On Structuring Proof Search for First Order Linear Logic

Paola Bruscoli and Alessio Guglielmi
Technische Universität Dresden
Hans-Grundig-Str. 25 - 01062 Dresden - Germany
Paola.Bruscoli@Inf.TU-Dresden.DE, Alessio.Guglielmi@Inf.TU-Dresden.DE

Abstract *We start from the* Forum *presentation of first order linear logic to design an equivalent system for which proof search is highly structured. We restrict formulae to a language of clauses and goals, without losing expressivity, in such a way that formulae have the same structure of* Forum *sequents. This means having a very big generalised connective that suffices for all of linear logic. We can then design a system with only two big rules, a left one and a right one. The behaviour of such system in proof search is operationally interesting and makes it suitable for further semantic investigations. We test the mutual harmony of the new rules by showing a cut elimination theorem.*

1 Introduction

Forum [9] is a presentation of linear logic which only produces uniform proofs. This guarantees that a sensible operational interpretation of proof search is possible. Surprisingly, Forum is complete for linear logic; this contrasts with the situation in classical logic, where a complete presentation that only produces uniform proofs is not possible. Given the linguistic flexibility of linear logic, i.e., its ability of interpreting a broad range of computational situations, Forum represents a major step forward towards practical applications.

This research is motivated by the search for adequate operational models of Forum, especially behavioural models like labelled event structures, which describe causal relations. These are particularly important for the domains of applications of Forum, namely the modelling of computations in concurrency and planning.

A proof in Forum mainly consists of small, deterministic steps, corresponding to applying each of several inference rules. This determinism is, of course, not a surprise, since Forum has been designed precisely for the purpose of reducing and isolating non-determinism. One soon realises that, in order to obtain sensible models, one must abstract away from situations in which several steps occur one after another and no choice at all is actually possible: when no choice is available, causality becomes trivial and uninteresting. It is of course possible to make this abstraction after a proof is produced, but it is desirable to design a system that makes the abstraction since the beginning. In other words, one wonders whether it is possible to carry Forum one step ahead: making further restrictions that, without losing expressivity, provide the right level of abstraction directly in the proof theory.

To get Forum, Miller imposed certain restrictions on the sequents, inference rules, and possible connectives of linear logic, but he left formula building free. In this paper we restrict the class of allowed formulae, along lines already imagined by Miller in [9], and we design correspondingly a system called G-Forum. By doing this, we have that formulae drive the construction of proofs in a very structured way, which allows us to individuate big chunks of derivations that essentially behave in a deterministic way: these will be the building blocks of our desired behavioural semantics. The restriction of formulae makes them isomorphic to the sequents in Forum. Of course, in order to claim that this is "good" proof theory, one has to

test the internal harmony of G-Forum. We do it the classical way: we prove a cut elimination theorem for the new system, and we show that it is not necessary to resort to Forum in the intermediate steps of the cut elimination procedure: G-Forum is enough, meaning that the new granularity of rules genuinely corresponds to what we can consider a generalised connective.

We leave the development of the behavioural semantics to a future paper (the reader can consult [6] for some preliminary results); in this paper we limit ourselves to defining G-Forum and show cut elimination for it. In sect. 2 we give a quick account of Forum, then we develop G-Forum in sect. 3.

2 First Order Forum

We deal with first order formal systems, and the following conventions apply.

2.1 Definition *First order variables* are denoted by x, y and z; *terms* are denoted by t, *atoms* by a, b, c, $\ldots$, $a(t_1, \ldots, t_h)$, $b(\ldots)$, $c(\ldots)$, $\ldots$. *Sequences* are denoted in vector notation, as in $\forall\vec{x}.a(\vec{t})$. *Formulae* are denoted by F and other letters which will be introduced later on. Formulae are considered equal under α-conversion.

This work is founded on linear logic; we are mainly interested in its first order sequent calculus presentation. We refer to the literature for details, especially to [5].

2.2 Definition The formal system of full *first order linear logic*, in its Gentzen's sequents presentation, and its language, are both denoted by FOLL. Formulae in FOLL are freely built from first order atoms and *constants* $\mathtt{1}$, $\bot$, $\top$, $\mathtt{0}$ by using *binary connectives* $\otimes$, $\parr$, $\&$, $\oplus$, $\multimap$, *modalities* $!$, $?$, *negation* $^\perp$ and the *quantifiers* $\forall$ and $\exists$. Constants $\mathtt{1}$, $\bot$ and connectives $\otimes$, $\parr$ and $\multimap$ are called the *multiplicatives*; $\top$, $\mathtt{0}$, $\&$ and $\oplus$ are called the *additives*. *Equivalence* is written $\equiv$. In linear logic $F \equiv F'$ iff $(F \multimap F') \,\&\, (F' \multimap F)$ is provable.

Intuitionistic implication $\Rightarrow$ admits the well-known decomposition $F \Rightarrow F' \equiv\ !F \multimap F'$; we can consider $\Rightarrow$ part of the calculus.

2.3 Definition The *binary connective* $\Rightarrow$ is introduced such that $F \Rightarrow F'$ is equivalent to $!F \multimap F'$.

2.4 Definition Multiplicative connectives, except $\multimap$, take precedence over additive ones; implications are the weakest connectives; modalities and quantifiers are stronger than binary connectives; negation takes precedence over everything. Implications associate to the right. Whenever possible, we omit parentheses.

For example, $!\forall x.a^\perp \multimap b \& c \parr d \Rightarrow e$ stands for $\big(!(\forall x.(a^\perp))\big) \multimap \big((b \& (c \parr d)) \Rightarrow e\big)$.

We briefly introduce the Forum formal system. The presentation corresponds to the one in [8], restricted to the first order case and with some minor modifications. An alternative and more detailed exposition can be found in [9].

2.5 Definition The language of *first order* Forum is the subset of FOLL freely built over atoms and the constants $\bot$ and $\top$ by use of the binary connectives $\parr$, $\&$, $\multimap$ and $\Rightarrow$ and of the quantifier $\forall$. We will say "Forum" instead of "first order Forum." Generic Forum formulae are denoted by A and B.

So, Forum presents fewer connectives than FOLL, by getting rid of some of

the redundant ones. It is not difficult to prove the following equivalences in FOLL:

$$\mathsf{1} \equiv \bot^{\bot}, \qquad \mathsf{0} \equiv \top^{\bot},$$
$$F \otimes F' \equiv (F^{\bot} \parr F'^{\bot})^{\bot}, \qquad F \oplus F' \equiv (F^{\bot} \,\&\, F'^{\bot})^{\bot},$$
$$!F \equiv (F \Rightarrow \bot)^{\bot}, \qquad ?F \equiv F^{\bot} \Rightarrow \bot,$$
$$\exists x.F \equiv (\forall x.F^{\bot})^{\bot},$$
$$F^{\bot} \equiv F \multimap \bot.$$

Then, one can equivalently write any FOLL formula into the Forum language.

