Pomset logic: a non-commutative extension of classical linear logic

Christian Retoré

retore@loria.fr http://www.loria.fr/~retore
Projet Calligramme, INRIA-Lorraine & CRIN-C.N.R.S.
B.P. 101 54 602 Villers lès Nancy cedex France

Abstract. We extend the multiplicative fragment of linear logic with a non-commutative connective (called `before`), which, roughly speaking, corresponds to sequential composition. This lead us to a calculus where the conclusion of a proof is a Partially Ordered MultiSET of formulae.
We firstly examine coherence semantics, where we introduce the `before` connective, and ordered products of formulae. Secondly we extend the syntax of multiplicative proof nets to these new operations.
We then prove strong normalisation, and confluence.
Coming back to the denotational semantics that we started with, we establish in an unusual way the soundness of this calculus with respect to the semantics. The converse, i.e. a kind of completeness result, is simply stated: we refer to a report for its lengthy proof.
We conclude by mentioning more results, including a sequent calculus which is interpreted by both the semantics and the proof net syntax, although we are not sure that it takes all proof nets into account.
The relevance of this calculus to computational linguistics, process calculi, and semantics of imperative programming is briefly explained in the introduction.

Introduction

Motivation. Linear logic [10, 36, 12] succeeds in modelling computational phenomena because it is both a neat logical system and a resource sensible logic. This explains its relevance to various areas like: (1) process calculi and concurrency, e.g. [1, 4, 18, 19, 25], (2) functional programming, e.g. [1, 14], (3) logic programming, e.g. [17, 25].

In some situations *non-commutative* features would be welcome, to handle phenomena like: prefixing or sequential composition in area (1), strategy optimisation in areas (2,3). In other areas these non-commutative features are even necessary to obtain a logical model: for instance if one wants a logical description of state manipulation as appearing in the semantics of imperative programming [35] or if one wants a logical system corresponding to a categorial grammar in computational linguistics [20, 24].

One possible direction consists in leaving out the exchange rule, i.e. in working with non-commutative connectives, as in [2, 3], but it is then highly difficult to also include commutative features, which are needed too.

Principles of the calculus. Here we *extend* multiplicative linear logic with a non commutative connective which is the only possible one according to coherence semantics. This connective, called `before` $<$, is non-commutative ($A < B \not\equiv B < A$), self-dual ($(A < B)^{\perp} \equiv A^{\perp} < B^{\perp}$), and associative. Furthermore, its semantical definition suggests considering n-ary connectives defined by partial orders.

This led us to consider in [28, pp. 85–139], a calculus where the connectives are linear multiplicative conjunction $\otimes$, disjunction $\wp$, and `before` $<$, and where the conclusion of a proof is a partially ordered multiset of formulae. This article is devoted to the syntax of proof nets for this calculus, which is a natural extension of multiplicative proof nets enriched with the *mix* rule [8, pp. 277–278].

Results. We first describe the semantical construction which gave rise to this calculus, and next we turn our attention to the proof net syntax.

As usual, we first define proof structure, and only the proof structures enjoying a global criterion are called proof nets.

The correctness criterion simply extends the usual one, in the style of [7, pp. 193–194], but we give here a formulation which is new with respect to [28, pp. 93–100], with edge bicoloured graphs, i.e. in the style of [32, p. 8]. It is thus immediate that this criterion may be checked in polynomial time: this makes the proof net syntax a sensible syntax by itself.

We then define cut-elimination, as three local graph rewriting rules. Once we have proved that these rules preserve the correctness of the proof net, we very easily get *strong normalisation* and *confluence*.

Next we provide a denotational semantics for this calculus using the semantics defined in the beginning, and prove that each proof admits a non-trivial semantics preserved under cut-elimination. This is done by an extension of the experiment method of [10, pp. 57–60]. In other words, we show that our calculus is sound w.r.t. this semantics, seizing the opportunity of a method which does not rely on an inductive definitions of the proofs. We state a kind of converse, i.e. a kind of completeness result with respect to this semantics, but the proof which can be found in [29, pp. 19–22], is too lengthy to be given here — as opposed to the usual case which is very simple, see [33].

Finally we briefly explain that the partial orders which correspond to the n-ary connectives definable from the binary connectives exactly correspond to series parallel orders — see e.g. [37]. The formulae corresponding to series-parallel orders do not involve the `times` connective, and the `before` connective corresponds to series composition, while the `par` connective corresponds to parallel composition.

We conclude by giving some more results that we obtained on this calculus, mainly in [28, pp. 85–139], and then raise the difficult question of finding a sequent calculus *exactly* corresponding to these proof nets — a sound one, found in [28, pp. 111–122], is briefly given, but we are unable to show that it takes all the proof nets into account.

Relation to other studies in linear logic. Regarding non-commutative linear logic, we succeed in having both the usual commutative connectives `par` and `times` together with a non-commutative connective, and to endow our calculus with a simple denotational semantics.

But this calculus is also a new approach for dealing with n-ary connectives, here defined by partial orders. Regarding the unrelated attempt of [9, 7], we succeed in giving the connectives we define a denotational semantics, and a sequent calculus — even if it is not a complete one.

In fact, with respect to the two aforementioned trends, our main success is to provide a simple computational meaning to our n-ary connectives and non-commutative features. Indeed, the cuts, i.e. the computation to be performed, are also involved in the partial order, and therefore a concurrent strategy is described within the syntax itself. This fits in with the computational interpretation of linear logic of [1] and is the starting point of the use [4, 5] and [26, 27] made of this calculus.

Computational meaning of `before` and of partial orders. As usual cut links may be viewed as particular final `times` links, which allows us to make them appear in the partial order. This partial order may thus be viewed as a strategy for computing the proof net, which is described within the syntax itself. This is a concurrent strategy, which simply consists in first evaluating the cuts (i.e. the computations to be performed) according to this order.

Notice that our calculus relates to true concurrency rather than to CCS-like calculi where $(P|Q) = P.Q + Q.P$ using the notation of [23].

Also notice that a cut between $A \wp B$ and $A^\perp \otimes B^\perp$ reduces to a cut between A and $A^\perp$ and a cut between B and $B^\perp$ that can be done in *parallel*, while a cut between $A < B$ and $A^\perp < B^\perp$ reduces to a cut between A and $A^\perp$ which is to be computed first and a cut between B and $B^\perp$ which is to be computed next — the identity $(A < B)^\perp \equiv A^\perp < B^\perp$ *without swap*, as opposed to [2, 3], is needed to allow such an interpretation.

That is the reason why $A < B$ is to be intuitively understood as *sequential composition*, as shown in [4].

In the plain logical calculus the order only describes a strategy, since we can forget (part of) it and still obtain a proof net. Nevertheless when there are proper axioms modelling some "real world" constraints, as the order appearing in these axioms merges with the order introduced by the proof net, we can no more forget parts of the order, and we are thus able to model temporal constraints, like "A ought to be done before B".

Applications. This work has already been used, on the basis of [28], to provide some solutions or insights to the aforementioned fields:

In [4, 5] the authors consider proof nets as processes, and show that the correctness of the proof net corresponds to freedom from deadlock. The connectives are interpreted as follows: $\wp$ parallel composition, $\otimes$ internal choice, and $<$ sequential composition.

In [26, 27] the author makes use of the calculus to model by local means state change in imperative programming. For instance, a buffer of type A has the type $!(A^\perp < A)$: thus it forces that it *first* get something of type A (write), and *then* produce something of type A (read).

