additive $\wedge$ ("plus", denoted $\oplus$). The atoms have the form A or $A^\perp$. The definition of the negation $()^\perp$, that is involutive, is extended to all formulae from the following rules: $(A \parr B)^\perp = (A^\perp \otimes B^\perp)$, $(A \otimes B)^\perp = (A^\perp \parr B^\perp)$, $(A \oplus B)^\perp = (A^\perp \& B^\perp)$ and $(A \& B)^\perp = (A^\perp \oplus B^\perp)$. The symbols $\mathbf{1}$, $\mathbf{0}$, $\top$ and $\perp$ respectively represent the neutrals of the $\otimes$, $\oplus$, $\&$ and $\parr$ connectives. The linear implication ($\multimap$) is defined by $A \multimap B = A^\perp \parr B$. The ! ("of course") and ? ("why not") connectives (called exponentials) are used to reintroduce in a controlled way the weakening and the contraction rules (but as logical rules). Let us mention that the intuitionistic implication $A \rightarrow B$ can be, with such operators, decomposed into two operations, i.e. $!A \multimap B$. See appendix 4. to have the rules of propositional **CLL**. The Multiplicative Linear Logic (only with the multiplicative connectives) without constants, is denoted **MLL**.

3.2 INTUITIONISTIC LINEAR LOGIC

In linear logic, compared to classical logic, the definition of an intuitionistic fragment is not so natural and direct from the initial sequent calculus. The Intuitionistic Linear Logic (**ILL**) has a mono-conclusion sequent calculus (see appendix 5.). Compared to the non-linear case, **ILL** is defined with a formula grammar that is different from the **CLL** one.
In fact, the rules corresponding to $\parr$, its neutral element $\perp$, the negation $^\perp$ and to ? led to consider a multi-conclusion representation. Then the grammar of formulae in **ILL** is $\phi ::= p \mid \phi \otimes \phi \mid \phi \multimap \phi \mid \phi \& \phi \mid \phi \oplus \phi \mid !\phi$, where p is an atomic formula or a constant $\top$, $\mathbf{0}$ or $\mathbf{1}$. In [24], Girard and Lafont illustrate the interest of linear logic, compared to **IL** to represent types of functions and linear λ-terms have been defined in this context [27]. Therefore, the $\otimes$ connective is a strict $\wedge$ while the $\&$ connective is a lazy $\wedge$. In this setting, the

real meaning of the $\parr$ connective is not clear with this interpretation of proofs as functions. The relationship between **CLL** and **ILL** can be expressed through the following result

Theorem 5 *Let be S a sequent built from the* **ILL** *(without constants) grammar, if S is provable in* **CLL** *then it is provable in* **ILL**.

Proof Let us consider a sequent provable in **CLL**, that only uses the **ILL** grammar. We first show that all sequents, in such a **CLL** proof, have a single formula as conclusion. For that, we observe that the axioms are mono-conclusion and then for each rule of **CLL**, if the premises are mono-conclusion then the conclusion is also mono-conclusion. Secondly, we show that a **CLL** rule applied to a mono-conclusion sequent gives the same result than the **ILL** corresponding rule. ■

Let us note that if we consider a version of **CLL** including the rules for the $\multimap$ connective then the **ILL** proof will be identical to the **CLL** proof. Works on **ILL** have been developed from linear λ-calculus [6, 8, 24], but the lack of the $\parr$ connective seems more motivated by technical reasons and then we could ask for an intuitionistic fragment of linear logic including it. In this context, the semantics of $\parr$ has to be clarified even if a natural interpretation seems to be connected to the concurrent execution of processes [1, 9].

3.3 FULL INTUITIONISTIC LINEAR LOGIC

A multi-conclusion intuitionistic multiplicative linear logic, called *Full Intuitionistic Linear Logic* (**FILL**) has been proposed by Hyland and de Paiva in [26]. That is a variant of intuitionistic linear logic whose logical connectives are all independent, i.e., not inter-derivable as they are in multiplicative **CLL**. It is analogous to the situation concerning the relationship between **IL** and **CL**. The interest of **FILL** is that it simultaneously embodies features of concurrent logical computations, induced by the $\parr$ connective, and the sequential properties of intuitionistic linear implication.

In **FILL** the negation $A^\perp$ is defined as $A \multimap \perp$ and it is not an involution. This system, that includes the multiplicative disjunction $\parr$, is a subsystem of **CLL** and a proof in **FILL** is a proof in **CLL** with a certain intuitionistic property. For instance, one has $A^\perp \parr B \vdash A \multimap B$ in **FILL** but not the converse. Moreover, the excluded middle $\vdash A^\perp \parr A$ cannot be proved in **FILL**. It is the same for $A^{\perp\perp} \vdash A$ and for $\vdash (A^\perp \parr A)^{\perp\perp}$. This last remark illustrates the difference from the situation in **IL** where $\vdash \neg A \vee A$ is not provable but $\neg\neg(\neg A \vee A))$ is provable. It comes from the use of contraction in **IL** and the effective difference between the $\vee_R$ and $\parr_R$ rules. Moreover, $(A \otimes B) \multimap (C \parr D) \vdash (A \multimap C) \parr (B \multimap D)$ is an example of a sequent provable in **CLL** but not in **FILL** (see [10] for more

details).
Let us mention also that the interaction between $\parr$ and the linear implication $\multimap$ is such that cut-elimination failed outright for some intuitive formulation of **FILL**. This problem has motivated, from the term assignment system in [26], some alternative presentations as the ones of Bierman [7] and Braüner and de Paiva [10]. The latter is based on a notion of *dependency* to capture, with a side condition of $\multimap_R$, the underlying notion of intuitionistic implication and to make the cut-elimination to go through in a straightforward manner. In fact, one defines, given a proof of $\Gamma, B \vdash A, \Delta$ in **CLL**, when the succedent formula occurrence A depends on the antecedent formula occurrence B. This notion of dependency between formulae occurrences allows to express the intuitionistic property that characterizes **FILL** [10]. The corresponding sequent calculus is given in Figure 1.2. Let τ be a **CLL** proof, for each formula C at the right-hand side of the sequents in τ, one defines the set $Dep_\tau(C)$ of the formulae of left-hand side on which it depends. In the rules, τ represents the proof-tree, the root of which is the conclusion of the rule, τ_1 (resp. τ_2) is the sub-proof, the root of which is the first (resp. second) premise, if it exists. Moreover the set of formulae occurrences $\Gamma[\Delta \mapsto \Delta']$ is defined as $(\Gamma \setminus \Delta) \cup \Delta'$ if $\Gamma \cap \Delta \neq \emptyset$ and as Γ otherwise. This notion of dependency allows a definition of **FILL** proof.

Definition 6 *A* **FILL** proof *is a proof of* **CLL** *such that, for any $\multimap_R$ rule and any formula C of Δ, we have $A \notin Dep_{\tau_1}(C)$ [10].*

Let us now consider the following proof:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{C \vdash C}\,ax \qquad \dfrac{}{B \vdash B}\,ax}{\underline{C \parr B} \vdash B, C}\,\parr_L \qquad \dfrac{}{A \vdash A}\,ax}{C \parr B, \underline{B \multimap A} \vdash C, A}\,\multimap_L}{B \multimap A \vdash \underline{(C \parr B) \multimap C}, A}\,\multimap_R$$

that is not a **FILL** proof. Here, the formula A in the right-hand side of the sequent $C \parr B, B \multimap A \vdash C, A$ depends on $C \parr B$. In fact we have $Dep(A) = \{C \parr B, B \multimap A\}$. The presentation of this system does not include a term assignment decorating the usual sequent calculus proofs like in [7, 26]. In the next section, we consider the relationships between **CLL** and **FILL** from both proof-theoretic and proof-search points of view. Then we propose new formulations of **FILL** with sequent calculi with labels that reflect the dependency conditions inside the inference rules of **CLL**.