2.6 Definition *Sequents* are expressions of the form

$$\begin{bmatrix}\Psi\\ \Gamma\end{bmatrix} A \vdash \begin{bmatrix}\Lambda\end{bmatrix} \qquad \text{or} \qquad \begin{bmatrix}\Psi\\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}\Xi\\ \Lambda\end{bmatrix},$$

where all formulae are Forum formulae and Ψ is a finite multiset of formulae (the *left classical context* or *classical program*); Γ is a finite multiset of formulae (the *left linear context* or *linear program*); A is a formula (the *left focused formula*); Ξ is a finite sequence of formulae (the *right linear context*); Λ is a finite multiset of atoms (the *atomic context*). Γ, Ξ and Λ are collectively referred to as the *linear context*. Ψ and Γ together are called the *program*. In the following Ψ, Γ, Ξ and Λ respectively stand for multisets, multisets and sequences of formulae and multisets of atoms. We write "Γ, A", or "A, Γ", instead of "$\Gamma \uplus \{A\}_+$" and "Γ, Γ'" instead of "$\Gamma \uplus \Gamma'$", where $\uplus$ is multiset union. Sequents are denoted by Σ. Sequents where no focused formula is present and Ξ is empty are called *state sequents*, are written

$$\begin{bmatrix}\Psi\\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}\Lambda\end{bmatrix}$$

and are denoted by S and R.

2.7 Definition An *inference rule* is an expression of the form $r\ \dfrac{\Sigma_1 \ \ldots \ \Sigma_h}{\Sigma}$, where $h \geqslant 0$, sequents $\Sigma_1, \ldots, \Sigma_h$ are the *premises* of the rule, Σ is its *conclusion* and r is the *name* of the rule. An inference rule with no premises is called an *axiom*.

2.8 Definition Let Forum be the first order proof system defined by inference rule schemes in fig. 1. Structural rules are: i (*initial*), d_{L} (*decide linear*), d_{C} (*decide classical*), a (*atom*). Logical rules, divided into left and right ones, are: $\bot_{\mathsf{L}}$, $\bot_{\mathsf{R}}$ (*bottom*); $\top_{\mathsf{R}}$ (*top*, there is no left rule); $\parr_{\mathsf{L}}$, $\parr_{\mathsf{R}}$ (*par*); $\&_{\mathsf{LL}}$, $\&_{\mathsf{LR}}$, $\&_{\mathsf{R}}$ (*with*); $\multimap_{\mathsf{L}}$, $\multimap_{\mathsf{R}}$ (*linear implication*); $\Rightarrow_{\mathsf{L}}$, $\Rightarrow_{\mathsf{R}}$ (*intuitionistic implication*); $\forall_{\mathsf{L}}$, $\forall_{\mathsf{R}}$ (*universal quantification*).

Consider proofs in Forum in a bottom-up reading. In absence of a left focused formula, the right linear context is acted upon by right rules until it is empty; at that point a formula becomes focused, in a d_{L} or d_{C} rule. Then left rules only are applicable, until new formulae reach the right linear context, through $\multimap_{\mathsf{L}}$ and $\Rightarrow_{\mathsf{L}}$ rules. Proofs in Forum are said to be *uniform* [8, 9, 10].

Our system's major differences with Forum as presented in [8] are: 1) our classical context is a multiset while in [8] it is a set; 2) our atomic context is a multiset while in [8] it is a sequence. These differences do not affect provability (and uniformity of proofs), as it can be proved trivially.

Representing derivations as directed trees whose nodes are sequents is typographically advantageous, especially in the cut elimination proof. The direction of the arrows corresponds to the tree growth during the search for a proof. It should be clear that there is no difference between this notation and the usual one.

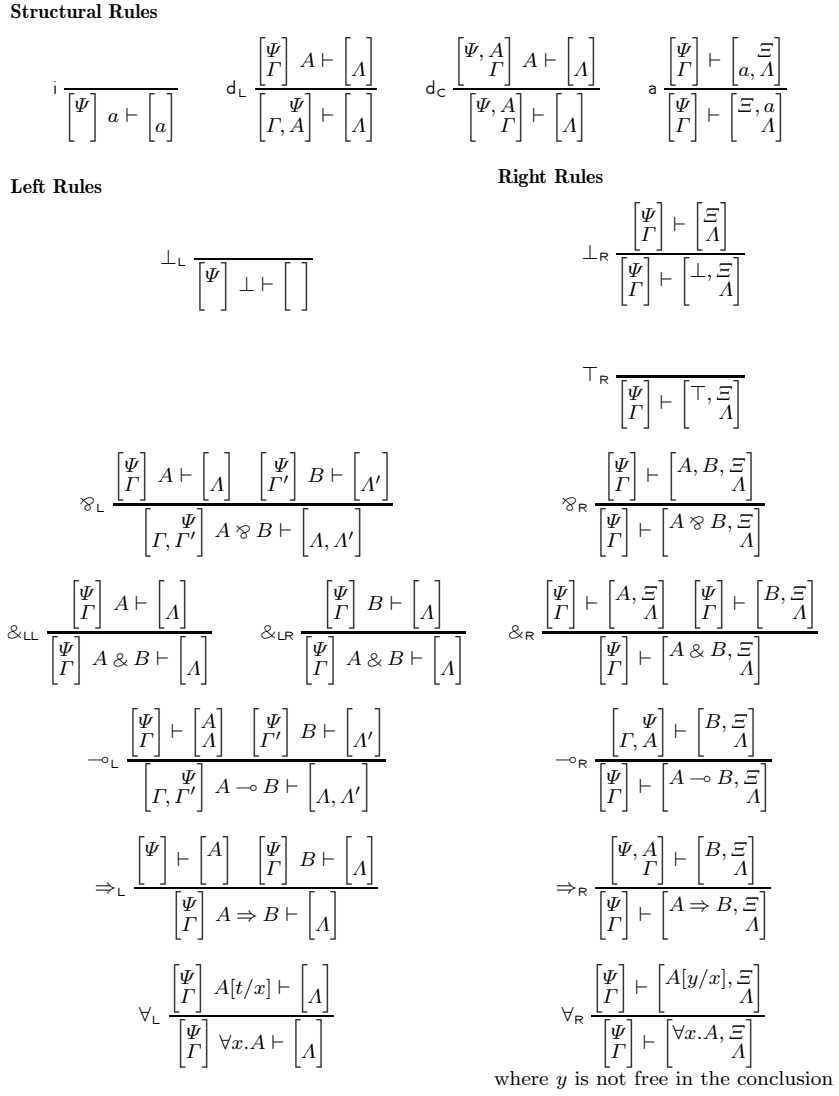
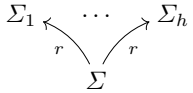


Fig. 1 *The first order* **Forum** *proof system*

2.9 Definition To every instance of an inference rule $r\,\dfrac{\Sigma_1 \quad \dots \quad \Sigma_h}{\Sigma}$, when $h > 0$ an *elementary derivation*



corresponds, *i.e.* a labeled directed tree whose root is labeled Σ, whose leaves are labeled Σ_1, …, Σ_h and whose arcs are labeled r; when $h = 0$ the corresponding