In [15] the author suggests that in the *proof search as computation* paradigm the order we have on the formulae, should force some subgoals to be proved before some others.

In [21, 34], this calculus is applied to categorial grammars — see e.g. [20, 24]. We seize the opportunity that this calculus is able to handle *partial* orders instead of *linear* orders [2, 3, 20] to provide a logical treatment of linguistics phenomena hitherto absent from categorial grammars, like relatively free word order, gapping, head wrapping, as explained in [21, 34]. In such a grammar, the lexicon associates each word with a partial proof net, which contains an axiom labelled with the word. To analyse a sentence we must first make a complete proof net with the parts corresponding to the words of the sentence, i.e. check if the criterion holds (the consumption of the valencies is correct), and then check whether the induced order is included in the word order of the sentence (the order of the words is correct). Just to mention one example, we are thus able to model correctly French perception verbs: in this grammar, both *Pierre entend Marie chanter* et *Pierre entend chanter Marie* are recognised as correct sentences, but, and that is the most important, they both come out with the *same* analysis.

1 A guideline: coherence semantics

General framework Coherence spaces are a denotational semantics tightly related to linear logic, as explained in [10, 36, 12], and in particular to its proof net syntax [10, 33, 29]. Actually linear logic was even discovered through this semantics, which belongs to the world of stable semantics, introduced in [6] and which incorporates more computational behaviour than plain Scott semantics.

A **coherence space** A is a simple graph, i.e. a set endowed with a symmetric and anti-reflexive relation. The vertices are called **tokens**, and their set is called the **web** of the coherence space A, denoted by $|A|$. Given two tokens $a, a' \in |A|$, write $a \frown a'[A]$ for a and a' are adjacent, or **strictly coherent**, and $a \smile a'[A]$ for a and a' are neither adjacent nor equal, i.e. are **strictly incoherent**.

A **clique** of a coherence spaces is a set of pairwise adjacent vertices or coherent tokens. A **linear morphism** or **linear map** from a coherence space A to a coherence space B, is a relation $\ell \in |A| \times |B|$ satisfying:

$$\forall (a, b), (a', b') \in \ell \quad \big(a = a' \Rightarrow (b = b' \vee b \frown b'[B])\big) \wedge \big(a \frown a'[A] \Rightarrow b \frown b'[B]\big)$$

Linear morphisms compose as relations, and coherence spaces with linear morphisms form a category. Let us write $A \equiv B$ whenever there exists a canonical linear morphism from A to B and one from B to A, one being the inverse of the other.

The interpretation of the different levels of linear logic within coherent spaces proceeds as follows:

Syntax	Semantics
formula F	**coherence space** also denoted by F
propositional variable α	arbitrary coherence space α
(n-ary) connective	(n-ary) operation on coherence spaces
proof of a formula F	**clique** of the corresponding coherence space F
proof of a sequent $A \vdash B$	linear morphism from A to B
normalisation of a proof	**equality** of the corresponding clique(s)

Figure 1

As usual a proof of a sequent is interpreted as the proof derived from it by replacing the left and right commas with the corresponding connectives, i.e. a proof of $A_1 \otimes \cdots \otimes A_n \vdash B_1 \wp \cdots \wp B_p$. Furthermore, given two coherence spaces A and B, there exists a coherence space $A \multimap B = A^\perp \wp B$, to be defined next, whose cliques correspond to linear morphisms from A to B.

Introducing `before` as *the* non-commutative multiplicative connective
As shown in Figure 1, the first step towards such a semantics is to interpret the formulae.

Once the interpretation of propositional variables is set, we just need to have an n-ary operation (more precisely, an n-ary functor) interpreting each n-ary connective. For instance linear negation, the unary connective $(\ldots)^\perp$, is the idempotent contravariant functor defined by:

$A^\perp$ is defined by $|A^\perp| = |A|$ and $a \frown a'[A^\perp]$ whenever $a \smile a'[A]$

$\ell^\perp = \{(b, a) \in |B| \times |A| \ / \ (a, b) \in \ell\}$

Among the connectives, the ones that map n coherence spaces $A_1, \ldots, A_n$ on a coherence space whose web is the Cartesian product $|A_1| \times \cdots |A_n|$ are said to be **multiplicative** connectives. So linear negation is a unary multiplicative connective. A connective is said to be positive whenever as a functor it is covariant in all its argument. With the help of the (contravariant) linear negation, they are the basic connectives from which the others may be defined as short hands.

Let $\heartsuit$ be a positive binary multiplicative connective. To define it we must specify according to the coherence of a and a' (in A), and to the one of b and b' (in B) the coherence of two pairs (a, b) and (a', b'). This can be pictured in a 3×3 array, but if $\heartsuit$ is positive, all the 9 cases are *a priori* filled in, but two, which are $\smile\heartsuit\frown$ and $\frown\heartsuit\smile$ (Figure 2). From this array, we observe that there only exist four binary multiplicative connectives positive in both their arguments, the two commutative ones being well-known, as shown in Figure 3.

$A\heartsuit B$		B		
		$\smile$	$=$	$\frown$
A	$\smile$	$\smile$	$\smile$	$\smile\heartsuit\frown$
	$=$	$\smile$	$=$	$\frown$
	$\frown$	$\frown\heartsuit\smile$	$\frown$	$\frown$

Figure 2

The multiplicative (positive) binary connectives				
$\smile\heartsuit\frown$	$\frown\heartsuit\smile$	commutative	notation	name
$\frown$	$\frown$	yes	$A \wp B$	`par`
$\smile$	$\smile$	yes	$A \otimes B$	`times`
$\frown$	$\smile$	no	$A < B$	`before`
$\smile$	$\frown$	no	$B < A$	"reverse `before`"

Figure 3

Regarding the two commutative connectives, they are known to be associative and to enjoy the De Morgan laws, which turn one into the other: $(A\otimes B)^{\perp}\equiv A^{\perp}\wp B^{\perp}$ and $(A\wp B)^{\perp}\equiv A^{\perp}\otimes B^{\perp}$. What are the corresponding properties of `before`, and what is its relation to its commutative companions? A mere computation shows that:

Proposition 1. *The* `before` *connective is:*

- non-commutative $A < B \not\equiv B < A$
- self dual $(A < B)^{\perp} \equiv A^{\perp} < B^{\perp}$
- associative $A < (B < C) \equiv (A < B) < C$
- *regarding linear implication it lies* in between `par` and `times`, *i.e. there exists a canonical linear morphism from* $A \otimes B$ *to* $A < B$ *and one from* $A < B$ *to* $A \wp B$. *— the relation(s) defining these linear maps simply being the identity relation:* $\{((a,b),(a,b))/(a,b)\in |A\times B|\}$.

Notice that coherence according to `before` may be defined in a lexicographic manner: $(a,b)\frown(a',b')[A<B]$ iff $(a\frown a'[A]\wedge b=b')\vee b\frown b'[B]$. This suggests introducing the following n-ary multiplicative connectives: let $(A_i)_{(i\in I)}$ be a family of coherence spaces ordered by a partial order $\mathfrak{u}$, written $A_k < A_l[\mathfrak{u}]$. We define the **ordered product** of this ordered family as the coherence space $\prod_{\mathfrak{u}} A_i$ whose web is the Cartesian product of the webs of the A_i's — $|\prod_{\mathfrak{u}} A_i| = |A_1|\times\ldots\times|A_n|$ — and whose strict coherence is defined by:

$$(a_1,\ldots,a_n)\frown(a'_1,\ldots,a'_n)\ [\textstyle\prod_{\mathfrak{u}} A_i] \text{ iff } \exists i\,.\, a_i\frown a'_i[A_i]\wedge\forall A_j{>}A_i[\mathfrak{u}]\ a_j{=}a'_j$$

Notice that $\prod_{\varnothing} A, B \equiv A \wp B$ while $\prod_{A<B} A, B \equiv A < B$.

