# Natural Deduction and Normalisation for Partially Commutative Linear Logic and Lambek Calculus with Product

Maxime Amblard and Christian Retoré

LaBRI & INRIA-futurs, Université de Bordeaux, France

{amblard,retore}@labri.fr

**Abstract.** This paper provides a natural deduction system for Partially Commutative Intuitionistic Multiplicative Linear Logic (PCIMLL) and establishes its normalisation and subformula property. Such a system involves both commutative and non commutative connectives and deals with context that are series-parallel multisets of formulæ. This calculus is the extension of the one introduced by de Groote presented by the second order for modelling Petri net execution, with a full entropy which allow order to be relaxed into any suborder — as opposed to the Non Commutative Logic of Abrusci and Ruet. Our result also includes, as a special case, the normalisation of natural deduction the Lambek calculus with product, which is unsurprising but yet unproved. Up to now PCIMLL with full entropy had no natural deduction. In particular for linguistic applications, such a syntax is much welcome to construct semantic representations from syntactic analyses.

## 1 Presentation

Non commutative logics arise naturally both in the mathematical perspective and in the modelling of some real world phenomena. Mathematically non commutativity is a natural both from the truth valued semantics viewpoint (phase semantics, based on monoids which can be non commutative) and from a syntactical one (sequent calculus with sequences rather than sets of formulae, proof nets which can have well bracketed axiom links). Non commutativity also appears from real world applications such as concurrency theory, like concurrent execution of Petri net, and in our favourite application, computational linguistic, and this goes back to the fifties and the apparition of the Lambek calculus. We first give a brief presentation of non commutative logics and then stress their interest for concurrency and computational linguistics.

*Non commutative linear logics* Linear logic [6] offered a logical view of the Lambek calculus [9] and non commutative calculi. During many years, the difficulty was to integrate commutative connective and non commutative connectives. A first solution, without a term calculus, was Pomset Logic, now studied with extended sequent calculi callled Calculus of Structures [7].

Another kind of calculus was introduced as a sequent calculus by de Groote in [5], which has to be intuitionistic to work neatly. It consists in a superposition of the Lambek calculus (non commutative) and of Intuitionnistic Linear Logic (commutative). For making a distinction bewteen the two connectives it is necessary that the context includes two different commas mimicking the conjunctions, one being commutative and the other being non commutative. Hence we deal with series-parallel partial orders over multisets of formulae as sequent right hand side. Let us write $(..., ...)$ for the parallel composition and $\langle ...; ... \rangle$ for the non commutative: hence $\langle \{a, b\}; \{c, d\} \rangle$ stands for the finite partial order $a<c, b<c, a<d, b<d$. Of course we would like the two conjunctions to be related. Either the commutative product is stronger than the non commutative one, or the other way round. Surprisingly, the two options work as well, provided one direction is fixed once and for all. This relationship between the two products results from a structural rule modifying the order.

Now a difference should be underlined between the Abrusci-Ruet classical calculus [3] and the intuitionistic one of de Groote's and concern precisely the order rule. The Abrusci-Ruet definitely has an intuitionistic version by limiting it to sequents with a single formula on the right and intuitionistic connectives, namely implications and conjunctions. But there is an important difference with de Groote's calculus: what can be the order rule stigmatising the relation between the two conjunctions? As expected by its inventro an shown in [**?**] the calculus introduced by de Groote can use a general forgetful rule, which replace an order with a smaller one with respect to the inclusion of relations:

$$\frac{\Gamma \text{ ordered by } I \ \vdash C}{\Gamma \text{ ordered by } J \ \vdash C} \, entropy$$

In de Groote's calculus $J$ can be any order such that $J \subset I$ (as set of ordered pairs of formulae in $\Gamma$) while in Ruet $J$ can only be obtained by turning some non commutative commas into commutative ones. This not equivalent to allowing as result any suborder as $J$. Indeed Bechet de Groote and the second author in [4] showed that four rewriting rules are needed to obtained all possible series-parallel partial suborders from some series parallel partial order. Here is a typical derivation that can be performed in de Groote's calculus and not in Ruet's.

$$\frac{(\{a, b\}; \{c, d\}) \vdash (a \otimes b) \odot (c \otimes d)}{\{(\{a, b\}; \{c), d\} \vdash (a \otimes b) \odot (c \otimes d)}$$

Abrusci-Ruet calculus admit a proof net syntax, which can be restricted to the intuitionistic case. Regarding the more flexible de Groote calculus, there neither exists proof nets, nor natural deduction: it only exists a sequent calculus which has been prove to enjoy cut-elimination in [13] (the semantic method of [5] for a simpler entropy can surely be adpated). This is what we propose in this paper, with normalisation. Firstly we thus obtain a calculus which is more convenient, in particular for computational linguistics applciations, because of the Curry-Howard isomorphism. Secondly normal deduction and the needed operations on orders might be a first step towards proof net syntax.