4. INTUITIONISTIC PROVABILITY IN LINEAR LOGIC

Let us now consider the intuitionistic provability from classical search in the case of linear logic. With a proof in **ILL**, we can propose a simple criterion

$$\frac{}{A \vdash A}\ ax \qquad Dep_\tau(A) = A$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ cut \qquad Dep_\tau(C) = \begin{cases} Dep_{\tau_1}(C) & \text{if } C \in \Delta \\ Dep_{\tau_2}(C)[A \mapsto Dep_{\tau_1}(A)] & \text{if } C \in \Delta' \end{cases}$$

$$\frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \multimap B, \Delta}\ \multimap_R \qquad Dep_\tau(C) = \begin{cases} Dep_{\tau_1}(C)[A \mapsto \emptyset] & \text{if } C \in \Delta \\ Dep_{\tau_1}(B)[A \mapsto \emptyset] & \text{if } C = A \multimap B \end{cases}$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \multimap B \vdash \Delta, \Delta'}\ \multimap_L \qquad Dep_\tau(C) = \begin{cases} Dep_{\tau_1}(C) & \text{if } C \in \Delta \\ Dep_{\tau_2}(C)[B \mapsto (Dep_{\tau_1}(A) \cup (A \multimap B))] & \text{if } C \in \Delta' \end{cases}$$

$$\frac{\Gamma, A \vdash \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \parr B \vdash \Delta, \Delta'}\ \parr_L \qquad Dep_\tau(C) = \begin{cases} Dep_{\tau_1}(C)[A \mapsto (A \parr B)] & \text{if } C \in \Delta \\ Dep_{\tau_2}(C)[B \mapsto (A \parr B)] & \text{if } C \in \Delta' \end{cases}$$

$$\frac{\Gamma \vdash A, B, \Delta}{\Gamma \vdash A \parr B, \Delta}\ \parr_R \qquad Dep_\tau(C) = \begin{cases} Dep_{\tau_1}(C) & \text{if } C \in \Delta \\ Dep_{\tau_1}(A) \cup Dep_{\tau_1}(B) & \text{if } C = A \parr B \end{cases}$$

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \otimes B \vdash \Delta}\ \otimes_L \qquad Dep_\tau(C) = Dep_{\tau_1}(C)[A, B \mapsto A \otimes B]$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma' \vdash B, \Delta'}{\Gamma, \Gamma' \vdash A \otimes B, \Delta, \Delta'}\ \otimes_R \qquad Dep_\tau(C) = \begin{cases} Dep_{\tau_1}(C) & \text{if } C \in \Delta \\ Dep_{\tau_2}(C) & \text{if } C \in \Delta' \\ Dep_{\tau_1}(A) \cup Dep_{\tau_2}(B) & \text{if } C = A \otimes B \end{cases}$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, \mathbf{1} \vdash \Delta}\ \mathbf{1}_L \qquad Dep_\tau(C) = Dep_{\tau_1}(C)$$

$$\frac{}{\vdash \mathbf{1}}\ \mathbf{1}_R \qquad Dep_\tau(\mathbf{1}) = \emptyset$$

$$\frac{}{\bot \vdash}\ \bot_R \qquad \text{nothing to define}$$

$$\frac{\Gamma \vdash \Delta}{\Gamma \vdash \bot, \Delta}\ \bot_L \qquad Dep_\tau(C) = \begin{cases} Dep_{\tau_1}(C) & \text{if } C \in \Delta \\ \emptyset & \text{if } C = \bot \end{cases}$$

Figure 1.2 a **FILL** sequent calculus

that corresponds to observe the presence of the $\parr$ and ? connectives and to test if the sequents have a single conclusion (see theorem 5). In **FILL**, the structure of the proofs is exactly the same than in **CLL** but we have to take into account the dependency relations. Compared to the previous study, we do not try to determine if a **CLL** proof contains a **FILL** proof but if it is a **FILL** proof. Among the different presentations of multiplicative **FILL** [7, 10, 26], mainly motivated by semantics analysis and by simple and readable cut-elimination proofs, the one of Braüner and de Paiva [10] appears to be the more related to our approach. In fact, our proposals can be seen as new proof systems for

FILL that could be useful for further works based on objects or processes [9]. We now present labelled proof systems for multiplicative **CLL**. The first one is a direct extension of the criterion defined for **CL** and the second one is more simple and appropriate to the case of linear logic. From now, we only consider the multiplicative part of the logical fragments.

4.1 A LABELLED SEQUENT CALCULUS

In this first proposal, the formulae are labelled with words but we have not only to simulate the use of special rules where some formulae are erased but also to effectively test the dependencies between formulae. As we consider transitive dependency chains that can have arbitrary length, we then use an infinite alphabet Σ including the 1, x, and 0 symbols. The position in the word is also used to determine where special rules have been applied. A formula is labelled by a list of words that allows several correspondences at the same position between two formulae.

Notation 7 *We denote $u :: l$ a list with u as head and l as tail. Lists are presented in extension with $;$ and for instance the list with the words u, v and w as elements is denoted $u; v; w$. Moreover $\emptyset$ denotes the empty list.*

At the beginning, each formula of the sequent to prove is labelled with the $\varepsilon :: \emptyset$ list (the empty word is the only element). The labelled sequent calculus is given in Figure 1.3, where labels are omitted when they are unchanged and n is the length of the head word for any conclusion formula. Let us note that, for a given n, the words a^n and 1^n allow to guarantee that the head words have all the same length. Moreover, in $\multimap_R$ we pad out the label of A with a safe number of 1. The notion of correspondence of definition 2 is extended to this new sequent calculus and leads to a suitable definition of *dependency chain* on sequences[2] of sequents.

Definition 8 *Let E be a sequence of sequents, a* dependency chain on E at position p between the letters α and β *is a sequence of pairs of the form $(\alpha, \alpha_1), (\alpha_1, \alpha_2), ..., (\alpha_n, \beta)$ ($n \in \mathbb{N}^*$), the elements of which are correspondences at position p in one of the sequents of E.*

Definition 9 *A sequence of sequents is said to be* intuitionistic *if for any sequent there exists no position p with a dependency chain between x and 0.*

Theorem 10 *Let P be a proof (in the system Figure 1.3), P is a* **FILL** *proof if and only if the sequence of its axiom sequents is intuitionistic.*

[2]We prefer to deal with sequences and not with sets mainly because of the further management of axiom sequents extracted from proofs.

$$\dfrac{\Gamma^{1u::l_1}, A^{x1^n::\emptyset} \vdash B^{1v::l_2}, \Delta^{0w::l_3}}{\Gamma^{u::l_1} \vdash (A \multimap B)^{v::l_2}, \Delta^{w::l_3}}\ \multimap_R$$

$$\dfrac{\Gamma \vdash A^{a^n::\emptyset}, \Delta \quad \Gamma', A^{a^n::\emptyset} \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ cut \qquad \text{with } a \text{ fresh letter}$$

$$\dfrac{\Gamma \vdash A^{a^n::\emptyset}, \Delta \quad \Gamma', B^{v::a^n::l} \vdash \Delta'}{\Gamma, \Gamma', (A \multimap B)^{v::l} \vdash \Delta, \Delta'}\ \multimap_L \qquad \text{with } a \text{ fresh letter}$$

$$\dfrac{}{A^{l_1} \vdash A^{l_2}}\ ax$$

$$\dfrac{\Gamma \vdash \Delta}{\Gamma, \mathbf{1}^l \vdash \Delta}\ \mathbf{1}_L \qquad \dfrac{}{\vdash \mathbf{1}^l}\ \mathbf{1}_R \qquad \dfrac{}{\perp^l \vdash}\ \perp_L \qquad \dfrac{\Gamma \vdash \Delta}{\Gamma \vdash \perp^l, \Delta}\ \perp_R$$

$$\dfrac{\Gamma, A^l \vdash \Delta \quad \Gamma', B^l \vdash \Delta'}{\Gamma, \Gamma', (A \parr B)^l \vdash \Delta, \Delta'}\ \parr_L \qquad \dfrac{\Gamma \vdash A^l, B^l, \Delta}{\Gamma \vdash (A \parr B)^l, \Delta}\ \parr_R$$

$$\dfrac{\Gamma, A^l, B^l \vdash \Delta}{\Gamma, (A \otimes B)^l \vdash \Delta}\ \otimes_L \qquad \dfrac{\Gamma \vdash A^l, \Delta \quad \Gamma' \vdash B^l, \Delta'}{\Gamma, \Gamma' \vdash (A \otimes B)^l, \Delta, \Delta'}\ \otimes_R$$

Figure 1.3 A labelled sequent calculus for **CLL**.

Proof The proof of this theorem is a directed adaptation of the proof we will develop in the next section for the second and main characterization. ■

We now present a proof in **CLL** of the sequent $C \multimap B, D \multimap A \vdash (C \parr D) \multimap B, A$, that is not a **FILL** proof, because of a dependency chain $(x, y), (y, 0)$.

$$\dfrac{\dfrac{\dfrac{\dfrac{}{C^x \vdash C^z}\,ax \quad \dfrac{}{D^x \vdash D^y}\,ax}{\underline{(C \parr D)^x} \vdash C^z, D^y}\,\parr_L \quad \dfrac{}{B^{1;z} \vdash B^1}\,ax}{\underline{(C \multimap B)^1}, (C \parr D)^x \vdash D^y, B^1}\,\multimap_L \quad \dfrac{}{A^{1;y} \vdash A^0}\,ax}{\dfrac{(C \multimap B)^1, \underline{(D \multimap A)^1}, (C \parr D)^x \vdash B^1, A^0}{C \multimap B, D \multimap A \vdash \underline{(C \parr D) \multimap B}, A}\,\multimap_R}\,\multimap_L$$

In our first criterion, the position p in a word is used to know for which applications of the $\multimap_R$ the letters appeared. It avoids to mix the dependency chains. But this principle has the disadvantages to complicate the system presentation by keeping several words for each formula and by creating words of the form y^n because we do not know at which position the y will be used. This proposal is then based on the notion of position but in several words.