The next sections present proofs whose semantical interpretations will be cliques of these ordered products of coherence spaces.

2 Language and sequents

Without any further structure on the conclusions of a proof, or sequent, there is no way to introduce this non-commutative connective `before`— there are only two possible multiplicative rules, which are the usual rules for `times` and `par` of linear logic.

As it is suggested by the semantics above to work with partially ordered multiset of formulae, let us look for a logical calculus whose conclusions will be ordered multisets of formulae of $\mathcal{F} ::= \mathcal{P} \mid \mathcal{F}^\perp \mid \mathcal{F} \wp \mathcal{F} \mid \mathcal{F} < \mathcal{F} \mid \mathcal{F} \otimes \mathcal{F}$, where $\mathcal{P}$ is a set of propositional variables.

On the set $\mathcal{F}$ of formulae, we have the following De Morgan laws:
$(A^\perp)^\perp \equiv A \quad (A \wp B)^\perp \equiv A^\perp \otimes B^\perp \quad (A < B)^\perp \equiv A^\perp < B^\perp \quad (A \otimes B)^\perp \equiv A^\perp \wp B^\perp$.

Therefore, as is usual in the theory of proof nets, we shall only consider formulae up to De Morgan equivalence. Indeed, each formula of $\mathcal{F}$ has a unique representative in the following set of formulae:

$$\mathcal{M} ::= \mathcal{P} \mid \mathcal{P}^\perp \mid \mathcal{M} \wp \mathcal{M} \mid \mathcal{M} < \mathcal{M} \mid \mathcal{M} \otimes \mathcal{M}$$

Consequently, we shall only consider formulae of $\mathcal{M}$, and $F^\perp$ should be understood as the unique formula F' of $\mathcal{M}$ which is equivalent to the formula $(F)^\perp$ of $\mathcal{F}$.

Another familiar property underlined by proof nets, is that a cut may be viewed as a **times** $K \otimes K^\perp$ between the two dual formulae K and $K^\perp$ that vanish in a cut rule.[1] Here we shall use this view of cuts as **times** formulae to keep a track of these cuts: thus the partial order on the formulae of a sequent may also involves the cuts, i.e. the computations to be performed.

The conclusion of a proof will be a partially ordered multi-set of formulae and cuts written:

$$\vdash A_1, \ldots, A_n, G_1^\bullet, \ldots, G_p^\bullet[\mathfrak{u}] \quad \text{with:}$$

- $A_1, \ldots, A_n$ being formulae of $\mathcal{M}$
- $G_1^\bullet = X_1 \otimes X_1^\perp, \ldots, G_p^\bullet = X_p \otimes X_p^\perp$, being cuts — where $\forall i \in [1,p]\ X_i \in \mathcal{M}$.
- $\mathfrak{u}$ being an order on the multiset $\{A_1, \ldots, A_n, G_1^\bullet, \ldots, G_p^\bullet\} \subset \mathcal{M}$

We now define a proof syntax dealing with such conclusions, in the proof net style, because this calculus is a simpler extension of the proof net syntax than of the sequent syntax.

3 Ordered proof nets

In this section we introduced ordered proof nets in the framework developed in [32] for the usual multiplicative calculus, by extending the definition of a R&B-graph to the directed case.

We first present ordered proof structures from a kind of sub-formula trees, R&B-trees — this is the most intuitive definition — , and then define them à la Girard with links, — and that will be more convenient for the proofs in the next sections.

Directed R&B-graphs A **R&B-graph** $G = (V; B, R)$ is an edge-bi-coloured graph such that:

[1] To be more precise, we should apply some second order existential quantification to this times formula to obtain $\exists X.X \otimes X^\perp$ which is equivalent to $\perp$.

– V is a set of **vertices**
– R is a set of ordered pairs $(x, y) \in V^2$ such that $x \neq y$ — in case we have both (x, y) and (y, x) in R we speak of the **R-edge** $x - y$; in case we have $(x, y) \in R$ and $(y, x) \notin R$ we speak of the **R-arc** $x \rightarrow y$.
– B is a set of ordered pairs $(x, y) \in V^2$ such that
  - $x \neq y$ — no loop
  - $(x, y) \in B \Rightarrow (y, x) \in B$ — B is a set of edges, as opposed to arcs, and the **B-edge** $\{(x, y), (y, x)\}$ will be simply denoted by $x - y$
  - $\forall x \exists! y \ (x, y) \in B$ — B is a perfect matching of the full graph including B and R edges

These R&B-graphs clearly can be pictured as edge-bicoloured graphs with vertices V in which B-edges are Bold (or Blue), an R-arcs or R-edges are Regular (or Red). In a R&B-graph, an **alternating path** p of length n from x_0 to x_n is a sequence of n consecutive arcs $p = (x_0, x_1)(x_1, x_2) \cdots (x_{n-1}, x_n)$ alternatively in B and in R, that is to say $(x_{i-1}, x_i) \in R \Rightarrow (x_i, x_{i+1}) \in B$ and $(x_{i-1}, x_i) \in B \Rightarrow (x_i, x_{i+1}) \in R$. An alternating path is said to be **elementary**, whenever no vertex appears more than twice in its formal expression. In this case we speak of an **æ-path**. In case $x_0 = x_n$ and n is even we speak of an alternating elementary circuit, **æ-circuit** for short.

Ordered proof nets as directed R&B-graphs

Definition 2. Given a formula C, we defines its R&B-trees as edge bicoloured graphs.[2] For every formula C, the discrete graph consisting of a vertex C° is a R&B-tree of C, with root C, and leaf C. If C is a non atomic formula, namely $C = A * B$ with $* \in \{\wp, \otimes, <\}$, and if $T(A)$ and $T(B)$ are respectively R&B-trees of A and B, then $T(A) * T(B)$, defined in Figure 4, is a R&B-tree of C, with root C, and both the leaves of $T(A)$ and $T(B)$ as leaves.