*Motivation for such calculi in concurrency and computational linguistics* Non commutativity in logic is rather natural in a resource consumption perspective. An hypothesis is viewed as a resource that can be use but then it is natural to think of how hypotheses are organised and accessible. As argued by Abrusci [2] and others, linearity is a mandatory condition for non commutativity. Observe that the first non commutative calculus, Lambek calculus [9] which was invented long before linear logic, is a linear calculus, whose relation to oher logical system, in particular intuitionistic has only been understood after the invention of linear logic by Girard [6].

Concurrency, in which the order of the computations or of the resources matters, is of course a natural application. In the framework of proofs as programs, with normalisation as the computational process; it is rather the pomset logic and the calculus of structure which are of some use, because the order applies to cuts that are the computations to be performed [12, 7] But in the framework of proof search as computation, in the logic programming style of Miller, see e.g. [8] or in planning, the calculus studied in this paper with an order on hypotheses is of course important. Indeed process calculi can be encoded in non commutative calculi and this was the main motivation for Ruet's work. The second author also provided a description of the parallel execution of a Petri net in the calculus we are studying – with the reverse entropy increasing order, but it does not change the properties of the calculus. It is a true concurrency approach, where $a||b$ is not reduced to $a; b \oplus b; a$ (where $\oplus$ is the non deterministic choice). An execution according to a series parallel partial order corresponds to a proof in the partially commutative calculus that we study in this paper; in this order based approach of parallel computations any set of minimal transitions can be fired simultaneously. [13]

Our main motivation for such calculi is computational linguistics and grammar formalisms. We are especially fond of logical description of grammar classes as introduced by Lambek because from a parse structure one is able to automatically compute the logical structure of the sentence. This especially true if the Lambek calculus or the partially commutative extensions, like the one we are studying, are given in a natural deduction format. Indeed, the syntactic categories can be turned into semantic categories on two types, individuals ($e$) and truth values ($t$), in such a way that the proof in the Lambek calculus (the syntactic analysis) can be turned into a proof intuitionistic logic, that is a lambda term describing a logical formula in Church's style.

Lambek calculus is definitely too restrictive as a syntactic formalism, in particular it only describes context free languages, and many common syntactic constructs are difficult to model: one would prefer the class of mildly context sensitive formalisms which are assumed to be large enough for natural language constructs, go beyond context-free languages, but admit polynomial parsing algorithms.

This is the reason to use partially commutative calculi. In particular Lecomte and the second author managed to give a logical presentation [10] of Stabler's minimalist grammars [14] in this the de Groote calculus, presented in natural

deduction to obtain semantic representation of the parsed sentences. In parsing as deduction paradigm and for other applications as well it is quite important to have normalisation, unicity of the normal form: indeed the normal form is the structure of the analysed sentence, and normalisation ensures the coherence of the calculus. The algorithm of normalisation, easily extracted from the proof is important as well: one define correct sentences as the ones such that some sequent can be proved, and both the parse structure and the semantic reading are obtained from the normal form.

## 2 Partially Commutative Linear Logic

The sequent calculus for Partially Commutative Linear Logic (PCIMLL) introduced by de Groote in [5] is a super imposition of commutative intuitionistic multiplicative linear logic and the Lambek calculus with product, that is non commutative intuitionistic multiplicative linear logic. Formulae are defined from a set of propositional variables P, by the commutative conjunction ($\otimes$) , the non commutative conjunction ($\odot$), the commutative implication ($\multimap$), the two non commutative implications ($/$ and $\backslash$):

$$\mathrm{L} ::= \mathrm{P} \mid \mathrm{L} \odot \mathrm{L} \mid \mathrm{L} \otimes \mathrm{L} \mid \mathrm{L}/\mathrm{L} \mid \mathrm{L}\backslash\mathrm{L} \mid \mathrm{L} \multimap \mathrm{L}$$

Right hand side, contexts are partially ordered multisets of formulae whose underlying order is series-parallel (sp). Such orders can be depicted by terms over formulae:

$$\mathrm{CTX} ::= \mathrm{L} | \langle \mathrm{CTX}; \mathrm{CTX} \rangle \mid \{\mathrm{CTX}, \mathrm{CTX}\}$$

For instance the context $\langle\langle B; \{A \multimap (B \setminus (D / C), A\}\rangle; C\rangle$ denotes the sp order $Succ(B) = \{A, A \multimap (B \setminus (D / C)\}$, $\mathrm{Succ}(A) = \mathrm{Succ}(A \multimap (B \setminus (D / C)) = \{C\}$ — $Succ(X)$ stands for the immediate successors of $X$ and this fucntion from the domain to the parts of of the domain compeltely determines a finite order.