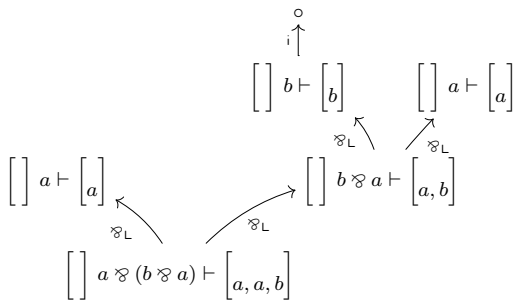


Fig. 2 *Example of derivation*

elementary derivation is

$$\begin{array}{c}\circ\\ r\uparrow\\ \Sigma\end{array},$$

where $\circ$ is a mark distinct from every sequent. *Derivations* are non-empty, finite directed trees whose root is labeled by a sequent and whose other nodes are labeled by sequents or $\circ$ marks and such that every maximal subtree of depth 1 is an elementary derivation. Derivations are denoted by Δ. Given a derivation Δ, we say that Δ' is a *subderivation* of Δ if Δ' is a derivation and a subtree of Δ. Δ' is a *principal subderivation* of Δ if it is a subderivation of Δ and the roots of Δ and Δ' coincide. Given Δ, its *premises* are the labels of the leaves of Δ other than $\circ$; its *conclusion* is the sequent labeling the root of Δ. The multiset of premises of Δ is indicated by $\widetilde{\Delta}$. A sequent Σ is a derivation (of depth 0) whose premise and conclusion is Σ. A derivation Δ such that its premises are $\Sigma_1, \ldots, \Sigma_h$ and its conclusion is Σ can be represented as

$$\begin{array}{c}\Sigma_1\ \cdots\ \Sigma_h\\ \overbrace{\quad\Delta\quad}\\ \Sigma\end{array}.$$

Sometimes the name of the derivation is not shown. If Δ is the derivation

$$\begin{array}{ccc}\overbrace{\quad\Delta_1\quad}^{\vec{\Sigma}_1} & & \overbrace{\quad\Delta_h\quad}^{\vec{\Sigma}_h}\\ \Sigma_1 & \cdots & \Sigma_h\\ & {\scriptstyle r}\nwarrow\quad\nearrow{\scriptstyle r} & \\ & \Sigma & \end{array},$$

where $h \geq 0$, we define its *depth* $\mathsf{d}(\Delta)$ as $\mathsf{max}\,\{\mathsf{d}(\Delta_1), \ldots, \mathsf{d}(\Delta_h)\} + 1$, where, for every sequent Σ, it holds $\mathsf{d}(\Sigma) = \mathsf{d}(\circ) = 0$. If Δ has no premises we say that Δ is a *proof*. Proofs are denoted by Π. We say that Π *proves* (or *is a proof of*) its conclusion. We say that a formula A is *provable* in Forum, or that Forum *proves* A, if a proof of $\left[\,\right] \vdash \left[A\right]$ exists.

The premises of the derivation in fig. 2 are $\left\{\left[\,\right]\ a \vdash \left[a\right], \left[\,\right]\ a \vdash \left[a\right]\right\}_+$ and its conclusion is $\left[\,\right]\ a \parr (b \parr a) \vdash \left[a, a, b\right]$. This derivation can be completed into a proof by applying two initial rules to its premises.

Arcs are not independent in the growth process of a derivation: all arcs propagating from a node correspond to the application of the same inference rule.

By looking at fig. 1 it is clear that if we make the classical context a set (as Miller does), derivability is not affected. In fact, the only impact is on the $\Rightarrow_{\mathsf{R}}$ rule, but things do not change, because the classical context is implicitly subject to weakening in all axioms.

2.10 Theorem *Every* Forum *formula is provable in* Forum *if and only if it is provable in* FOLL. (Miller [8, 9])

3 Derivations at a Higher Level of Abstraction

Consider a formula $\delta = G_1 \Rightarrow \cdots \Rightarrow G_{k''} \Rightarrow H_1 \multimap \cdots \multimap H_{k'} \multimap a_1 \parr \cdots \parr a_k$. In Forum, in a bottom-up construction of a derivation, from $\begin{bmatrix} \\ \end{bmatrix} \vdash \begin{bmatrix} \delta \end{bmatrix}$ we are always led to the state sequents $\begin{bmatrix} G_1, \ldots, G_{k''} \\ H_1, \ldots, H_{k'} \end{bmatrix} \vdash \begin{bmatrix} a_1, \ldots, a_k \end{bmatrix}$. Let us call *clauses* formulae like δ, where formulae G_i and H_j (*goals*) are of the form $\forall \vec{x}.(\delta_1 \mathbin{\&} \cdots \mathbin{\&} \delta_h)$, and where in the $\&$ conjunction only clauses are allowed.

In this section we derive a proof system equivalent to FOLL. The new proof system is in fact the old Forum proof system seen at a coarser abstraction level: rules are essentially macro derivations composed of many Forum rules, and the only formulae allowed are goals (and clauses).

3.1 Goals and Clauses

We define goals and clauses, which are Forum formulae of a constrained shape; then we show that their language is equivalent to Forum and then to FOLL.

3.1.1 Definition Goals and clauses are recursively defined this way:

1) A *goal* is a formula of the form

$$\forall \vec{x}.(\delta_1 \mathbin{\&} \cdots \mathbin{\&} \delta_h),$$

where $\vec{x}$ can be empty, $h \geqslant 0$ and every δ_i is a clause. When $h = 0$ a goal is $\forall \vec{x}.\top$.

2) A *clause* δ is a formula of the form

$$G_1 \Rightarrow \cdots \Rightarrow G_{k''} \Rightarrow H_1 \multimap \cdots \multimap H_{k'} \multimap a_1 \parr \cdots \parr a_k,$$

where $k, k', k'' \geqslant 0$, formulae G_i and H_i are goals and formulae a_i are atoms. Goals G_i are called the *classical premises* of δ, goals H_i are its *linear premises* and $a_1 \parr \cdots \parr a_k$ is the *head* of the clause. We define $\mathsf{hd}(\delta) = \{a_1, \ldots, a_k\}_+$, $\mathsf{lp}(\delta) = \{H_1, \ldots, H_{k'}\}_+$ and $\mathsf{cp}(\delta) = \{G_1, \ldots, G_{k''}\}_+$. When $k = 0$ the head is $\bot$. When $k' = 0$ and $k'' = 0$ clauses assume the following special forms, respectively:

$$G_1 \Rightarrow \cdots \Rightarrow G_{k''} \Rightarrow a_1 \parr \cdots \parr a_k \quad \text{and}$$
$$H_1 \multimap \cdots \multimap H_{k'} \multimap a_1 \parr \cdots \parr a_k.$$

Goals are denoted by G and H, clauses by δ and γ.

Clearly, a clause is also a goal.

3.1.2 Theorem *Every formula in* FOLL *is equivalent to a goal in* Forum.
Proof We show that, taken any formula in Forum, we can exhibit an equivalent goal. We use the following absorption equivalences:

1) $F \parr \bot \equiv F$;

2) $F \parr \top \equiv \top$;

3) $F \,\&\, \top \equiv F$.

We also use the following equivalences:

4) $F \parr (F' \,\&\, F'') \equiv (F \parr F') \,\&\, (F \parr F'')$;

5) $\forall x.F \parr F' \equiv \forall x.(F \parr F')$, whenever x is not free in F' and

6) $\forall x.F \,\&\, F' \equiv \forall x.(F \,\&\, F')$, whenever x is not free in F'.

Let A be a formula in Forum: the proof is by induction on its structure. The base cases being trivial, consider given B and B'; by the induction hypothesis we suppose we are also given two goals G and G' such that

$$B \equiv G = \forall \vec{x}.(\delta_1 \,\&\, \cdots \,\&\, \delta_h),$$
$$B' \equiv G' = \forall \vec{y}.(\delta'_1 \,\&\, \cdots \,\&\, \delta'_{h'}),$$

where $\vec{x}$ and $\vec{y}$ may be empty and h and h' may be 0. The following cases may occur.