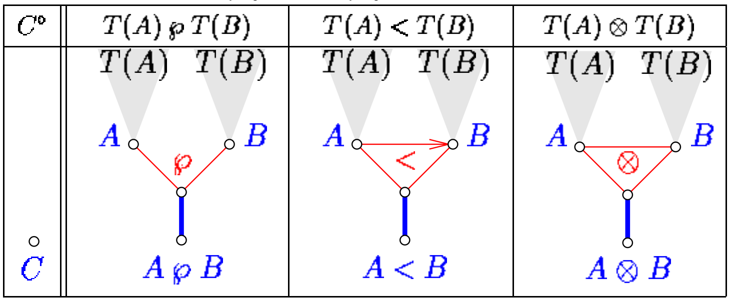


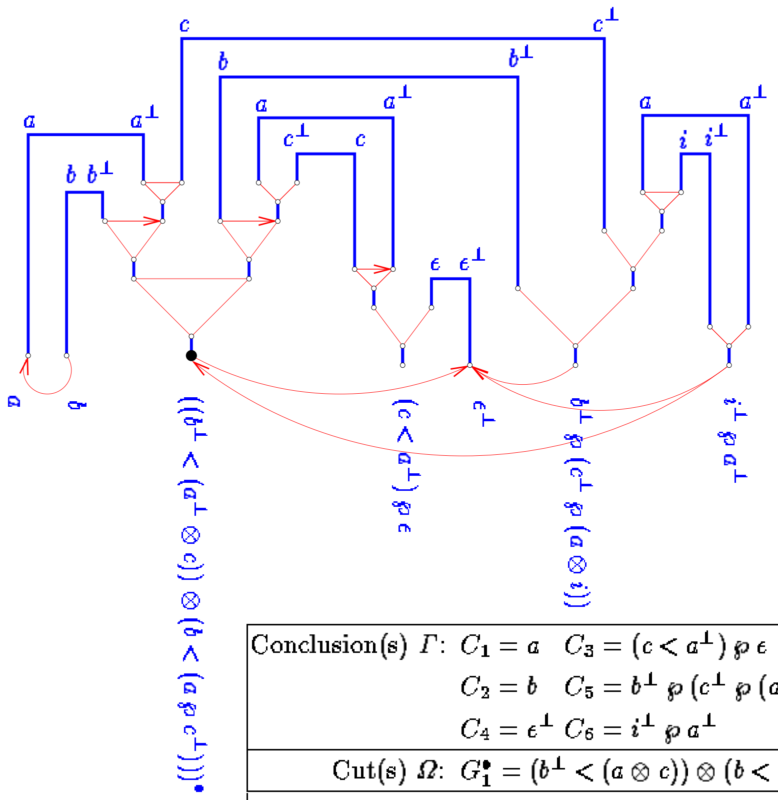
Figure 4

Thus each vertex of a R&B-tree $T(C)$ of a formula C, is labelled with a sub-formula of C, or with a connective. Notice that only the labels of the leaves are needed to reconstruct all the labels of a R&B-tree.

[2] They are neither trees not R&B-graphs; but they look like truncations of the sub-formula tree, and there is *at most*, instead of *exactly*, one B-edge incident to a vertex.

Definition 3. A proof structure with $C_1, \ldots, C_n, G_1^\bullet, \ldots, G_p^\bullet$ as conclusions and cuts — thus $G_i = X_i \otimes X_i^\perp$ — with order $\mathfrak{u}$ on its conclusions and cuts is a R&B-graph which consists of:

- a family of R&B-trees $T(C_1), \ldots, T(C_n), T(X_1) \otimes T(X_1^\perp), \ldots, T(X_p) \otimes T(X_p^\perp)$ where $T(X)$ denotes a R&B-tree of X. This part represents the syntactic forest of the sequent $C_1, \ldots, C_n, X_1 \otimes X_1^\perp, \ldots, X_p \otimes X^\perp$
- a family of B-edges, called *axioms*, each of them linking two leaves being the negation one of the other, in such a way that for each leaf there exists exactly one axiom incident to it.
- a family of R-*arcs*, representing the *order*, i.e. there is one such R-arc from a conclusion or cut X to another Y whenever $X < Y[\mathfrak{u}]$
- a special mark, $\bullet$, on the roots of the $T(G_i)$ to make a distinction between a cut and a conclusion $X \otimes X^\perp$ which is not considered as a cut.



Conclusion(s) Γ:	$C_1 = a$	$C_3 = (c < a^\perp) \wp \epsilon$
	$C_2 = b$	$C_5 = b^\perp \wp (c^\perp \wp (a \otimes i))$
	$C_4 = \epsilon^\perp$	$C_6 = i^\perp \wp a^\perp$
Cut(s) Ω:	$G_1^\bullet = (b^\perp < (a \otimes c)) \otimes (b < (a \wp c))$	
Order on $\Gamma \cup \Omega$:	$C_6 < G_1^\bullet$; $G_1 < C_4$; $C_5 < C_4$	

Figure 5

Now, as usual, not every proof structure corresponds to a proof, but only the proof nets:

Definition 4. A **proof net** is a proof structure which contains no æ-circuit. [3]

The example of figure 5 is a proof net but if the conclusion $C_5 = b^\perp \wp (c^\perp \wp (a \otimes i))$ were to be replaced with $C'_5 = b^\perp < (c^\perp \wp (a \otimes i))$, there would be an æ-circuit: $[b] -_{\mathtt{B}} [b^\perp] \rightarrow_{\mathtt{R}} [<] -_{\mathtt{B}} [b^\perp < (c^\perp \wp (a \otimes i))] -_{\mathtt{B}} [\wp] -_{\mathtt{R}} [c^\perp] -_{\mathtt{B}} [c] -_{\mathtt{R}} [\otimes] -_{\mathtt{B}} [a^\perp \otimes c] -_{\mathtt{R}} [<] -_{\mathtt{R}} [b^\perp < (a^\perp \otimes c)] -_{\mathtt{R}} [b < (a \wp c^\perp)] -_{\mathtt{B}} [<] -_{\mathtt{R}} [b]$

Here is an important proposition, albeit easy, which means that the proof net syntax is a sensible syntax by itself:

Proposition 5. *There is a polynomial (cubic) algorithm which checks whether a proof structure is a proof net.*

Proof. Let X and Y be two vertices in a given proof structure. Checking whether there exists an æ-path from X to Y starting with its unique incident B-edge is a standard breadth search algorithm. Each B-edge is visited once in each direction, and it is thus quadratic in twice the number of B-edges, i.e. in the number of vertices (the B-edges are a perfect matching of the graph). If we take $Y = X$ and repeat this for any vertex X, we get a cubic algorithm which checks the absence or presence of æ-circuit.

Let us now define the **links** of the ordered proof structures, i.e. the bricks they are made of, and their premises and conclusions. A $*$-link, with $* \in \{\otimes, \wp, <\}$ is the R&B-graph on four vertices A, B, $*$, $A * B$, which appears in figure 4; A and B are said to be its premises, and $A * B$ is said to be its conclusion. An axiom link is an axiom, i.e. a B-edge whose end vertices are A and $A^\perp$. This link has no premise and two conclusions, namely A and $A^\perp$. An ordered proof structure may also be defined as a set of links such that each formula is the conclusion of exactly one link, and the premise of at most one conclusion, plus a family of R-arcs which is an order between conclusions, i.e. between the formulae which are not the premise of any link.