4.2 ANOTHER LABELLED SEQUENT CALCULUS

An alternative, not directly connected to our initial criterion, consists in using a different marking for each special rule. In this case, the position p in a label has no more importance, the words can have different lengths and we only need one word per formula. The alphabet is then $\Sigma = \Sigma_1 \cup \Sigma_2$, with Σ_1

$$\frac{\Gamma, A^x \vdash B^v, \Delta^{xw}}{\Gamma \vdash (A \multimap B)^v, \Delta^w} \multimap_R \qquad \text{with } x \text{ fresh letter of } \Sigma_1$$

$$\frac{\Gamma \vdash A^y, \Delta \quad \Gamma', A^y \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ cut \qquad \text{with } y \text{ fresh letter of } \Sigma_2$$

$$\frac{\Gamma \vdash A^y, \Delta \quad \Gamma', B^{yv} \vdash \Delta'}{\Gamma, \Gamma', (A \multimap B)^v \vdash \Delta, \Delta'} \multimap_L \qquad \text{with } y \text{ fresh letter of } \Sigma_2$$

$$\frac{}{A^v \vdash A^w}\ ax$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, \mathbf{1}^v \vdash \Delta}\ \mathbf{1}_L \qquad \frac{}{\vdash \mathbf{1}^v}\ \mathbf{1}_R \qquad \frac{}{\bot^v \vdash}\ \bot_L \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \bot^v, \Delta}\ \bot_R$$

$$\frac{\Gamma, A^v \vdash \Delta \quad \Gamma', B^v \vdash \Delta'}{\Gamma, \Gamma', (A \parr B)^v \vdash \Delta, \Delta'} \parr_L \qquad \frac{\Gamma \vdash A^v, B^v, \Delta}{\Gamma \vdash (A \parr B)^v, \Delta} \parr_R$$

$$\frac{\Gamma, A^v, B^v \vdash \Delta}{\Gamma, (A \otimes B)^v \vdash \Delta} \otimes_L \qquad \frac{\Gamma \vdash A^v, \Delta \quad \Gamma' \vdash B^v, \Delta'}{\Gamma, \Gamma' \vdash (A \otimes B)^v, \Delta, \Delta'} \otimes_R$$

Figure 1.4 Another labelled sequent calculus for **CLL**.

and Σ_2 being disjoint. In fact, Σ_1 is used for the extremities of the dependency chains and Σ_2 for the intermediate symbols.
We begin with the empty word on each formula of the given sequent. The rules of this new labelled sequent calculus are given in Figure 1.4. As before, labels are omitted when they are unchanged. As the notion of position has no more significance here, then the appropriate notions of *correspondence* and of *dependency chains* are defined in the following way:

Definition 11 *A* correspondence *on a sequent* $\Gamma \vdash \Delta$ *is a pair* (α, β)*, where* α *is a letter in a label of a formula in* Γ *and* β *a letter of a label of the same formula in* Δ.

Definition 12 *Let* E *be a sequence of sequents, and let* $S_0, ..., S_n$ *be a subsequence of* E*, a* dependency chain on E between α and β *is a sequence of pairs of the form* (α_0, α_1), (α_1, α_2),...,(α_n, α_{n+1}) *(*$n \in \mathbb{N}^*$*) such that* $\alpha_0 = \alpha$, $\alpha_{n+1} = \beta$*, and* (α_i, α_{i+1}) *is a correspondence on* S_i.

4.2.1 Some properties. Let us begin to prove some properties that are useful to understand the behaviour of the labels in this setting.

Property 13 *Let* P *be a proof (in the system of Figure 1.4) of* $\Gamma \vdash \Delta$*, if the conclusion of a subproof contains a letter* $\alpha \in \Sigma_2$ *then* α *is always on the same side in all sequents of* P.

Let us also mention that if (α, β) is a correspondence on a sequent S inside a proof then all symbols in a dependency chain between α and β are in fact introduced in the subproof, the conclusion of which is S.

Property 14 *Let P be a proof (in the system of Figure 1.4) of $\Gamma \vdash \Delta$ with a dependency chain between α and β on the axiom sequents; for any subproof that contains this chain, if its conclusion has a correspondence (α, β) then the formula labelled with β depends (in the sense of 1.2) on the one labelled with α.*

Proof By induction on the depth of the proof tree.
Base case: $n = 1$; the property is verified for the axiom rules.
Induction case: Let us assume that the property is verified for n and consider the last inference rule applied in the proof.
- For the rules with one premise, it is trivial.
- For the rules with two premises, we have the following cases:
i) for the $\invamp_L$ and $\otimes_R$ rules, we verify that the dependency chain is completely included in one of the subtrees.
ii) for the cut and $\multimap_L$, it is the same proof argument if the new introduced letter y is not in the dependency chain. Else if it appears in the chain and if there is a correspondence (α, β) at the root, then the subchain between α and y (resp. y and β) is completely included in the left (resp. right) subtree.
By the induction hypothesis we have dependencies between the formulae labelled by α and y in the left-hand side premise and by y and β in the right-hand side premise. The result is then deduced from an analysis of the rule of the dependencies calculus. ■

Property 15 *Let P be a proof (in the system of Figure 1.4) of $\Gamma, E \vdash F, \Delta$, such that $E \in Dep(F)$; if the formulae E and F are in a correspondence (α, β), then there exists a dependency chain between α and β (on the sequence of axiom sequents).*

Proof By induction on the depth of the proof tree, with verification of the property for each rule. ■

From these properties, we can deduce the following results:

Lemma 16 *Let P be a proof (in the system of Figure 1.4) of a sequent $\Gamma \vdash \Delta$,*
(i) If there exists a dependency chain on the axiom sequents between two occurrences of the same letter of Σ_1, then the dependency condition (of definition 6) is not verified where this letter appears.
(ii) Let SP be a subproof, the last rule of which is $\multimap_R$, and such that there exists $A \in \Delta$ verifying $B \in Dep(A)$, then SP has a dependency chain between two occurrences of the same letter of Σ_2.

Proof Both results are direct consequences of the previous properties. ■

4.2.2 The intuitionistic characterization. These preliminary results are necessary to prove the next theorem. Its proof is based on the equivalence between the criteria we define and the calculus of dependencies of the multiplicative **FILL** system.

Definition 17 *A sequence of sequents is said to be* intuitionistic *if there is no dependency chain between two occurrences of the same letter of* Σ_1.

Theorem 18 *A proof P (in the system of Figure 1.4) is a* **FILL** *proof if and only if the sequence of axiom sequents, ordered from left to right, is intuitionistic.*

Proof The proof is a direct consequence of the previous lemma. ■

This proof that corresponds to the criterion of section 4.2, can be easily adapted to the one of section 4.1. Let us note that the order of premises in each rule of Figure 1.4 has to be respected during the proof construction. Let us consider again the example of section 4.1 with this new characterization. The proof is **CLL** but not **FILL** because of a dependency chain $(x, y), (y, x)$. Here $x \in \Sigma_1$ and $(y, z) \in \Sigma_2^2$.

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{C^x \vdash C^z}\,ax \quad \dfrac{}{D^x \vdash D^y}\,ax}{\underline{(C \parr D)^x} \vdash C^z, D^y}\,\parr_L \quad \dfrac{}{B^z \vdash B}\,ax}{\underline{C \multimap B}, (C \parr D)^x \vdash D^y, B}\,\multimap_L \quad \dfrac{}{A^y \vdash A^x}\,ax}{C \multimap B, \underline{D \multimap A}, (C \parr D)^x \vdash B, A^x}\,\multimap_L}{C \multimap B, D \multimap A \vdash \underline{(C \parr D) \multimap B}, A}\,\multimap_R
$$

Let us now present an example that illustrates that a sequent can have a **CLL** proof that is a **FILL** proof but also another one that is not a **FILL** proof. Here, we only use one atomic formula A, the different occurrences of which that are distinguished with indices. Therefore we present two labelled proofs of the following sequent: $A_4 \parr A_5, A_8 \vdash A_1 \multimap (A_6 \otimes A_7), A_2 \otimes A_3$.
The first proof is not a **FILL** proof because of a correspondence (x, x) in one of the axioms.