- $A = B \parr B'$. By applications of equivalence 5 and renaming of bounded variables, if necessary, we get

$$A \equiv \forall \vec{z}.\big((\delta_1 \,\&\, \cdots \,\&\, \delta_h) \parr (\delta'_1 \,\&\, \cdots \,\&\, \delta'_h)\big).$$

  If $h = 0$ or $h' = 0$ we can conclude that $A \equiv \forall \vec{z}.\top$, by making use of equivalence 2. Otherwise, we may repeatedly apply equivalence 4 above, and we get:

$$A \equiv \forall \vec{z}.\Big(\big((\delta_1 \parr \delta'_1) \,\&\, \cdots \,\&\, (\delta_h \parr \delta'_1)\big) \,\&\, \cdots \,\&\, \big((\delta_1 \parr \delta'_{h'}) \,\&\, \cdots \,\&\, (\delta_h \parr \delta'_{h'})\big)\Big).$$

  For $1 \leqslant i \leqslant h$ and $1 \leqslant j \leqslant h'$, let

$$\delta_i = G^i_1 \Rightarrow \cdots \Rightarrow G^i_{h''_i} \Rightarrow H^i_1 \multimap \cdots \multimap H^i_{h'_i} \multimap a^i_1 \parr \cdots \parr a^i_{h_i},$$
$$\delta'_j = G'^j_1 \Rightarrow \cdots \Rightarrow G'^j_{k''_j} \Rightarrow H'^j_1 \multimap \cdots \multimap H'^j_{k'_j} \multimap a'^j_1 \parr \cdots \parr a'^j_{k_j}.$$

  Since $F \Rightarrow F' \equiv !F \multimap F'$ and $F \multimap F' \equiv F^\perp \parr F'$, commutativity of $\parr$ suffices to show that

$$\begin{aligned}\delta_i \parr \delta'_j \equiv G^i_1 \Rightarrow \cdots \Rightarrow G^i_{h''_i} \Rightarrow G'^j_1 \Rightarrow \cdots \Rightarrow G'^j_{k''_j} \Rightarrow \\ H^i_1 \multimap \cdots \multimap H^i_{h'_i} \multimap H'^j_1 \multimap \cdots \multimap H'^j_{k'_j} \multimap \\ a^i_1 \parr \cdots \parr a^i_{h_i} \parr a'^j_1 \parr \cdots \parr a'^j_{k_j}.\end{aligned}$$

  Special cases where $h_i = 0$ or $k_j = 0$ are handled by equivalence 1 above.

- $A = B \,\&\, B'$. By applications of equivalence 6 and renaming of bounded variables, if necessary, we get

$$A \equiv \forall \vec{z}.(\delta_1 \,\&\, \cdots \,\&\, \delta_h \,\&\, \delta'_1 \,\&\, \cdots \,\&\, \delta'_h).$$

  If $h = 0$ or $h' = 0$ use equivalence 3.

- $A = B \multimap B'$. By using equivalences 5 and 4, and by renaming bounded variables if necessary, we have:

$$\begin{aligned}A &\equiv G^\perp \parr \forall \vec{y}.(\delta'_1 \,\&\, \cdots \,\&\, \delta'_h) \\ &\equiv \forall \vec{z}.(G^\perp \parr (\delta'_1 \,\&\, \cdots \,\&\, \delta'_h)) \\ &\equiv \forall \vec{z}.((G^\perp \parr \delta'_1) \,\&\, \cdots \,\&\, (G^\perp \parr \delta'_h)).\end{aligned}$$

$$\begin{array}{c}
\begin{bmatrix}\Psi, G_1, \ldots, G_{k''} \\ \Gamma, H_1, \ldots, H_{k'}\end{bmatrix} \vdash \begin{bmatrix}\Xi \\ a_1, \ldots, a_k, \Lambda\end{bmatrix} \\
{\scriptstyle (\parr_{\mathsf{R}} \text{ or } \mathtt{a})^\star}\uparrow \\
\begin{bmatrix}\Psi, G_1, \ldots, G_{k''} \\ \Gamma, H_1, \ldots, H_{k'}\end{bmatrix} \vdash \begin{bmatrix}a_1 \parr \cdots \parr a_k, \Xi \\ \Lambda\end{bmatrix} \\
{\scriptstyle \multimap^\star_{\mathsf{R}}}\uparrow \\
\begin{bmatrix}\Psi, G_1, \ldots, G_{k''} \\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}H_1 \multimap \cdots \multimap H_{k'} \multimap a_1 \parr \cdots \parr a_k, \Xi \\ \Lambda\end{bmatrix} \\
{\scriptstyle \Rightarrow^\star_{\mathsf{R}}}\uparrow \\
\begin{bmatrix}\Psi \\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}G_1 \Rightarrow \cdots \Rightarrow G_{k''} \Rightarrow H_1 \multimap \cdots \multimap H_{k'} \multimap a_1 \parr \cdots \parr a_k, \Xi \\ \Lambda\end{bmatrix}
\end{array}$$

Fig. 3 *Clause reduction right inference rule* δ_{R}

By commutativity of $\parr$ it is easily seen that every $(G^\perp \parr \delta'_i)$ is a clause. If $B' \equiv \top$ then $A \equiv \top$.

- $A = B \Rightarrow B'$. The argument goes as in the previous case.
- $A = \forall x.B$. Trivial.

□

3.1.3 Corollary *Every formula in* FOLL *is equivalent to a clause.*

Proof Let F be a formula and $G \equiv F$, where G is obtained as in th. 3.1.2. Then, $(G \multimap \bot) \multimap \bot$ is a clause equivalent to G. □

From the proof of th. 3.1.2 an obvious algorithm can be derived to transform a Forum formula into a goal. Please note that if A is a Forum formula and G the equivalent goal found by the given procedure, then the set of logical constants appearing in G is not greater than that of A. In fact, in none of the cases considered new connectives are introduced. If G is not a clause, and a clause is required, then $(G \multimap \bot) \multimap \bot$ introduces $\bot$, not necessarily present in A. It could be possible to prove the result in a shorter way, but we would have to give up this separation property.

Not introducing new connectives in the translation to a goal has obvious benefits concerning modularity. In particular, this property takes care of some concerns of Miller in [8] about clauses (similar to ours) with degenerate head $\bot$. Clauses of that kind, when at the left of $\vdash$, are always available to rewritings, what is of course cause of explosion of the search space of proofs.

3.2 Deriving in the Right Context

3.2.1 Definition Let δ_{R} be the following *clause reduction right* inference rule, shown in fig. 3 in terms of Forum rules:

$$\delta_{\mathsf{R}} \frac{\begin{bmatrix}\Psi, \mathsf{cp}(\delta) \\ \Gamma, \mathsf{lp}(\delta)\end{bmatrix} \vdash \begin{bmatrix}\Xi \\ \mathsf{hd}(\delta), \Lambda\end{bmatrix}}{\begin{bmatrix}\Psi \\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}\delta, \Xi \\ \Lambda\end{bmatrix}}.$$

In the figure $k > 0$ and $k', k'' \geqslant 0$. Starred inference rule names mean repeated application of the rule, or no application at all; $(\parr_{\mathsf{R}}$ or $\mathtt{a})$ stands for "application of one of either $\parr_{\mathsf{R}}$ or $\mathtt{a}$." In the special case where $k = 0$ the upper sequence of $(\parr_{\mathsf{R}}$ or $\mathtt{a})$ rules is replaced by a single application of $\bot_{\mathsf{R}}$.