4 Cut elimination

We now define cut-elimination as a local graph rewriting system which turns a proof net into a proof net, in such a way that *the restriction of the order to the conclusions is preserved under cut elimination.* A cut is a **times** link between two dual formulae K and $K^\perp$. Each of the formulae K and $K^\perp$ is the conclusion of a unique link, say k and $k^\perp$. There are three elementary steps of cut elimination to be described, according to the nature of the links k and $k^\perp$:

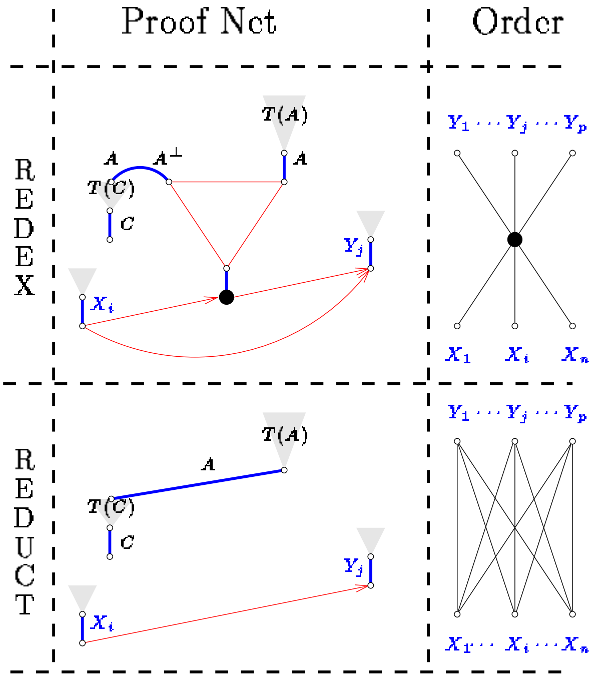
AX/? k or $k^\perp$ is an **axiom** link

BF/BF k and $k^\perp$ are **before** links

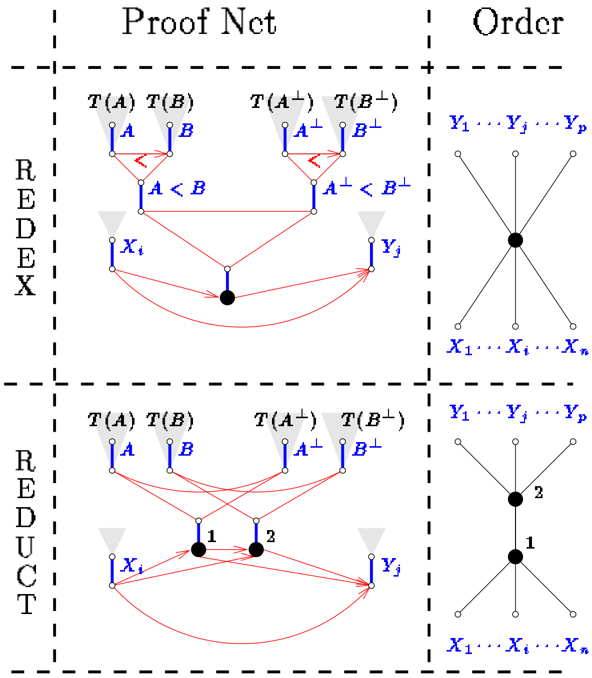
TS/PAR k is a **times** link and $k^\perp$ a **par** link.

[3] Beware that the adjective *elementary* in *alternating elementary circuit* is necessary. For instance, there exist proof nets of the usual multiplicative calculus, hence of this extension that we are presently defining, which contain alternating circuits.

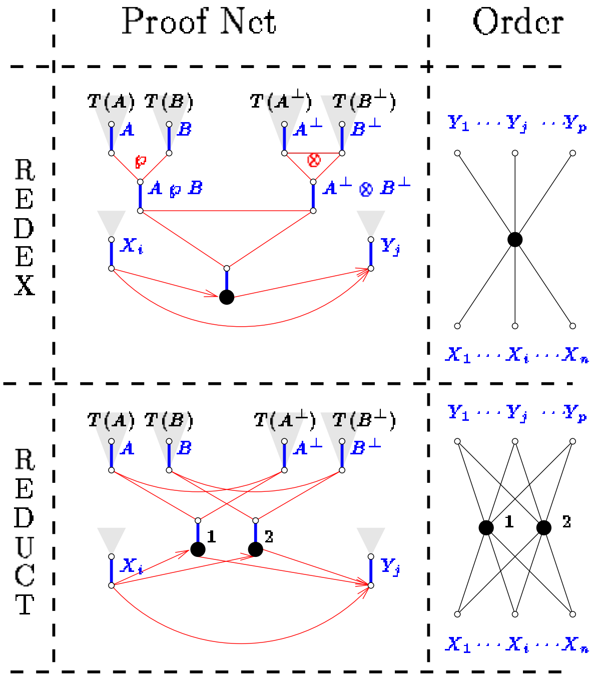
The AX/? **elementary step** Here is the picture of this elementary step. The vertex X_i (resp. Y_j) is a conclusion or a cut which is, according to the order between conclusions and cuts, below (resp. above) the cut we are reducing and there may be several such conclusions X_i (resp. Y_j). We first suppress the **axiom** link and then the **cut**/**times** link and its incident R-arcs. We then identify the vertices labelled with A.
The order on the conclusions and cuts in the reduct simply is the restriction of the order in the redex to the remaining conclusions and cuts.



The BF/BF **elementary step** Here is the picture of this elementary step. The vertex X_i (resp. Y_j) is a conclusion or a cut which is, according to the order on conclusions and cuts, below (resp. above) the cut we are reducing, and there may be several such conclusions X_i (resp. Y_j).
In the order, the cut $\bullet$ is split into two cuts $\bullet_1$ and $\bullet_2$ with $\bullet_1 < \bullet_2$, which occupy the place of $\bullet$.



The TS/PAR elementary step Here is the picture of this elementary step. The vertex X_i (resp. Y_j) is a conclusion or a cut which is, according to the order on conclusions and cuts, below (resp. above) the cut we are reducing, and there may be several such conclusions X_i (resp. Y_j).
In the order, the cut $\bullet$ is split into two cuts $\bullet_1$ and $\bullet_2$, unordered, which occupy the place of $\bullet$.



Proposition 6. *The elementary steps* AX/?, TS/PAR, *and* BF/BF *preserve the absence of æ-circuit, as well as the order between the remaining conclusions and cuts.*

Proof. The last part of the statement is obvious for all elementary steps, from the Hasse diagrams of the orders of the redex and the reduct.

Let us call Π the original proof net and Π' its reduct. We now show that if Π contains no æ-circuit, so does Π', for each elementary step.

AX/? Observe that any æ-path in the reduct defines an æ-path (whose endings have the same colour) in the reduct.

BF/BF Let Π^- be the part of Π which is common to Π', and let π' the subgraph of Π' consisting in the two B-edges 1 and 2 incident with $\bullet^1$ and $\bullet^2$, and their adjacent R-edges. We assume there is an æ-circuit in Π'.

1. If it uses the R-arc $\bullet^1 \to \bullet^2$, then it uses:
   - (either the R-edge from A to 1 or the R-edge from $A^\perp$ to 1), and
   - the B-edge 1, the R-arc $\bullet^1 \to \bullet^2$, the B-edge 2, and (either the R-edge from B to 2 or the R-edge from $B^\perp$ to 2).

   Because Π is a proof net, Π^- can contain neither any æ-path from B to A, nor one from $B^\perp$ to $A^\perp$, nor one from B to $A^\perp$, nor one from $B^\perp$ to A. Hence there is no æ-circuit using $\bullet^1 \to \bullet^2$.
2. If it uses an R-edge from some X_i to $\bullet^1$ or $\bullet^2$, there should be in Π^- an æ-path from a vertex of π' to X, but this is impossible because Π is a proof net.

3. For a symmetrical reason, it cannot use an R-edge from $\bullet^1$ or $\bullet^2$ to some Y_j.
4. Hence it can not use the B-edges 1 and 2, nor their adjacent R-edges and arcs, i.e. it can only use the R-edges $A - A^\perp$ and $B - B^\perp$. Because Π is a proof net, it is obvious that it cannot use only one of these two R-edges. For the same reason, Π^- can contain neither a path from A or $A^\perp$ to B or $B^\perp$ nor one from B or $B^\perp$ to A or $A^\perp$. But Π^- can only contain an æ-path from A to B and one from $A^\perp$ to $B^\perp$, and this cannot produce an æ-circuit with the R-edges $A - A^\perp$ and $B - B^\perp$.