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A_1^x \vdash A_2^x}\,ax \quad \dfrac{}{A_4 \vdash A_3^x}\,ax}{A_1^x, A_4 \vdash \underline{(A_2 \otimes A_3)^x}}\,\otimes_{R-2,3} \quad \dfrac{}{A_5 \vdash A_6}\,ax}{A_1^x, \underline{A_4 \parr A_5} \vdash A_6, (A_2 \otimes A_3)^x}\,\parr_L \quad \dfrac{}{A_8 \vdash A_7}\,ax}{A_1^x, A_4 \parr A_5, A_8 \vdash \underline{A_6 \otimes A_7}, (A_2 \otimes A_3)^x}\,\otimes_{R-6,7}}{A_4 \parr A_5, A_8 \vdash \underline{A_1 \multimap (A_6 \otimes A_7)}, A_2 \otimes A_3}\,\multimap_R
$$

One can easily verify that $A_1 \in Dep(A_2 \otimes A_3)$ in the premise of the $\multimap_R$ rule. In the following proof

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A_1^x \vdash A_6}\,ax \quad \dfrac{}{A_5 \vdash A_7}\,ax}{A_1^x, A_5 \vdash \underline{A_6 \otimes A_7}}\,\otimes_{R-6,7} \quad \dfrac{}{A_4 \vdash A_3^x}\,ax}{A_1^x, \underline{A_4 \parr A_5} \vdash A_3^x, A_6 \otimes A_7}\,\parr_L \quad \dfrac{}{A_8 \vdash A_2^x}\,ax}{A_1^x, A_4 \parr A_5, A_8 \vdash \underline{(A_2 \otimes A_3)^x}, A_6 \otimes A_7}\,\otimes_{R-2,3}}{A_4 \parr A_5, A_8 \vdash \underline{A_1 \multimap (A_6 \otimes A_7)}, A_2 \otimes A_3}\,\multimap_R$$

there is no correspondence in the axiom sequents and consequently no dependency chain. Thus it is a **FILL** proof.

4.3 AN ALGORITHM FOR PROOF CONSTRUCTION

From the theorem 18 and this intuitionistic characterization, we can easily derive a corresponding algorithm. In fact, if we have a **CLL** proof built with the labelled sequent calculus then it is sufficient to analyse the labels of the axiom sequents and to do the test of the dependency chains. Such an algorithm could be presented in the following way:

For each sequent S in which a $x \in \Sigma_1$ is a label of a left-hand side formula, one call the procedure `Search_chain(`x`,`S`,`x`)` that is defined as by

Procedure `Search_chain(origin,sequent,current_letter)`

For all correspondences (`current_letter`$, \alpha$),

- if $\alpha =$ `origin` then FAILURE
- else, for all axiom sequents S' on the right-hand side of `sequent`, call `Search_chain(origin,` S'`,` α`)`

It returns a failure message if and only if the given proof is not a **FILL** proof. In general, with the current examples, the size of the word is not too big and then the number of cases the algorithm has to deal with is reasonable. A study of the complexity could be interesting but would need some choices about implementation. Such investigations are out of the purposes of this paper.

4.4 SOME REMARKS

In fact, such a method allows to verify if a given **CLL** proof is also a **FILL** proof. An interesting point could be to analyse, in case of failure, how the labels can be used to propose some heuristics or proof transformations, to obtain more directly and efficiently some **FILL** proofs, if they exist, by classical search. For instance, in the proofs of section 4.2.2, we observe that the application of the $\otimes_R$ rule on a marked formula allows to eliminate one possibility of correspondence.
Bierman has analysed with Parigot's techniques the relationships between **ILL** and **CLL** to propose a classical linear λ-calculus. This work appears fruitful from the resulting programming language point of view and of more use than such a similar language based on proof-nets [8]. Therefore we could consider

an approach similar to [30] by using a linear λ-calculus to encode the proofs and then analyse if they are "intuitionistic". This approach seems difficult to implement and to effectively use. Anyway, an alternative could be based on the notion of *proof-net* that can be seen as a counterpart of natural deduction in linear logic [23]. It could be an appropriate semantical and syntactical representation of proof-objects (or terms) for the analysis of intuitionistic provability.

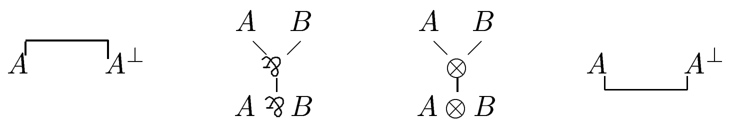
5. PROOF-NETS AND INTUITIONISTIC PROVABILITY

We now propose an alternative approach to consider the intuitionistic provability in **CLL**. It is based on a specific structure of linear logic, i.e. *proof-net*, that was naturally introduced by Girard as a counterpart of natural deduction in linear logic [21].

5.1 SOME DEFINITIONS

A proof-net is a particular graph, the nodes of which are formulae of linear logic [23]. It is naturally defined for the multiplicative linear logic **MLL** (without constants) as recalled in this subsection and can be seen as an efficient tool for automated deduction in such a fragment [15]. There are also proof-nets definitions for other **CLL** fragments, like **MALL** (Multiplicative and Additive Linear Logic) [23] but they are more complicated to handle.

Definition 19 *A* proof-structure *is a graph, the vertices of which are formulae, inductively built from the following substructures (called* links*) :*

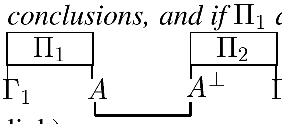


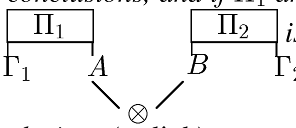
where A and B are formulae. In such a graph, any formula is the premise of at least one link and the conclusion of exactly one link.

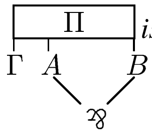
A proof-structure can be also seen as the set of the subformulae of the formula or sequent to prove (represented as a decomposition tree) plus a set of axiom-links such that each atom of the tree is the conclusion of exactly one axiom-link. A proof-net, for a given sequent, is in fact a proof-structure that corresponds to legal proofs [21]. There are several equivalent definitions of what a proof-net is, that are generally based on criteria that characterize the proof-structures that are proof-nets [21], like for instance the Danos-Regnier characterization [12]. The construction of a proof-net can be seen as the search of the connections (axiom-links) that characterize the provability of a given sequent [15].

To simplify the different representations, we often omit the formulae labelling the nodes and only represents the connectives. Here we recall the inductive definition of proof-net, introduced by Bellin [5].

Definition 20 *A* **MLL** proof-net *and its* set of conclusions *are inductively defined in the following way:*

- *If A is any multiplicative formula,* A $A^{\perp}$ *is a proof-net, with A and* $A^{\perp}$ *as conclusions* (axiom-link)
- *If* Π_1 *(resp.* Π_2*) is a proof-net with* $\Gamma_1 \cup \{A\}$ *(resp.* $\Gamma_2 \cup \{A^{\perp}\}$*) as conclusions, and if* Π_1 *and* Π_2 *are disjoint (not connected), then*


is a proof-net with $\Gamma_1 \cup \Gamma_2$ *as conclusions* (cut-link).
- *If* Π_1 *(resp.* Π_2*) is a proof-net with* $\Gamma_1 \cup \{A\}$ *(resp.* $\Gamma_2 \cup \{B\}$*) as conclusions, and if* Π_1 *and* Π_2 *are disjoint (not connected), then*


is a proof-net with $\Gamma_1 \cup \Gamma_2 \cup \{A \otimes B\}$ *as conclusions* ($\otimes$-link).
- *If* Π *is a proof-net with* $\Gamma \cup \{A, B\}$ *as conclusions, then*


is a proof-net with $\Gamma \cup \{A \parr B\}$ *as conclusions* ($\parr$-link).

Some reductions operations can be defined on proof-nets that correspond to the notion of reduction in λ-calculus. There exists a theorem of cut-elimination for the proof-nets [23] and from now we only consider proof-nets without cut-links. Our goal is to define intuitionistic proof-nets from the classical proof-nets. Let us recall that the order of the premises in the proof-nets is only essential in the non-commutative logic [18].