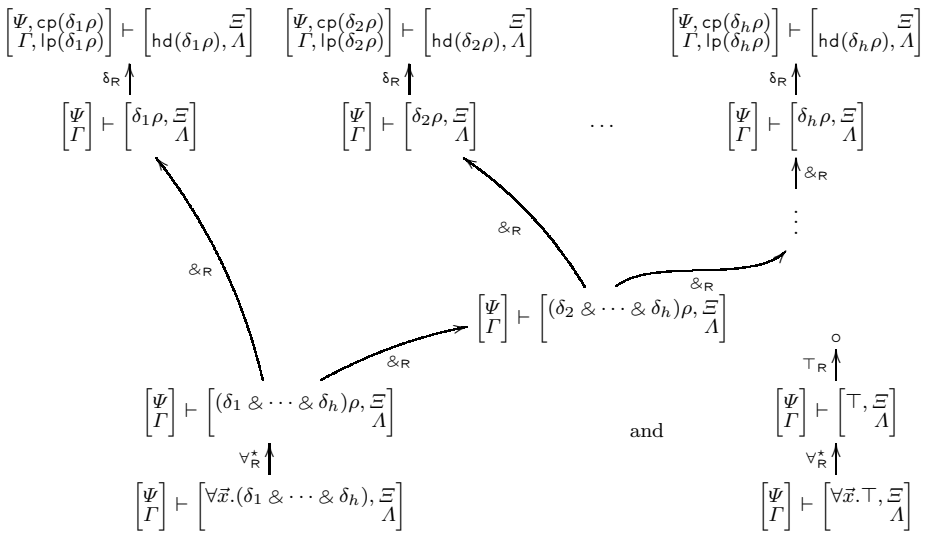


Fig. 4 *Goal reduction right inference rule* $\mathsf{G_R}$ *when* $h > 0$ *and* $h = 0$

3.2.2 Proposition *Every proof of* $\begin{bmatrix}\Psi\\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}\delta, \Xi\\ \Lambda\end{bmatrix}$ *has shape* $\dfrac{\begin{bmatrix}\Psi'\\ \Gamma'\end{bmatrix} \vdash \begin{bmatrix}\Xi\\ \Lambda'\end{bmatrix}}{\begin{bmatrix}\Psi\\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}\delta, \Xi\\ \Lambda\end{bmatrix}}\,\delta_{\mathsf{R}}$.

Proof By reasoning bottom-up, each application of an inference rule is compulsory. □

All Forum inference rules applied in δ_{R} are right ones. This is of course an aspect of the fact that Forum produces only uniform proofs (see [8, 9, 10]). Rules $\top_{\mathsf{R}}$, $\&_{\mathsf{R}}$ and $\forall_{\mathsf{R}}$ are still missing: they will appear in the reduction of goals.

We can build on δ_{R} an inference rule which reduces goals in the right linear context.

3.2.3 Definition Let $\mathsf{G_R}$ be the following *goal reduction right* inference rule, shown in fig. 4 in terms of δ_{R} and Forum rules:

$$\mathsf{G_R}\,\frac{\begin{bmatrix}\Psi, \mathsf{cp}(\delta_1\rho)\\ \Gamma, \mathsf{lp}(\delta_1\rho)\end{bmatrix} \vdash \begin{bmatrix}\Xi\\ \mathsf{hd}(\delta_1\rho), \Lambda\end{bmatrix} \quad \cdots \quad \begin{bmatrix}\Psi, \mathsf{cp}(\delta_h\rho)\\ \Gamma, \mathsf{lp}(\delta_h\rho)\end{bmatrix} \vdash \begin{bmatrix}\Xi\\ \mathsf{hd}(\delta_h\rho), \Lambda\end{bmatrix}}{\begin{bmatrix}\Psi\\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}\forall\vec{x}.(\delta_1 \,\&\, \cdots \,\&\, \delta_h), \Xi\\ \Lambda\end{bmatrix}},$$

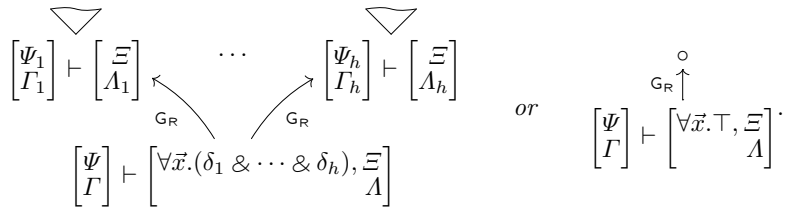
where $\vec{x}$ can be empty, ρ is an appropriate renaming substitution and $h \geqslant 0$. In the figure only one choice among the possible associations of & connectives has been considered, but every choice leads to the same multiset of premises.

This whole reduction phase is deterministic: in the end a goal is reduced to pieces with no choice about the possible outcome, except for the rather immaterial choice of eigenvariables in $\mathsf{G_R}$ rules. The Forum system has been designed to reduce choices to a minimum, in a bottom-up construction of a proof. Still, some "unnecessary" sequentialization exists: in the case above it resides in the binary treatment of associative connectives. We can consider the $\mathsf{G_R}$ rule at the abstraction level in

which all premises are reached at the same time in a parallel way, thus hiding the sequentialization at the Forum's level of abstraction. In other words we can consider every instance of the G_{R} rule a representative of an equivalence class of derivations, differing only on the associations of $\&$ connectives.

We can perform on the G_{R} rule the same kind of simple reasoning we did for δ_{R} in prop. 3.2.2.

3.2.4 Proposition *Every proof of* $\begin{bmatrix}\Psi\\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}G, \Xi\\ \Lambda\end{bmatrix}$ *has shape*



G_{R} defines the behaviour of goals when they appear at the right of $\vdash$. They generate as many threads in the computation as there are clauses in the conjunction. These threads are independent, and can be considered *parallel* computations. When G_{R} is applied to a $\forall\vec{x}.\top$, it just *terminates* a (thread of a) computation.

3.2.5 Definition A G-*state sequent* is a state sequent of the kind $\begin{bmatrix}\Psi\\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}\\ \Lambda\end{bmatrix}$, where all formulae in Ψ and Γ are goals.

By 3.1.2 and 3.2.4 we can always reduce provability of a Forum formula (therefore of a FOLL's one, by 2.10) to provability of some G-state sequents. Moreover, we can always reduce provability of a given formula to the provability of exactly one G-state sequent by employing the double negation equivalence $G \equiv (G \multimap \bot) \multimap \bot$: this last formula is a clause.

3.3 Deriving in the Left Context

G-State sequents embody the natural notion of state of our computations. A G-state sequent represents state at a certain level of abstraction; to proceed, clauses from its program must be applied to its atomic context. Left rules come into play: application of clauses is mainly accomplished by $\multimap_{\mathsf{L}}$ and $\Rightarrow_{\mathsf{L}}$ rules.

3.3.1 Definition Let h be the following *head matching* inference rule, where $k \geqslant 0$:

$$\mathsf{h}\ \frac{}{\begin{bmatrix}\Psi\\ \end{bmatrix}\ a_1 \parr \cdots \parr a_k \vdash \begin{bmatrix}\\ a_1, \ldots, a_k\end{bmatrix}}.$$

Fig. 5 shows how h corresponds to Forum inference rules. The same considerations made above about the associativity of $\&$ hold here for $\parr$.