Therefore an æ-circuit of Π' does not pass through π', hence should be included in Π^- which is a subgraph of the proof net Π, contradiction.

TS/PAR We proceed as in the BF/BF case, except that the first item can be left out, and that the last sentence of the fourth item should be replaced with: *But Π^- can only contain an æ-path from A to B and one from B to A, and this cannot produce an æ-circuit with the* R*-edges $A - A^\perp$ and $B - B^\perp$.*

Confluence and strong normalisation During all elementary steps, the number of B-edges of the proof net decreases. Moreover the redex configurations are disjoint. Therefore, the previous proposition 6 entails the following:

Theorem 7. *The calculus of ordered proof nets enjoys strong normalisation and confluence: a proof net with conclusions and cuts $F_1, ..., F_n, G_1^\bullet, ..., G_p^\bullet$ ordered by $\mathfrak{u}$ reduces to a cut free proof net with conclusions $F_1, ..., F_n$ ordered by $\mathfrak{u}|_{F_1,...,F_n}$.*

5 Denotational semantics for this calculus

We compute the semantics $\| \Pi \|$ of a proof net Π by extending the method of experiments of [10, pp. 57–60] or of [29, 33] to this calculus.

We assume that we have an interpretation, i.e. that any atomic formula is associated with a coherence space. Consequently, to each formula F is associated a coherence space also denoted by F, by means of the interpretation of the connectives given in the first section.

Definition 8. Let Π be proof net with conclusion $\vdash F_1, ..., F_p, F_1^\bullet, ..., F_n^\bullet[\mathfrak{u}]$. An **experiment** of Π is a labelling of the sub-formulae appearing in the proof net satisfying:

- the label of a sub-formula F is a token u of the corresponding coherence space F $(u \in |F|)$.
- the conclusion $A * B$ with $* \in \{\wp, \otimes, <\}$ of a link $\{\wp, \otimes, <\}$ has the label (t_1, t_2) iff its two premises A and B respectively have the labels t_1 and t_2
- the two conclusions A and $A^\perp$ of an axiom link have the same label — this makes sense because $|A| = |A^\perp|$.

Hence an experiment of Π is completely defined by associating a point of the web $|A| = |A^{\perp}|$ to every axiom $A - A^{\perp}$ of Π.

The experiment is said to **succeed** whenever: the label (t_1, t_2) of a cut $F_j^{\bullet} = E \otimes E^{\perp}$ satisfies $t_1 = t_2$.

Finally, the **semantics** of Π is the set $||\Pi||$ of the tuples $(t_1, ..., t_p)$ such that there exists a successful experiment according to which the conclusions of the proof net, i.e. $F_1, ..., F_p$, are respectively labelled $t_1, ..., t_p$.

Proposition 9. *Let Π be a proof net with conclusion $\vdash F_1, ..., F_p, F_1^{\bullet}, ..., F_n^{\bullet}[\mathfrak{u}]$, let $\mathfrak{v} = \mathfrak{u}|_{F_1,...,F_p}$. Then $||\Pi||$ is a clique of $\prod_{\mathfrak{v}} F_1, ..., F_p$.*

Proof. We assume that two experiments yields two strictly incoherent tuples $(t_1, ..., t_p)$ and $(t'_1, ..., t'_p)$ of $\prod_{\mathfrak{v}} F_1, ..., F_p$, and show that Π contains an æ-circuit. The method consists in extending an æ-path, until we obtain an æ-circuit.

We write $A : \frown$ (resp. $\smile$) whenever the labels of A according to the two experiments are strictly coherent (resp. incoherent) in A.

The æ-path to be constructed has a marking up or down, and throughout its construction fulfils the following requirement:

1. the path always ends on a sub-formula of a conclusions or cut, which may be a conclusion but not a cut.
2. if the mark is up (resp. down) the path ends on a formula $F : \smile$ (resp. $F : \frown$)
3. when the path uses an edge of a par or `before` branching from a premise to the conclusion (resp. from the conclusion to a premise), the two experiments are strictly coherent (resp. strictly incoherent) both in the premise and the conclusion.

We start with the empty path $t_i \smile t'_i : F_i$, and marking up — our assumption makes sure there is such an F_i. Assuming we already have built a path satisfying (1), (2) and (3), we review all its possible endings and markings.

Here are three cases, the others being similar — see [29, pp. 24–29], for a complete description of all the cases:

- *The path ends on the A premise of $A < B$ with marking* down So $A : \frown$ (2). If $B : \smile$ we extend our path using the R-arc from A to B and put the marking up. If $B : \frown$, if the path already used the R-edge $A < B - B$ it used it from $A < B$ to B (3), and using the arc $A \to B$ we have an æ-circuit. Otherwise we extend our æ-path with the R-edge $A - A < B$ and keep the marking down.
- *The path ends in a conclusion, with marking* down. So $A_q : \frown$ (2), and there is an arc from A_q to some $A_r : \smile$. Indeed, if $(a_1, ..., a_p) \smile (a'_1, ..., a'_p) [\prod_{\mathfrak{v}} A_i]$ and $a_k \frown a'_k$ then $\exists A_l > A_k[\mathfrak{u}]\ a_l \smile a'_l[A_l]$. We extend our æ-path using this arc, and put the marking up.
- *The path ends on the premise E of a cut $F_s^{\bullet} = (E \otimes E^{\perp})^{\bullet}$, with marking* down. So $E : \frown$ (2), and, because the experiments succeed, $E^{\perp} : \smile$. We extend the elementary alternating path with the R-edge $E - E^{\perp}$, with marking up — so (1) is still fulfilled.

It is easily seen that (1), (2) and (3) are preserved while extending the path.

Theorem 10. *Every proof net Π has a non-trivial semantics preserved by cut elimination.*

Proof. When Π is a cut-free proof net, any of its experiments succeeds, so the square of the cardinal of $||\Pi||$ is the product of the cardinals of the webs corresponding to leaves. So the semantics of a cut-free proof net is always non-trivial.

Let Π' be a proof net obtained from Π by one of the elementary cut-elimination steps; the *successful* experiments of Π and Π' clearly are in a one-to-one correspondence. Therefore we can speak of a denotational semantics: $||\Pi|| = ||\Pi'||$.

The theorem follows from these two remarks.

In [29] we have established the "strong" converse which follows. It expresses a kind of completeness of coherence semantics w.r.t. ordered proof nets. The proof, which is very simple for the usual multiplicative calculus enriched with the *mix* rule, see [33], is, in the case of ordered proof structures and nets too lengthy to be given here; one should refer to [29, pp. 19–22].

Theorem 11. *There exits a four-token coherence space Z such that, when we interpret each atomic formula by Z, a proof structure Π is a proof net iff $||\Pi||$ is a clique.*

6 Other results on this calculus

We mention here some more results which may be of interest to the reader.

η-rule The η-rule holds for this calculus [28, p. 104]: thus the axiom links may be restricted to atomic formulae.

Extension to full linear logic There is no difficulty in extending this calculus to the modalities "?" and "!" and additive connectives, using proof nets with boxes [10, pp. 43–46] for "!" and "&". The order outside a !-box is the same as inside. The order on both proof nets included in a &-box is asked to be the same, and is the one on the outside of the &-box.