5.2 ORIENTED PROOF-NETS

The construction of proof-nets from a given formula or sequent, is based on the use of negation and on the following equivalences: $(E \otimes F)^{\perp} = E^{\perp} \parr F^{\perp}$, $(E \parr F)^{\perp} = E^{\perp} \otimes F^{\perp}$, and $E \multimap F = E^{\perp} \parr F$. In **ILL** the linear implication cannot be defined from the $\parr$ connective because of the lack of negation. But in the spirit of using a classical proof to analyse the intuitionistic provability, it is interesting to keep the classical definition (and these equivalences) and to define the intuitionistic one from it. It is done by the introduction of a

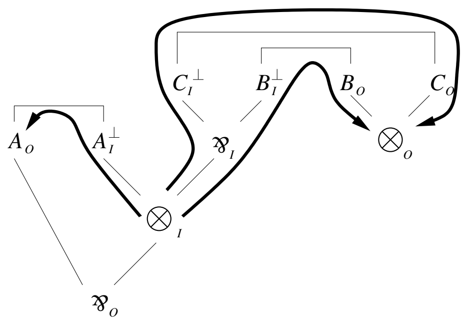


Figure 1.5 An oriented proof-net for $\vdash (A \parr (B \otimes C)) \multimap A,\ \ B \otimes C$.

corresponding orientation to annotate each node of the structure with either the label I (for Input) or the label O (for Output) [5]. It allows to consider the classical structure and to add through some labels necessary information to detect, during the construction, if a proof-net is intuitionistic or not.

Definition 21 *Let $\Gamma \vdash \Delta$ be a* **ILL** *sequent, the orientation of the formula tree is defined in the following way:*
i) the formulae of Γ (resp. Δ) are labelled with I (resp. O).
ii) a formula that is premise of a $\parr$ or $\otimes$ connective that does not come from an implication transformation, has the same label as the formula conclusion.
iii) a formula on the right-hand (resp. left-hand) side of a $\parr$ or $\otimes$ connective that comes from an implication transformation, has the same (resp. opposite) label as the formula conclusion.

Then a proof-net with such labels or annotations is called an oriented proof-net. The example of Figure 1.5 presents an oriented proof-net corresponding to the sequent $\vdash (A \parr (B \otimes C)) \multimap A,\ B \otimes C$. If we transform the $\multimap$ connective and annotate the formulae then we obtain the following sequent, that is classically equivalent to the initial one: $\vdash ((A_I^\perp \otimes (B_I^\perp \parr C_I^\perp)_I)_I \parr A_O)_O,\ (B_O \otimes C_O)_O$.

Property 22 *An oriented proof-net has at least one conclusion with label O.*

Proof By induction from the proof-nets definition. ■

Let us remark that there are proof-nets without conclusions labelled with I, for instance for provable sequents having no formula on the left-hand side.

Definition 23 *A* **ILL** proof-net *is an oriented proof-net without nodes of the following form:* $\begin{array}{c} O \quad O \\ \parr \\ O \end{array}$ *or* $\begin{array}{c} I \quad I \\ \otimes \\ I \end{array}$

Theorem 24 *A sequent is provable in* **ILL** *if and only if its associated oriented proof-net is a* **ILL** *proof-net.*

Proof It is a direct consequence of theorem 5. ■

5.3 FILL PROOF-NETS

The orientation defined in the previous section can be used to give a definition of a multiplicative **FILL** proof-net. The definition we propose is adapted from the one of Bellin for the multiplicative **FILL + MIX** [4].

Definition 25 *A* dependency path *is a path between a conclusion labelled with I and a conclusion labelled with O in a proof-net, that verifies that one always go upwards (resp. downwards) through I- (resp.O-) oriented nodes.*

The possible dependency subpaths, for each type of node, are given in Figure 1.6. Let us recall that a substructure S' of a proof-structure S is a proof-structure that is a subgraph. Then a *subnet* of a proof-net is a substructure that is a proof-net [5]. Moreover, let P_n be a proof-net and A be one of its nodes, the *empire of* A is the largest subnet of P_n with A as conclusion.

Definition 26 *A* **FILL** proof-net *is an oriented proof-net such that for any node* $\begin{array}{cc} A_I & B_O \\ \multicolumn{2}{c}{\parr_O} \end{array}$ *, there is no dependency path between A and a conclusion of the empire of A that is different from this occurrence of B.*

The Figure 1.7 presents two different proof-nets that correspond to the sequent proofs of section 4.2.2. The atomic formulae, that are A (resp. $A^{\perp}$) for the atoms labelled with O (resp. I), are not displayed. The first proof-net is not **FILL** because of the existence of a dependency path between the premise of the $\parr$ labelled with I and a node labelled with O which is not a premise of the $\parr$ connective.

5.4 FILL PROOF-NETS CONSTRUCTION

We have previously studied the problem of automated construction of proof-nets in different fragments of **CLL** like **MLL**, **MALL** or Non-Commutative **MLL** [17, 18, 19]. Some of the underlying principles could be applied for the case of multiplicative **FILL** fragment.

5.4.1 Principles for construction. We recall here the principles of our approach of proof-nets construction, taking the **MLL** fragment as illustration. More details can be found in [17, 19]. The construction is based on the inductive definition of proof-nets (see previous section) and instead of constructing

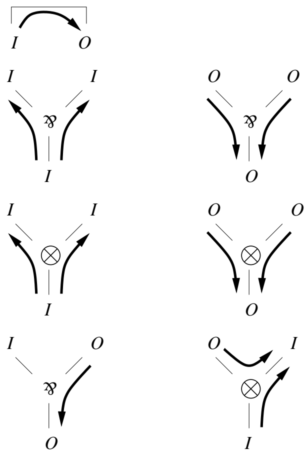


Figure 1.6 Dependency subpaths in **FILL** proof-nets

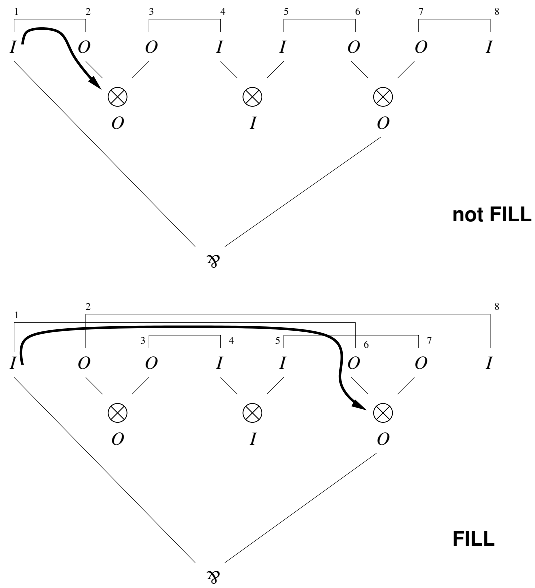


Figure 1.7 Proof-nets for $A_4 ⅋ A_5, A_8 \vdash A_1 \multimap (A_6 \otimes A_7), A_2 \otimes A_3$

a proof-structure and verifying with some criteria if it is a proof-net or not, subnets are built step by step to finally obtain a proof-net. Let us consider a **MLL** sequent of the form $\vdash \Delta$. At first, we decompose the formulae to obtain a decomposition tree of the sequent. The construction of a proof-net

corresponds to the construction of axiom-links that added to the decomposition tree forms a proof-structure that is in fact a proof-net. But how to find the suitable axiom-links (or connections) ? To find and justify the appropriate axiom-links, if they exist, we try to build, from some axiom-links (that are elementary proof-nets), some subnets that also are proof-nets, and to assemble them in the right way to finally obtain a proof-net, by using three procedures: one to select, in the decomposition tree, a leaf to treat; one to treat the leaves, of the decomposition tree, by constructing some axiom-links; one to extend the existing structures (by assembling subnets) with links. As we aim to reduce the number of disjoint subnets during this construction process, we choose the formula to treat in branches, some leaves of which have already been treated. Another strategy, based on permutability results and on the inductive definition of proof-nets consists in always dealing with the $\otimes$ connective before the $⅋$ connective, if possible, either to search the next leaf to treat or to assemble subnets. The search (-construction) of a proof-net follows these principles. To start we have to choose a branch and the successive nodes to finally obtain a leaf and we consider the $\otimes$-nodes as a priority. Having selected such a leaf, we construct an initial subnet by construction of an axiom-link with a dual formula. Each time a new subnet is created, we want to assemble some of the subnets by extension with links in the following way:
if the node under the current formula is a

- $\otimes$-node, both premises of which belong to disjoint subnets, then mix them into one new net with an extension with this node.
- $⅋$-node, both premises of which belong to the same subnet, then add the node to this subnet.
- $⅋$-node, both premises of which belong to disjoint subnets, then keep this node in a waiting position.
- $\otimes$-node, only one of the premises being treated, then treat the other premise as a priority.