3.3.2 Proposition *If the sequent* $\begin{bmatrix}\Psi\\ \Gamma\end{bmatrix}\ a_1 \parr \cdots \parr a_k \vdash \begin{bmatrix}\\ \Lambda\end{bmatrix}$ *is provable, then* Γ *is empty,* $\Lambda = \{a_1, \ldots, a_k\}_+$ *and the only proof is* $\mathsf{h}\ \dfrac{}{\begin{bmatrix}\Psi\\ \end{bmatrix}\ a_1 \parr \cdots \parr a_k \vdash \begin{bmatrix}\\ a_1, \ldots, a_k\end{bmatrix}}$.

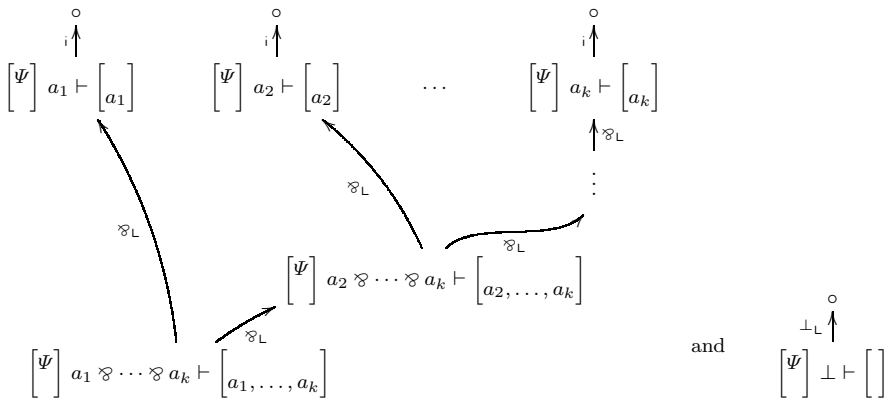


Fig. 5 *Head matching inference rule* h *when* $k > 0$ *and* $k = 0$

3.3.3 Definition Let δ_{L} be the following *clause reduction left* inference rule, shown in fig. 6 in terms of h and Forum rules:

$$\delta_{\mathsf{L}} \frac{\left[\Psi\right] \vdash \left[G_1\right] \quad \cdots \quad \left[\Psi\right] \vdash \left[G_{k''}\right] \quad \left[\begin{matrix}\Psi\\ \Gamma_1\end{matrix}\right] \vdash \left[\begin{matrix}H_1\\ \Lambda_1\end{matrix}\right] \quad \cdots \quad \left[\begin{matrix}\Psi\\ \Gamma_{k'}\end{matrix}\right] \vdash \left[\begin{matrix}H_{k'}\\ \Lambda_{k'}\end{matrix}\right]}{\left[\begin{matrix}\Psi\\ \Gamma\end{matrix}\right] \; \delta \vdash \left[\Lambda\right]},$$

where $\delta = G_1 \Rightarrow \cdots \Rightarrow G_{k''} \Rightarrow H_1 \multimap \cdots \multimap H_{k'} \multimap a_1 \otimes \cdots \otimes a_k$, and $k, k', k'' \geqslant 0$, and where $\Gamma_1 \uplus \cdots \uplus \Gamma_{k'} = \Gamma$ and $\Lambda_1 \uplus \cdots \uplus \Lambda_{k'} \uplus \{a_1, \ldots, a_k\}_+ = \Lambda$.

As an outcome of the reduction of the left focused clause, we have a multiset of premises which will be further reduced by as many G_{R} rules. They, in turn, will produce G-state sequents. The most degenerate instances of δ_{L} have no premises. Special cases where there are no classical or linear premises are easily inferable from the general scheme provided. Thanks to uniform provability, all non-determinism in searching for Forum proofs resides in left rules. Much of it can be concentrated into a decision rule, but one should notice that δ_{L} is also nondeterministic in the splitting of the linear contexts.

3.3.4 Definition Let d be the *decision* inference rule, defined in the following two (non-mutually exclusive) cases, and shown in Fig. 7 in terms of Forum rules:

$$\mathsf{d} \frac{\left[\begin{matrix}\Psi\\ \Gamma\end{matrix}\right] \; \delta_l\sigma \vdash \left[\Lambda\right]}{\left[\begin{matrix}\Psi\\ \Gamma, G\end{matrix}\right] \vdash \left[\Lambda\right]} \qquad \text{or} \qquad \mathsf{d} \frac{\left[\begin{matrix}\Psi, G\\ \Gamma\end{matrix}\right] \; \delta_l\sigma \vdash \left[\Lambda\right]}{\left[\begin{matrix}\Psi, G\\ \Gamma\end{matrix}\right] \vdash \left[\Lambda\right]},$$

where the conclusions are G-state sequents. In the first case Γ, G is the *selected context*, in the second it is Ψ, G. $G = \forall\vec{x}.(\delta_1 \,\&\, \ldots \,\&\, \delta_h)$, where $h > 0$ and $\vec{x}$ can be empty, is the *selected goal*; $\delta_l\sigma$ is the *selected clause*, $1 \leq l \leq h$, and σ is a substitution whose domain is $\vec{x}$.