Relation to the usual multiplicative calculus enriched with the mix *rule* This calculus is a faithful extension of proof nets with *mix* [8, pp. 277–278]. Firstly the ordered proof nets without any `before` link exactly are the *mix* proof net. Secondly, in a proof net including some `before` links, each `before` link may be turned into a `par` or `times` link in order to get a correct *mix* proof net. It is trivial for cut-free proof nets, turning `before` links into `par` links, but it is trickier for non-cut-free proof nets, see [28, pp. 104–106].

A modality corresponding to `before` Answering with respect to `before` a question of [11, p. 257], we found, in the category of coherence spaces a self-dual modality which enables contraction w.r.t. `before` on both sides [30, 31]. Its syntax, now under study, is intended for a constructive treatment of classical logic.

7 The problem of finding a sequent calculus

Finding a sequent calculus corresponding to proof nets dealing with n-ary connectives is not easy — e.g. for the n-ary connectives of [7, pp. 196–197] there is no sequent calculus at all.

Because of the De Morgan laws, we can limit ourselves to a calculus of right handed sequent. For this pomset logic, we only have an imperfect solution, the calculus of ordered sequents presented in [28, pp. 111–122]. The axiom is $\vdash A, A^{\perp}[\emptyset]$ — thus proofs start with an empty order on the conclusions and cuts — and the other rules are the following:

$$\frac{\vdash \Gamma, A, B[\mathfrak{u}]}{\vdash \Gamma, A \wp B[\mathfrak{u}']}\wp \qquad \frac{\vdash \Gamma, A, B[\mathfrak{u}]}{\vdash \Gamma, A < B[\mathfrak{u}']}< \qquad \frac{\vdash \Gamma[\mathfrak{u}] \quad \vdash \Delta[\mathfrak{v}]}{\vdash \Gamma, \Delta[\mathfrak{w}]}\,mix \qquad \frac{\vdash \Gamma, A[\mathfrak{u}] \quad \vdash \Delta, B[\mathfrak{v}]}{\vdash \Gamma, \Delta, A \otimes B[\mathfrak{w}]}\otimes$$

In the $\wp$- and $<$-rule the order $\mathfrak{u}'$ is obtained from $\mathfrak{u}$ by identifying A and B, to a formula respectively called $A \wp B$ or $A < B$. But, for applying the $\wp$-rule, A and B are asked to have exactly the same predecessors and successors, while, for applying a $<$-rule, A is asked to be the only predecessor of B and B is asked to be the only successor of A.

In the *mix* and $\otimes$-rule, $\mathfrak{w}$ is any order such that $\mathfrak{w}|_{\bar{\mathfrak{u}}} = \mathfrak{u}$ and $\mathfrak{w}|_{\bar{\mathfrak{v}}} = \mathfrak{v}$ (where $\bar{\mathfrak{u}}$ denotes the domain of the order $\mathfrak{u}$, e.g. Γ, A in the case of the $\otimes$-rule) which satisfies the following property:

$$\forall U, U' \in \bar{\mathfrak{u}}\ \forall V, V' \in \bar{\mathfrak{v}} \quad U{<}V'[\mathfrak{w}] \wedge V{<}U'[\mathfrak{w}] \ \Rightarrow\ U{<}U'[\mathfrak{u}] \vee V{<}V'[\mathfrak{v}]$$

— in the $\otimes$-rule case, one should read $A \otimes B$ as A (resp. B) when it is considered as a formula in $\bar{\mathfrak{u}}$ (resp. $\bar{\mathfrak{v}}$). Furthermore, for the $\otimes$-rule, if $U{<}V[\mathfrak{w}]$ (resp. $U{>}V[\mathfrak{w}]$) with $U \in \bar{\mathfrak{u}}$, $V \in \bar{\mathfrak{v}}$ then $U{<}A[\mathfrak{u}] \vee B{<}V[\mathfrak{v}]$, (resp. $U{>}A[\mathfrak{w}] \vee B{>}V[\mathfrak{w}]$).

The proofs of this sequent calculus translate into ordered proof *nets* and are interpreted by coherence semantics; it is even showed to be the largest "standard" sequent calculus with these properties. Nevertheless, we are unable to prove that every ordered proof net does actually correspond to a proof of this sequent calculus. The techniques of [32] give the hope of a better solution, and we will present an outcome of this study, which enlightens the meaning of `before`, and then explain where the difficulty lies.

Definable connectives: series parallel orders A first interesting step is to look at the definable n-ary connectives, i.e. the ones which behaves like formulae. Let us explain what this means by analogy with the usual multiplicative calculus, whose only connectives are $\wp$ and $\otimes$. Given any proof structure Π with conclusions $A_1, \ldots, A_p, B_1, \ldots, B_n$, there is a formula $\mathcal{F}(A_1, \ldots, A_p)$, namely $A_1 \wp \cdots \wp F_p$ such that Π is a proof net iff Π' is, where Π' is the proof structure with the same axiom links but where the conclusions $A_1, \ldots, A_n$ are replaced by the single conclusion $\mathcal{F}(A_1, \ldots, A_n)$ — Π' obtained from Π by writing the (R&B) formula tree of $\mathcal{F}$ below the conclusions A_i. Moreover, the formula $\mathcal{F}$ is unique up to the commutativity and associativity of $\wp$.

Here, assuming the order on the A_i's is $\mathfrak{u}$, and that no A_i is related to a B_j by the order, we have a similar result, which enlightens the meaning of the connectives. In [28, pp. 106–109], we have shown that:

- The formula $\mathcal{F}$ does not use the `times` connective.
- The possibility of writing a `par` link with A_1 and A_2 as hypotheses exactly corresponds to A_i and A_j having the same predecessors and successors.
- The possibility of writing a `before` link with A_1 and A_2 as hypotheses exactly corresponds to A_i being the only predecessor of A_j and A_j the only successor of A_i
- In both cases the order which has to be taken on $A_1 \wp A_2, A_3, ..., A_p$ or $A_1 < A_2, A_3, ..., A_p$ is obtained by identifying A_1 and A_2.

So the question turns out to be: when does $\mathfrak{u}$ reduce to a single point by these contractions? We have proved that it is the case iff $\mathfrak{u}$ is an N-free order.[4] The sequence of contractions, i.e. the way of writing $\mathcal{F}$ from $\wp$ and $<$, is unique up to the commutativity of $\wp$ and to the associativity of $\wp$ and $<$.

But the N-free orders are known to be series parallel orders [37, pp. 310–311], i.e. to belong to the smallest class of orders closed under disjoint union and ordinal sum, and reading our formula from outermost connectives to innermost ones we obtain the decomposition of $\mathfrak{u}$ as a series parallel order, $\wp$ corresponding to parallel composition, and $<$ to serial composition.

The funny thing about it is that these orders were introduced to model concurrency constraints, which exactly correspond to the inner strategy described by the order. Nevertheless our calculus is able to deal with *any* constraint order, although the constraints expressed by a non-series-parallel order are harder to understand.

Towards a complete sequent calculus Restricting ourselves to definable connectives seems a sensible approach, but the connective `times` does not fit in this setting. This is why we are now thinking of a more general calculus, dealing with a class of relations which include series-parallel orders and series parallel graphs: thus the connective `times`, corresponding to undirected series composition, fits in. This leaves the ordered proof nets almost unchanged, but allows a unary rule for the connective `times`, ruled by the relation.