We go on by choosing a new formula to deal with, treating at first the $\otimes$ connectives and then the $⅋$ connectives and trying to assemble subnets with the $⅋$ connectives being in the waiting position. At the end, if the proof-net exists then all formulae are treated and the resulting structure (or net) is the final proof-net. More details that are not necessary here can be found in [17, 18, 19]. Such an algorithm can be used to define a connection-based proof-search method [15].

5.4.2 FILL vs CLL proof-nets. To take into account the criteria on paths in **FILL** proof-nets, we add a piece of information during the creation of each new node of the **CLL** net to memorize all the dependency chains between a conclusion with label I and a conclusion with label O of the current subnet. For that, we have to follow the construction rules given in Figure 1.6. It leads

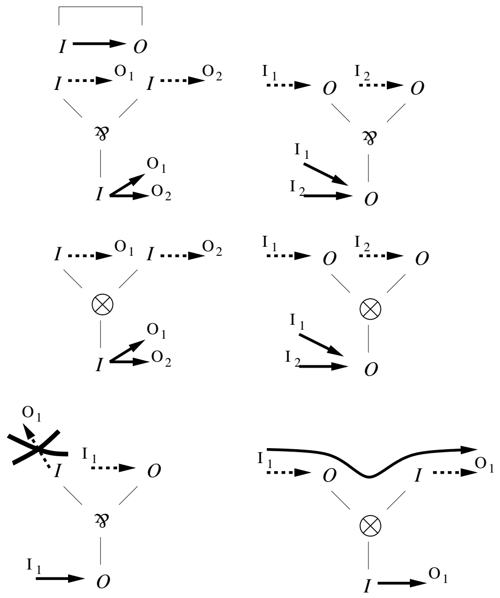


Figure 1.8 Update of dependency paths for **FILL** proof-nets.

to the following algorithm, the essential points of which being summarized in Figure 1.8, where the dashed arrows are replaced by the plain arrows. Thus the intuitionistic provability is considered as a perturbation of classical proof-net construction.

During the creation of axiom-links: there exists a dependency path between the formula with label I and the one with label O.

During the creation of $\begin{array}{c} I \quad I \\ \parr \\ I \end{array}$ **or** $\begin{array}{c} I \quad I \\ \otimes \\ I \end{array}$ **links:** the dependency paths between one of the premises and a conclusion labelled with O are all replaced by dependency paths between the new conclusion labelled with I and the same conclusion labelled with O.

During the creation of $\begin{array}{c} O \quad O \\ \parr \\ O \end{array}$ **or** $\begin{array}{c} O \quad O \\ \otimes \\ O \end{array}$ **links:** the dependency paths between a conclusion labelled with I and one of the premises are all replaced by dependency paths between the same conclusion labelled with I and the new conclusion labelled with O.

During the creation of $\begin{array}{cc} I & O \\ \multicolumn{2}{c}{\parr} \\ \multicolumn{2}{c}{O} \end{array}$ **link:** if there exists a dependency path between the premise labelled with I and a conclusion O, different from the premise labelled with O, the proof-net is not **FILL**. Else, the dependency paths between a conclusion labelled with I and the premise labelled with O are all replaced by dependency paths between the same conclusion labelled with I and the new conclusion labelled with O; moreover the dependency paths, the origin of which being labelled with I, are suppressed.

During the creation of $\begin{array}{cc} I & O \\ \multicolumn{2}{c}{\otimes} \\ \multicolumn{2}{c}{I} \end{array}$ **:** the dependency paths between a conclusion labelled with I and a premise labelled with O are all replaced by the paths of same origin but extended with the paths starting from the premise labelled with I; the dependency paths starting of the premise labelled with I are replaced by paths starting from the new conclusion labelled with I.

During the construction of classical **MLL** proof-nets, we memorize the information about dependency paths between labelled occurrences of formulae and we then dynamically verify the condition of definition 26 characterizing a multiplicative **FILL** proof-net. The algorithm always build such a proof-net if there exists.

5.4.3 An example. Let us come back to the examples of Figure 1.7. The algorithm builds the decomposition tree and then develops the proof-net from the leaves (or axiom-links), in a top-down approach. It tries to create an axiom-link between a node labelled with I (that corresponds to $A^{\perp}$) and a node labelled with O (that corresponds to A). Then it memorizes a path between these formulae respectively labelled with I and O. Let us assume that there is an axiom-link between the formulae 1 and 2 (see the first example). We go on with a $\otimes$ connective linked to one of these treated formulae and decide to consider the other premise. We then create the axiom-link between formulae 3 and 4 and a path from 4 to 3. Moreover, we can construct the $\otimes$-link and the two paths memorized are modified to point out on this $\otimes$-link. Therefore, the algorithm constructs the axiom-links and the $\otimes$-links and concludes with the construction of the $\parr$-link. In this first example, the situation at the moment of the creation of $\parr$-link is presented in Figure 1.9. In fact, it is impossible to create this link because of the existence of the bold arrow and this proof-net is not **FILL**. Then by backtracking there are other attempts of axiom-links construction leading to a proposal that allows to construct the $\parr$-link and then

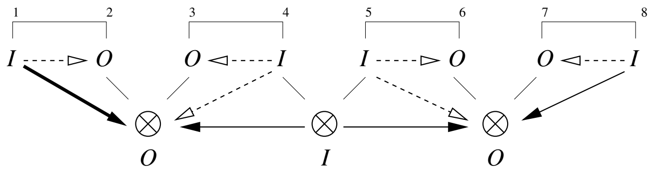


Figure 1.9 Dependency paths before the creation of the ⅋-link in Figure 1.7.

to a **FILL** proof-net (see the second example). Let us mention that, in general, the number of possibilities for the axiom-links is limited.

5.4.4 Main results. To prove the correction of the algorithm, it is necessary to consider two main properties.

Property 27 *The algorithm memorizes all the dependency paths between a conclusion labelled with I and a conclusion labelled with O, and only these ones.*

Proof The property is verified, case by case, for each type of node. ■

Let us mention now a lemma about the existence of dependency path.

Lemma 28 *Let P_n be a proof-net and A a node of P_n labelled with I, there exists a dependency path between A and a conclusion of P_n, labelled with O.*

Proof The property 22 indicates that there always exists a conclusion labelled with O. This lemma is proved in a similar way, by induction on the proof-net definition. The axiom-link case is trivial. For the ⅋- and $\otimes$-links we can distinguish the case where the node I is the new one that has just been introduced. The significative case is the one of $\otimes$, when the induction hypothesis gives a path between a node labelled with I and the unique premise of this $\otimes$-link that is labelled with O. A second use of the induction hypothesis (on the other net) allows the resulting path to be connected to the previous one. ■

This result leads to the following property:

Property 29 *Let us consider a node $A_I ⅋ B_O$ in a proof-net; then the existence of a dependency path between A_I and a conclusion labelled with O (different from B_O) is equivalent to the existence of a dependency path between A_I and a conclusion of the empire of A in the final proof-net, labelled with O.*

Proof The proof uses classical properties of proof-nets, that are not detailed here (see [21]). It follows a schema similar to the proof of property 27 and uses the lemma 28 to deal with the case of the $\otimes$-link, the premises of which do not have the same orientation. ■

Theorem 30 *Let* $\Gamma \vdash \Delta$ *be a* **CLL** *sequent, the algorithm builds a* **FILL** *proof-net for it iff it is provable in* **FILL**.

Proof It is a consequence of the algorithm properties and also of the previous properties and lemmas. ■

5.5 SOME REMARKS

The proof-nets construction algorithm for the multiplicative **FILL** has the property to simultaneously construct a sequent proof in a top-down (from axioms to final sequent) approach [17, 18]. Then we obtain an algorithm to directly construct a **FILL** proof if there exists and not only some criteria to verify if a **CLL** proof is also a **FILL** proof. As the previous algorithm finds a **FILL** proof if there exists, the use of proof-nets allows to solve the completeness problem w.r.t. intuitionistic provability we have mentioned in previous sections. This significant point illustrates, in the setting of linear logic, the interest of proof-net as an alternative tool for efficient automated deduction. We also observe the importance of the definition of labels for such a characterization.

It appears from the works of Asperti on logical analysis of distributed systems that **MLL** plus the **MIX** rule might be more suitable for the representation of concurrent logical computations [3]. This rule

$$\frac{\vdash \Gamma \quad \vdash \Delta}{\vdash \Gamma, \Delta}\ Mix$$

added to **MLL** has been also studied from the proof-net point of view by showing that the verification of correctness for proof-nets in this logic is equivalent to a successful termination of a concurrent game in the style of Petri nets [3]. Refinements and relationships with term assignments are proposed in [4] where a system of proof-nets for **MLL** (without constants) + **MIX** is given. From such a study of (multiplicative) classical linear logic, [4] has also proposed a proof-nets system for (multiplicative) **FILL** + **MIX**(without constants). Our results about intuitionistic provability can be extended and adapted to study the intuitionistic provability in **MLL** + **MIX**. It means that we can analyse it from the construction of a classical **MLL** + **MIX** proof-net with labels to detect the existence of a **FILL** + **MIX** proof-net. Such works could be significant for efficient proof-search in the context of analysis and proofs in concurrent and distributed systems.