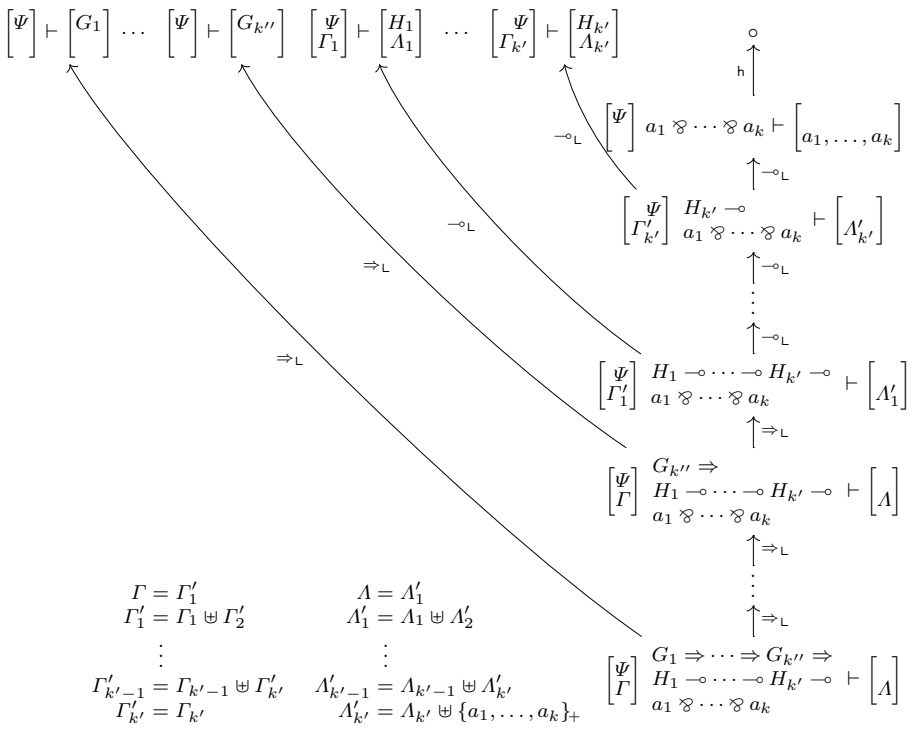


Fig. 6 *Clause reduction left inference rule* $\delta_{\multimap}$

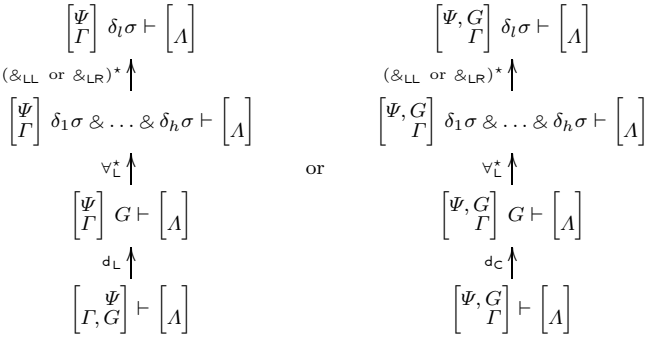


Fig. 7 *Decision inference rule* d *in its two possibilities*

3.3.5 Proposition *All proofs of* $\begin{bmatrix}\Psi\\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}\Lambda\end{bmatrix}$ *have shape* $\begin{array}{c}\begin{bmatrix}\Psi\\ \Gamma'\end{bmatrix}\ \delta \vdash \begin{bmatrix}\Lambda\end{bmatrix}\\ \mathsf{d}\uparrow\\ \begin{bmatrix}\Psi\\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}\Lambda\end{bmatrix}\end{array}$, *for some* Γ', *and the inference rule above* d *is* $\delta_{\multimap}$.

We can build on d and $\delta_{\multimap}$ an inference rule which reduces goals in the program.

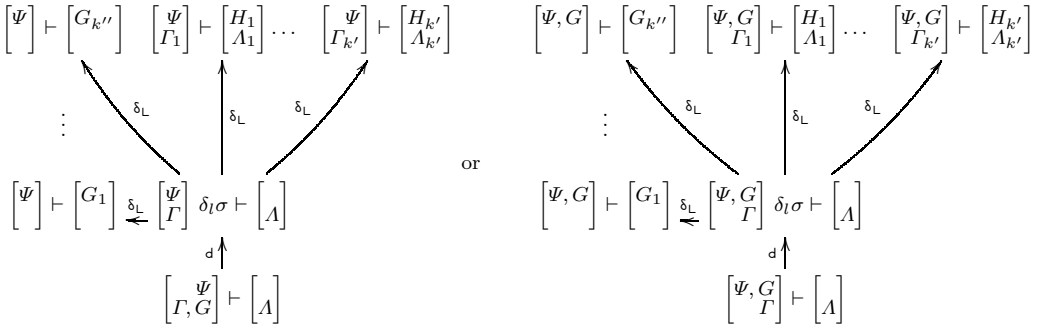


Fig. 8 *Goal reduction left inference rule* G_{L} *in its two possibilities*

3.3.6 Definition Let G_{L} be the *goal reduction left* inference rule, defined in the following two (non-mutually exclusive) cases and shown in fig. 8 in terms of d and δ_{L} rules:

$$\mathsf{G}_{\mathsf{L}}\ \frac{\begin{bmatrix}\Psi\end{bmatrix} \vdash \begin{bmatrix}G_1\end{bmatrix} \quad \dots \quad \begin{bmatrix}\Psi\end{bmatrix} \vdash \begin{bmatrix}G_{k''}\end{bmatrix} \quad \begin{bmatrix}\Psi\\ \Gamma_1\end{bmatrix} \vdash \begin{bmatrix}H_1\\ \Lambda_1\end{bmatrix} \quad \dots \quad \begin{bmatrix}\Psi\\ \Gamma_{k'}\end{bmatrix} \vdash \begin{bmatrix}H_{k'}\\ \Lambda_{k'}\end{bmatrix}}{\begin{bmatrix}\Psi\\ \Gamma, G\end{bmatrix} \vdash \begin{bmatrix}\Lambda\end{bmatrix}} \qquad \text{or}$$

$$\mathsf{G}_{\mathsf{L}}\ \frac{\begin{bmatrix}\Psi, G\end{bmatrix} \vdash \begin{bmatrix}G_1\end{bmatrix} \quad \dots \quad \begin{bmatrix}\Psi, G\end{bmatrix} \vdash \begin{bmatrix}G_{k''}\end{bmatrix} \quad \begin{bmatrix}\Psi, G\\ \Gamma_1\end{bmatrix} \vdash \begin{bmatrix}H_1\\ \Lambda_1\end{bmatrix} \quad \dots \quad \begin{bmatrix}\Psi, G\\ \Gamma_{k'}\end{bmatrix} \vdash \begin{bmatrix}H_{k'}\\ \Lambda_{k'}\end{bmatrix}}{\begin{bmatrix}\Psi, G\\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}\Lambda\end{bmatrix}},$$

where $G = \forall \vec{x}.(\delta_1 \,\&\, \dots \,\&\, \delta_h)$, $\vec{x}$ can be empty, $\delta_l\sigma = G_1 \Rightarrow \dots \Rightarrow G_{k''} \Rightarrow H_1 \multimap \dots \multimap H_{k'} \multimap a_1 \parr \dots \parr a_k$, for $1 \leq l \leq h$ and $k, k', k'' \geqslant 0$, and where $\Gamma_1 \uplus \dots \uplus \Gamma_{k'} = \Gamma$ and $\Lambda_1 \uplus \dots \uplus \Lambda_{k'} \uplus \{a_1, \dots, a_k\}_+ = \Lambda$.

3.4 A System for Goals

In the previous section we built two big inference rules, a left one and a right one. Their definition might look complex, but it is in fact rather straightforward once one knows the (operational) meaning of linear logic connectives. One should notice that if the language of formulae were not restricted to goals, such an enterprise would really be cumbersome and complex, and probably pointless. Our point is, instead, that goals in Forum actually define sort of a 'generalised' connective. In the sequent calculus, connectives are defined by inference rules, and of course inference rules should behave correctly. An important technical meaning of this last requirement is that the rules allow for a cut elimination theorem.

We know already cut elimination from linear logic, but that theorem is proved inside the sequent system where all the usual connectives are defined. Even if we know that cut is admissible in our system, in order to test its internal harmony we have to provide a cut elimination proof that does not appeal to the underlying 'small' connectives. In this section we expose the main ideas, the details are available in the journal version of the paper [3].

3.4.1 Definition Let G-Forum be the formal system whose sequents are G-state sequents or sequents of the form $\begin{bmatrix}\Psi\\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}G\\ \Lambda\end{bmatrix}$, where Ψ and Γ contain goals, and whose inference rules are G_{L} and G_{R}.

Let us first define two natural cut rules for G-Forum.