This approach looks very promising. For the usual multiplicative calculus, it has already brought some results, [32]. But this non-commutative case, which involves directed graphs, is more difficult to handle than the usual one for which numerous proofs are known [10, 7, 8, 32]. Indeed, connectivity questions are much more complex for directed graphs (see e.g. [16]) and, even in [22], there is

[4] The Hasse diagram of the restriction of $\mathfrak{u}$ to four points never is N.

nothing about matchings for directed graphs... Furthermore, we are not simply looking for a bridge, as in the usual calculus, but for an edge cut-set with a given property.

Roughly speaking, we are looking for an inductive definition of the graphs with a perfect matching which possesses no æ-circuit; when the graphs are not directed it is done [32], but otherwise we just obtained some partial results, and we think it is too early to tell more than these hints.

Acknowledgements. Thanks to Jean-Yves Girard for his suggestions and encouragements, to Jacques van de Wiele for helpful advice, and to Maurice Pouzet who pointed to my attention the already existing work on series-parallel orders. Many thanks to Roger Hindley, François Lamarche and the referees for their numerous corrections.

References

1. S. Abramsky. Computational interpretations of linear logic. *Theoretical Computer Science*, 111:3–57, 1993.
2. V. M. Abrusci. Phase semantics and sequent calculus for pure non-commutative classical linear logic. *Journal of Symbolic Logic*, 56(4):1403–1451, December 1991.
3. V. M. Abrusci. Non- commutative proof nets. In [13], pages 271–296, 1995.
4. A. Asperti. A linguistic approach to dead-lock. Technical Report LIENS 91-15, Dép. Maths et Info, Ecole Normale Supérieure, Paris, October 1991.
5. A. Asperti and G. Dore. Yet another correctness criterion for multiplicative linear logic with mix. In A. Nerode and Yu. Matiyasevich, editors, *Logical Fundations of Computer Science*, volume 813 of *Lecture Notes in Computer Science*, pages 34–46, St. Petersburg, July 1994. Springer Verlag.
6. G. Berry. *Modèles complètements adéquats et stables des lambda calculs typés.* Thèse d'état, spécialité sciences mathématiques, Université Paris 7, 1979.
7. V. Danos and L. Regnier. The structure of multiplicatives. *Archive for Mathematical Logic*, 28:181–203, 1989.
8. A. Fleury and C. Retoré. The mix rule. *Mathematical Structures in Computer Science*, 4(2):273–285, June 1994.
9. J.-Y. Girard. Multiplicatives. *Rendiconti del Seminario dell'Universitá é Politecnico Torino*, pages 11–33. October 1986. Special issue on Logic and Computer Science.
10. J.-Y. Girard. Linear logic. *Theoretical Computer Science*, 50(1):1–102, 1987.
11. J.-Y. Girard. A new constructive logic: classical logic. *Mathematical Structures in Computer Science*, 1(3):255–296, November 1991.
12. J.-Y. Girard. Linear logic: its syntax and semantics. In [13], pages 1–42, 1995.
13. J.-Y. Girard, Y. Lafont, and L. Regnier, editors. *Advances in Linear Logic*, volume 222 of London Mathematical Society Lecture Notes. Cambridge University Press, 1995.
14. G. Gonthier, J.-J. Lévy and M. Abadi. The geometry of optimal lambda reduction. In *Principles Of Programming Languages*, pages 15–26, 1992.
15. A. Guglielmi. Concurrency and plan generation in a logic programming language with a sequential operator. In P. van Hentenryck, editor, *International Conference on Logic Programming*, pages 240–254, Genova, 1994. M.I.T. Press.
16. Y. O. Hamidoune. Connexité dans les graphes orientés. *Journal of Combinatorial Theory (B)*, 30:1–11, 1981.

17. J. S. Hodas and D. Miller. Logic programming in a fragment of intuitionistic linear logic. *Information and computation*, pages 327–365, 1994.
18. M. Kanovitch. Linear logic as a logic of computation. *Annals of pure and applied logic*, 67:183–212, 1994.
19. N. Kobayashi and A. Yonezawa. Higher-order concurrent linear logic programming. In *Theory and Practice of Parallel Programming*, volume 907 of *LNCS*, pages 137–166. Springer Verlag, 1995.
20. J. Lambek. The mathematics of sentence structure. *American mathematical monthly*, pages 154–170, 1958.
21. A. Lecomte and C. Retoré. Pomset logic as an alternative categorial grammar. In G. Morrill and R. Oehrle, editors, *Formal Grammar*, pages 181–196, FoLLI.
22. L. Lovàsz and M.D. Plummer. *Matching Theory*, volume 121 of *Mathematics Studies*. North-Holland, 1986. Annals of Discrete Mathematics 29.
23. R. Milner. *Communication and Concurrency*. International series in computer science. Prentice Hall, 1989.
24. G. V. Morrill. *Type Logical Grammar*. Kluwer Academic Publishers, Dordrecht and Hingham, 1994.
25. G. Perrier. Concurrent Programming as Proof Net Construction. Research Report 96-R-132, CRIN-CNRS, Nancy, September 1996.
26. U. S. Reddy. A linear logic model of state. Electronic manuscript, University of Illinois (anonymous FTP from cs.uiuc.edu), October 1993.
27. U. S. Reddy. Global state considered unnecessary: An introduction to object-based semantics. *Journal of Lisp and Symbolic Computation*, 1996. (to appear).
28. C. Retoré. *Réseaux et Séquents Ordonnés*. Thèse de Doctorat, spécialité Mathématiques, Université Paris 7, février 1993.
29. C. Retoré. On the relation between coherence semantics and multiplicative proof nets. Rapport de Recherche 2430, INRIA, décembre 1994.
30. C. Retoré. A self-dual modality for "before" in the category of coherence spaces and in the category of hypercoherences. Rapport de Recherche 2432, INRIA, décembre 1994.
31. C. Retoré. Une modalité autoduale pour le connecteur "précède". In Pierre Ageron, editor, *Catégories, Algèbres, Esquisses et Néo-Esquisses*, Publications du Département de Mathématiques, Université de Caen, pages 11–16, septembre 1994.
32. C. Retoré. Perfect matchings and series-parallel graphs: multiplicative proof nets as R&B-graphs. In J.-Y. Girard, M. Okada, and A. Scedrov, editors, *Linear'96*, volume 3 of *Electronic Notes in Computer Science*. Elsevier, 1996. (available from http://pigeon.elsevier.nl/mcs/tcs/pc/menu.html).
33. C. Retoré. A semantical characterisation of the correctness of a proof net. *Mathematical Structures in Computer Science*. (to appear)
34. I. Schena. Pomset logic and discontinuity in natural language. In C. Retoré, editor, *Logical Aspects of Computational Linguistics*, pages 7–12. CRIN-C.N.R.S. & INRIA-Lorraine, 1996. (to appear in LNCS/LNAI series, Springer-Verlag).
35. R. D. Tennent. *Semantics of programming languages*. Prentice-Hall International, London, 1991.
36. A. S. Troelstra. *Lectures on Linear Logic*, volume 29 of *Center for the Study of Language and Information Lecture Notes*. Cambridge University Press, 1992.
37. J. Valdes, R.E. Tarjan, and E. L. Lawler. The recognition of Series-Parallel digraphs. *SIAM Journal of Computing*, 11(2):298–313, May 1982.