6. CONCLUSION

In this paper, we have defined labelled sequent calculi to analyse and characterize the intuitionistic provability from proof-search in classical logic. Our first approach, in **CL** and **CLL**, is based on characterizations of the axiom sequents of a given labelled sequent proof. It does not use specific terms as classical or linear λ-terms to represent proofs (coming from natural deduction systems) as in [30]. We could argue that such sequent calculi seem more suitable from both proof-theoretic and proof-search points of view. In case of **CLL**, a recent study has defined a classical linear λ-calculus [8] and similar calculi could be potentially used to encode **CLL** proofs and to characterize the intuitionistic proofs, but it could be hard to handle mainly if provability is the main interest. Anyway, we cannot have a completeness result, i.e. the intuitionistic provability cannot be claimed from any given classical proof. Our labelled proof systems can be seen as new formulations, based-on labels, on the **FILL** system and could be independently interesting from the proof-theoretic point of view. We have proposed an alternative to this characterization in **CLL** that is based on the proof-net notion, that can be seen as a counterpart of natural deduction in linear logic. For that, we give a new labelled proof system and an algorithm that allow to build an intuitionistic proof-net (and a sequent proof) from a classical proof-net search, for which we have well-defined principles. With such a structure we obtain the completeness result missing with the other systems. This point illustrates the interest of the proof-net notion as an efficient tool for proof analysis and proof-search in linear logic. Because of the relationships between proof-nets and some term assignments systems [1] and also because of possible choices between intuitionistic or classical logical fragments to consider some programming paradigms, like proofs-as-programs or proofs-as-processes, our results and their possible extensions, for instance with the **MIX** rule or with the additive connectives, could be very fruitful. The result, in terms of proof-nets, is particular to linear logic but could encourage us to have similar structures or some graph-theoretic characterizations of provability in other substructural logics and therefore to derive new methods to analyse and search proofs. Significant points about the connected algorithmic and implementation problems will be studied in further work.

References

[1] S. Abramsky. Computational interpretations of Linear Logic. *Theoretical Computer Science*, 111(1-2):3–58, 1993.

[2] V. Alexiev. Applications of Linear Logic to Computation: An Overview. *Bulletin of the IGPL*, 2(1):77–107, 1994.

[3] A. Asperti. Causal dependencies in multiplicative linear logic with MIX. *Math. Struct. in Comp. Science*, 5:351–380, 1995.

[4] G. Bellin. Subnets of proof-nets in multiplicative linear logic with MIX. *Math. Struct. in Comp. Science*, 7:663–699, 1997.

[5] G. Bellin and J. van de Wiele. Subnets of proof-nets in MLL^-. In J.Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*, pages 249–260. Cambridge University Press, 1995.

[6] N. Benton, G. Bierman, V. de Paiva, and M. Hyland. A term calculus for intuitionistic linear logic. In *Int. Conference on Typed Lambda Calculi and Applications, LNCS 664*, pages 75–90, Utrecht, The Netherlands, March 1993.

[7] G.M. Bierman. A note on Full Intuitionistic Linear Logic. *Annals of Pure and Applied Logic*, 79:281–287, 1996.

[8] G.M. Bierman. Towards a classical linear λ-calculus (preliminary report). *Electronic Notes in Theoretical Computer Science*, 3, 1996.

[9] E. Boudinet and D. Galmiche. Proofs, concurrent objects and computations in a FILL framework. In *Workshop on Object-based Parallel and Distributed Computation, OBPDC'95, LNCS 1107*, pages 148–167, Tokyo, Japan, 1996.

[10] T. Braüner and V. De Paiva. Cut elimination for Full Intuitionnistic Linear Logic. Unpublished draft, 1996.

[11] M. D'Agostino and D.M. Gabbay. A Generalization of Analytic Deduction via Labelled Deductive Systems. Part I: Basic substructural logics. *Journal of Automated Reasoning*, 13:243–281, 1994.

[12] V. Danos and L. Regnier. The structures of multiplicatives. *Arch. Math. Logic*, 28:181–203, 1989.

[13] R. Dyckhoff. Contraction-free sequent calculi for intuitionistic logic. *Journal of Symbolic Logic*, 57:795–807, 1992.

[14] U. Engberg and G. Winskel. Completeness results for linear logic on Petri nets. *Annals of Pure and Applied Logic*, 86:101–135, 1997.

[15] D. Galmiche. Connection methods in Linear Logic and Proof nets Construction. *Theoretical Computer Science*, 1999. Accepted for publication.

[16] D. Galmiche and D. Larchey-Wendling. Provability in intuitionistic linear logic from a new interpretation on Petri nets - extended abstract. *Electronic Notes in Theoretical Computer Science*, 17, 1998.

[17] D. Galmiche and B. Martin. Proof search and proof nets construction in linear logic. In *4th Workshop on Logic, Language, Information and Computation, Wollic'97*, Fortaleza, Brasil, August 1997. Logic Journal of IGPL, vol. 5-6, pp 883-885.

[18] D. Galmiche and B. Martin. Proof nets construction and automated deduction in non-commutative linear logic - extended abstract. *Electronic Notes in Theoretical Computer Science*, 17, 1998.

[19] D. Galmiche and G. Perrier. A procedure for automatic proof nets construction. In *International Conference on Logic Programming and Automated Reasoning, LNAI 624*, pages 42–53, St. Petersburg, Russia, July 1992.

[20] J.Y. Girard. Linear logic and parallelism. In *Mathematical Models for the Semantics of Parallelism, LNCS 280*, pages 166–182, Roma, Italia, October 1986.

[21] J.Y. Girard. Linear logic. *Theoretical Computer Science*, 50(1):1–102, 1987.

[22] J.Y. Girard. Linear logic: its syntax and semantics. In J.Y. Girard, Y. Lafont, and L. Regnier, editors, *Advances in Linear Logic*, pages 1–42. Cambridge University Press, 1995.

[23] J.Y. Girard. Proof nets: the parallel syntax for proof theory. In Ursini and Agliano, editors, *Logic and Algebra*, New York, 1995. M. Dekker.

[24] J.Y. Girard and Y. Lafont. Linear logic and lazy computation. In *TAPSOFT 87, LNCS 250*, pages 52–66, Pisa, Italia, March 1987.

[25] J. Hodas and D. Miller. Logic programming in a fragment of intuitionistic linear logic. *Journal of Information and Computation*, 110:327–365, 1994.

[26] M. Hyland and V. de Paiva. Full Intuitionistic Linear Logic (extended abstract). *Annals of Pure and Applied Logic*, 64:273–291, 1993.

[27] P. Lincoln and J. Mitchell. Operational aspects of linear lambda calculus. In *7th IEEE Symposium on Logic in Computer Science*, pages 235–246, Santa-Cruz, California, 1992.

[28] D. Miller. Forum: A multiple-conclusion specification logic. *Theoretical Computer Science*, 165(1):201–232, 1996.

[29] M. Parigot. $\lambda\mu$-calculus: an algorithmic interpretation of classical natural deduction. In *International Conference on Logic Programming and Automated Reasoning, LNAI 624*, pages 190–201, St. Petersburg, Russia, July 1992.

[30] E. Ritter, D. Pym, and L. Wallen. On the intuitionistic force of classical search. In *5th Int. Workshop TABLEAUX'96, LNAI 1071*, pages 294–313, Terrasini, Palermo, Italy, May 1996.

[31] H. Schellinx. Some syntactical observations on linear logic. *Journal of Logic and Computation*, 1(4):537–559, 1991.

[32] L.A. Wallen. *Automated Proof search in Non-Classical Logics*. MIT Press, 1990.