3.4.2 Definition The following inference rules $\bowtie\!\lhd_{\mathsf{L}}$ and $\bowtie\!\lhd_{\mathsf{C}}$ are respectively called *cut linear* and *cut classical*:

$$\bowtie\!\lhd_{\mathsf{L}} \frac{\begin{bmatrix}\Psi\\ \Gamma\end{bmatrix} \vdash \begin{bmatrix}G\\ \Lambda\end{bmatrix} \quad \begin{bmatrix}\Psi'\\ G,\Gamma'\end{bmatrix} \vdash \begin{bmatrix}\Xi'\\ \Lambda'\end{bmatrix}}{\begin{bmatrix}\Psi,\Psi'\\ \Gamma,\Gamma'\end{bmatrix} \vdash \begin{bmatrix}\Xi'\\ \Lambda,\Lambda'\end{bmatrix}} \quad \text{and} \quad \bowtie\!\lhd_{\mathsf{C}} \frac{\begin{bmatrix}\Psi\end{bmatrix} \vdash \begin{bmatrix}G\end{bmatrix} \quad \begin{bmatrix}G,\Psi'\\ \Gamma'\end{bmatrix} \vdash \begin{bmatrix}\Xi'\\ \Lambda'\end{bmatrix}}{\begin{bmatrix}\Psi,\Psi'\\ \Gamma'\end{bmatrix} \vdash \begin{bmatrix}\Xi'\\ \Lambda'\end{bmatrix}}.$$

Ξ' is either empty or a singleton. In both rules G is called the eigenformula. System G-Forum$^{\bowtie\!\lhd_{\mathsf{L}},\bowtie\!\lhd_{\mathsf{C}}}$ is G-Forum where $\bowtie\!\lhd_{\mathsf{L}}$, $\bowtie\!\lhd_{\mathsf{C}}$ are allowed in proofs.

The following is the cut elimination theorem. Its proof follows a traditional argument in which one deals with contraction by a generalised cut rule that cuts on several copies of the same eigenformula (see for example [4]). We use the cut classical rule, in a certain generalisation $\bowtie\!\lhd'_{\mathsf{C}}$, together with a contraction rule, to make G-Forum$^{\bowtie\!\lhd_{\mathsf{L}},\bowtie\!\lhd_{\mathsf{C}}}$ more general, and then we prove cut elimination on this more general system. The core of the proof is the elimination of the $\bowtie\!\lhd_{\mathsf{L}}$ rule. Actually, the design of the rules G_{L} and G_{R}, and the crucial decisions about the exact definition of goals, all come from a careful analysis of what is needed in this part of the cut elimination argument. The induction measure is based on cut-rank.

3.4.3 Theorem *For every proof in* G-Forum$^{\bowtie\!\lhd_{\mathsf{L}},\bowtie\!\lhd_{\mathsf{C}}}$ *there exists a proof in* G-Forum *with the same conclusion.*

Proof (Sketch) We consider a contraction rule, $>$, and a cut classical rule in a more general form, $\bowtie\!\lhd'_{\mathsf{C}}$, and we show that they hold:

$$\text{G-Forum}^{>,\bowtie\!\lhd_{\mathsf{L}},\bowtie\!\lhd'_{\mathsf{C}}} \to \text{G-Forum}^{>,\bowtie\!\lhd_{\mathsf{L}}} \to \text{G-Forum}^{>} \to \text{G-Forum},$$

i.e., a proof in a left system can be transformed into a proof in a system at its right, having the same conclusion. The leftmost system is more general than G-Forum$^{\bowtie\!\lhd_{\mathsf{L}},\bowtie\!\lhd_{\mathsf{C}}}$, what yields the result. □

4 Conclusions

In this paper we showed G-Forum, an asymmetrical sequent system for linear logic, and we proved cut elimination for it. Like in the case of Forum, the left-right asymmetry of sequents is motivated by the necessity of limiting proof search to uniform proofs. We also imposed an asymmetry to formulae, in such a way that their structure matches that of sequents. This new asymmetry is motivated by the desire of structuring proofs by easily definable, big building blocks, suitable to semantic understanding. We consider the situation where two asymmetries match rather pleasant, and more symmetric than the same in Forum, where the structure of formulae is symmetric, so at odds with that of sequents.

The result is a system for which it is natural to define cut rules and for which it is possible to prove cut elimination by a procedure that rewrites proofs inside the system, without resorting to Forum or plain linear logic. This guarantees that the

new system has a good proof theoretical standing, which usually means that it is a good basis for further, fruitful research.

In a forthcoming paper we will show how to associate to G-Forum a labelled event structure semantics, i.e., a behavioural model of computation, along the lines initiated in [6]. In another paper we will apply G-Forum and its semantics to problems of partial order planning.

The methods in this paper are about studying the structure of proofs at a coarser abstraction level than the one provided by the sequent calculus. In another research project we are pursuing about the calculus of structures [2, 7, 1, 11] (see also our web site at http://www.ki.inf.tu-dresden.de/~guglielm/Research), we study proofs at a *finer* level than provided by the sequent calculus. We do so for exploring properties of locality and modularity, which are important for computer science, that are otherwise not available. In the future, we plan to adapt the techniques in this paper to the calculus of structures (for example, of linear logic), in order to cover the full range of abstractions: from the finer, suitable for distributed implementation, to the coarser, suitable for semantics.

References

[1] Kai Brünnler and Alwen Fernanto Tiu. A local system for classical logic. In R. Nieuwenhuis and A. Voronkov, editors, *LPAR 2001*, volume 2250 of *Lecture Notes in Artificial Intelligence*, pages 347–361. Springer-Verlag, 2001. URL: http://www.ki.inf.tu-dresden.de/~kai/LocalClassicalLogic-lpar.pdf.

[2] Paola Bruscoli. A purely logical account of sequentiality in proof search. In Peter J. Stuckey, editor, *Logic Programming, 18th International Conference*, volume 2401 of *Lecture Notes in Artificial Intelligence*, pages 302–316. Springer-Verlag, 2002. URL: http://www.ki.inf.tu-dresden.de/~paola/bvl/bvl.pdf.

[3] Paola Bruscoli and Alessio Guglielmi. On structuring proof search for first order linear logic. Technical Report WV-03-10, Technische Universität Dresden, 2003. URL: http://www.ki.inf.tu-dresden.de/~paola/sps/sps.pdf.

[4] Jean Gallier. Constructive logics. Part I: A tutorial on proof systems and typed λ-calculi. *Theoretical Computer Science*, 110:249–339, 1993.

[5] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[6] Alessio Guglielmi. *Abstract Logic Programming in Linear Logic—Independence and Causality in a First Order Calculus*. PhD thesis, Università di Pisa, 1996.

[7] Alessio Guglielmi. A system of interaction and structure. Technical Report WV-02-10, Technische Universität Dresden, 2002. Submitted to ACM Transactions on Computational Logic. URL: http://www.ki.inf.tu-dresden.de/~guglielm/Research/Gug/Gug.pdf,.

[8] Dale Miller. A multiple-conclusion meta-logic. In S. Abramsky, editor, *Ninth Annual IEEE Symp. on Logic in Computer Science*, pages 272–281, Paris, July 1994.

[9] Dale Miller. Forum: A multiple-conclusion specification logic. *Theoretical Computer Science*, 165:201–232, 1996.

[10] Dale Miller, Gopalan Nadathur, Frank Pfenning, and Andre Scedrov. Uniform proofs as a foundation for logic programming. *Annals of Pure and Applied Logic*, 51:125–157, 1991.

[11] Lutz Straßburger. MELL in the calculus of structures. Technical Report WV-01-03, Technische Universität Dresden, 2001. URL: http://www.ki.inf.tu-dresden.de/~lutz/els.pdf, to appear in Theoretical Computer Science.

Typeset with Xy-pic