Appendix

1. A SEQUENT CALCULUS FOR CL

$$\frac{}{A \vdash A}\ ax \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ cut$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta}\ w_L \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, A}\ w_R \qquad \frac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta}\ c_L \qquad \frac{\Gamma \vdash \Delta, A, A}{\Gamma \vdash \Delta, A}\ c_R$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \rightarrow B \vdash \Delta}\ \rightarrow_L \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \rightarrow B, \Delta}\ \rightarrow_R$$

$$\frac{\Gamma, A \vdash \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \vee B \vdash \Delta}\ \vee_L \qquad \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash A \vee B, \Delta}\ \vee_{R1} \qquad \frac{\Gamma \vdash B, \Delta}{\Gamma \vdash A \vee B, \Delta}\ \vee_{R2}$$

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\ \wedge_L \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma \vdash B, \Delta}{\Gamma \vdash A \wedge B, \Delta}\ \wedge_R$$

$$\frac{\Gamma \vdash A, \Delta}{\Gamma, \neg A \vdash \Delta}\ \neg_L \qquad \frac{\Gamma, A \vdash \Delta}{\Gamma \vdash \neg A, \Delta}\ \neg_R$$

In the paper we use an equivalent presentation, without weakening rules (w_L and w_R) and where the axiom rule is replaced by $\frac{}{\Gamma, A \vdash A, \Delta}\ ax$.

2. A SEQUENT CALCULUS FOR IL

$$\frac{}{A \vdash A}\ ax \qquad \frac{\Gamma \vdash A \quad \Gamma', A \vdash B}{\Gamma, \Gamma' \vdash B}\ cut$$

$$\frac{\Gamma, A, A \vdash B}{\Gamma, A \vdash B}\ c \qquad \frac{\Gamma \vdash B}{\Gamma, A \vdash B}\ w \qquad \frac{}{\bot \vdash A}\ \bot_L$$

$$\frac{\Gamma \vdash A \quad \Gamma, B \vdash C}{\Gamma, A \rightarrow B \vdash C}\ \rightarrow_L \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B}\ \rightarrow_R$$

$$\frac{\Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma, A \vee B \vdash C}\ \vee_L \qquad \frac{\Gamma \vdash A}{\Gamma \vdash A \vee B}\ \vee_{R1} \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \vee B}\ \vee_{R2}$$

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

3. ANOTHER SEQUENT CALCULUS FOR IL

$$\frac{}{A \vdash A}\ ax \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ cut$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta}\ w_L \qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, A}\ w_R \qquad \frac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta}\ c_L \qquad \frac{\Gamma \vdash \Delta, A, A}{\Gamma \vdash \Delta, A}\ c_R$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \rightarrow B \vdash \Delta}\ \rightarrow_L \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \rightarrow B, \Delta}\ \rightarrow_R$$

$$\frac{\Gamma, A \vdash \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \vee B \vdash \Delta}\ \vee_L \qquad \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash A \vee B, \Delta}\ \vee_R \qquad \frac{\Gamma \vdash B, \Delta}{\Gamma \vdash A \vee B, \Delta}\ \vee_R$$

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\ \wedge_L \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma \vdash B, \Delta}{\Gamma \vdash A \wedge B, \Delta}\ \wedge_R$$

$$\frac{\Gamma \vdash A, \Delta}{\Gamma, \neg A \vdash \Delta}\ \neg_L \qquad \frac{\Gamma, A \vdash}{\Gamma \vdash \neg A, \Delta}\ \neg_R$$

4. A SEQUENT CALCULUS FOR CLL

$$\frac{}{A \vdash A}\ ax \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\ cut$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, \mathbf{1} \vdash \Delta}\ \mathbf{1}_L \quad \frac{}{\vdash \mathbf{1}}\ \mathbf{1}_R \quad \frac{}{\bot \vdash}\ \bot_L \quad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \bot, \Delta}\ \bot_R \quad \frac{}{\Gamma, \mathbf{0} \vdash \Delta}\ \mathbf{0}_L \quad \frac{}{\Gamma \vdash \top, \Delta}\ \top_R$$

$$\frac{\Gamma \vdash A, \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \multimap B, \vdash \Delta, \Delta'}\ \multimap_L \qquad \frac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \multimap B, \Delta}\ \multimap_R$$

$$\frac{\Gamma, A \vdash \Delta \quad \Gamma', B \vdash \Delta'}{\Gamma, \Gamma', A \parr B \vdash \Delta, \Delta'}\ \parr_L \qquad \frac{\Gamma \vdash A, B, \Delta}{\Gamma \vdash A \parr B, \Delta}\ \parr_R$$

$$\frac{\Gamma, A, B \vdash \Delta}{\Gamma, A \otimes B \vdash \Delta}\ \otimes_L \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma' \vdash B, \Delta'}{\Gamma, \Gamma' \vdash A \otimes B, \Delta, \Delta'}\ \otimes_R$$

$$\frac{\Gamma, A \vdash \Delta}{\Gamma, A \& B \vdash \Delta}\ \&_{L1} \qquad \frac{\Gamma, B \vdash \Delta}{\Gamma, A \& B \vdash \Delta}\ \&_{L2} \qquad \frac{\Gamma \vdash A, \Delta \quad \Gamma \vdash B, \Delta}{\Gamma \vdash A \& B, \Delta}\ \&_R$$

$$\frac{\Gamma, A \vdash \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \oplus B \vdash \Delta}\ \oplus_L \qquad \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash A \oplus B, \Delta}\ \oplus_{R1} \qquad \frac{\Gamma \vdash B, \Delta}{\Gamma \vdash A \oplus B, \Delta}\ \oplus_{R2}$$

$$\frac{\Gamma \vdash A, \Delta}{\Gamma, A^{\perp} \vdash \Delta}\ neg_L \qquad \frac{\Gamma, A \vdash \Delta}{\Gamma \vdash A^{\perp}, \Delta}\ neg_R$$

$$\frac{\Gamma \vdash \Delta}{\Gamma, !A \vdash \Delta}\ \mathrm{w}_L \quad \frac{\Gamma \vdash \Delta}{\Gamma \vdash ?A, \Delta}\ \mathrm{w}_R \quad \frac{\Gamma, !A, !A \vdash \Delta}{\Gamma, !A \vdash \Delta}\ \mathrm{c}_L \quad \frac{\Gamma \vdash ?A, ?A, \Delta}{\Gamma \vdash ?A, \Delta}\ \mathrm{c}_R$$

$$\frac{\Gamma, A \vdash \Delta}{\Gamma, !A \vdash \Delta}\ \mathrm{dir}_L \quad \frac{\Gamma \vdash A, \Delta}{\Gamma \vdash ?A, \Delta}\ \mathrm{dir}_R \quad \frac{!\Gamma, A \vdash ?\Delta}{!\Gamma, ?A \vdash ?\Delta}\ \mathrm{pro}_L \quad \frac{!\Gamma \vdash A, ?\Delta}{!\Gamma \vdash !A, ?\Delta}\ \mathrm{pro}_R$$

5. A SEQUENT CALCULUS FOR ILL

$$\frac{}{A \vdash A}\ ax \qquad \frac{\Gamma \vdash A \quad \Gamma', A \vdash B}{\Gamma, \Gamma' \vdash B}\ cut$$

$$\frac{\Gamma \vdash A}{\Gamma, \mathbf{1} \vdash A}\ \mathbf{1}_L \quad \frac{}{\vdash \mathbf{1}}\ \mathbf{1}_R \quad \frac{}{\Gamma, \mathbf{0} \vdash A}\ \mathbf{0}_L \quad \frac{}{\Gamma \vdash \top}\ \top_R$$

$$\frac{\Gamma \vdash A \quad \Gamma', B \vdash C}{\Gamma, \Gamma', A \multimap B, \vdash C}\ \multimap_L \qquad \frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\ \multimap_R$$

$$\frac{\Gamma, A \vdash C}{\Gamma, A \& B \vdash C}\ \&_{L1} \qquad \frac{\Gamma, B \vdash C}{\Gamma, A \& B \vdash C}\ \&_{L2} \qquad \frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \& B}\ \&_R$$

$$\frac{\Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma, A \oplus B \vdash C}\ \oplus_L \qquad \frac{\Gamma \vdash A}{\Gamma \vdash A \oplus B}\ \oplus_{R1} \qquad \frac{\Gamma \vdash B}{\Gamma \vdash A \oplus B}\ \oplus_{R2}$$

$$\frac{\Gamma, A, B \vdash C}{\Gamma, A \otimes B \vdash C}\ \otimes_L \qquad \frac{\Gamma \vdash A \quad \Gamma' \vdash B}{\Gamma, \Gamma' \vdash A \otimes B}\ \otimes_R$$

$$\frac{\Gamma \vdash B}{\Gamma, !A \vdash B}\ \mathrm{w}_L \quad \frac{\Gamma, !A, !A \vdash B}{\Gamma, !A \vdash B}\ \mathrm{c}_L \quad \frac{\Gamma, A \vdash B}{\Gamma, !A \vdash B}\ \mathrm{dir} \quad \frac{!\Gamma \vdash A}{!\Gamma \vdash !A}\ \mathrm{pro}$$