The term denoting a given sp order is unique up to the associativity of series and parallel composition and to the commutativity of parallel composition and to avoid rewriting associated with associativity or commutativity we consider equal two contexts that are associated with the same sp orpder on the same multiset of formulae. Capital greek letters stand for contexts. An expression like $\Gamma[]$ means a context with one distinguished element $[*$, while an expression like $\Gamma[\Delta]$ means that the special element $[*]$ has been replaced with the context $\Delta$. More details can be found in [4, 13].

- $\Gamma' \sqsubset \Gamma$ means that the domains of the two sp orders, the set of occurrences of formulae, are the same and that $A < B$ in $\Gamma'$ if and only if $A < B$ in $\Gamma$ as well — $A$ and $B$ being /emphoccurrences of formulae. The inclusion $\sqsubset$ of series parallel partial orders can be simulated by term rewriting as shown in [4].

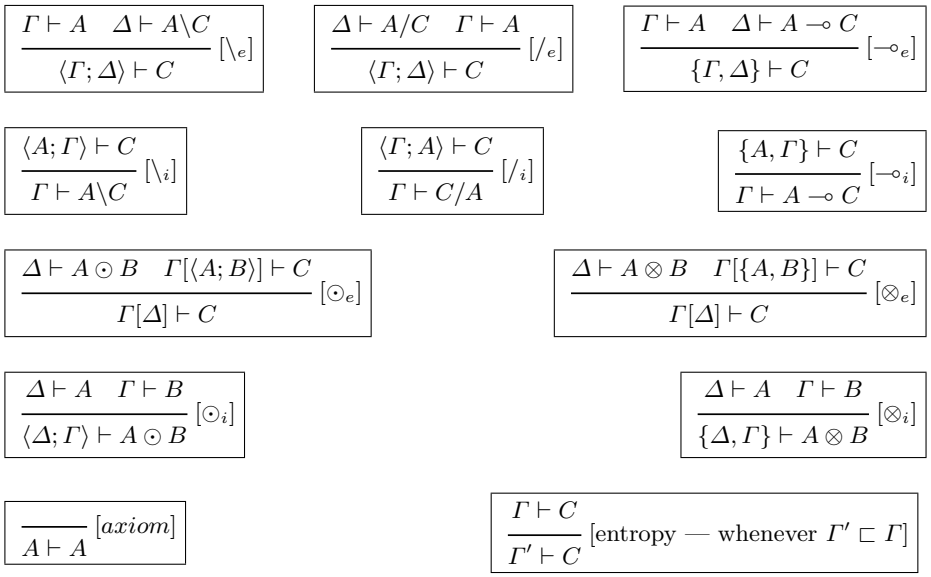


**Fig. 1.** The natural deduction rules for PCIMLL

– Notice that for applying $\otimes_e$ and $\odot_e$ rules, $A$ and $B$ must be equivalent:

$$\forall X \neq A, B \begin{cases} X < A \Leftrightarrow x < B \\ X > A \Leftrightarrow x > B \end{cases}$$

In the $\otimes_e$ rule has $A \not\leq B$ and $A \not\geq B$, while in the $\otimes_e$ rule one has $A < B$.
– Our formulation in a lambda calculus style of the elimination of the multiplicative linear logic conjunction is due Abramsky in [1] — the corresponding term would be $let\ x = (u, v) in \quad t(u, v)$.
– Although we do have normalisation and sub-formula property (next sections) we do not have complicated rules of the kind introduced in [11] for MLL. We assume her rules are motivated by the exponential connectives and other properties.

## 3 Normalisation of PCMLL

**Proposition 1 (Product eliminations can move upwards)** *Let $R$ be an $\otimes_e$ rule (resp. an $\odot_e$ rule) of $\Gamma[\Delta] \vdash C$ between a proof $\delta_0$ of $\Delta \vdash A \otimes B$ and a proof of $\Gamma[\{A, B\}] \vdash C$ (resp. $\Gamma[\langle A; B\rangle] \vdash C$) obtained by a rule $R'$ from a proof $\delta_1$ of some sequent $\Theta[\{A, B\}] \vdash X$ (resp. $\Theta[\langle A; B\rangle] \vdash X$) and possibly, if $R'$ is binary, of another proof $\delta_2$ of some sequent $\Psi \vdash U$ .*

*Then one can obtain a proof of the same sequent $\Gamma[\Delta] \vdash C$ by first applying an $\otimes_e$ rule (resp. an $\odot_e$ rule) between the proof $\delta_0$ of $\Delta \vdash A \otimes B$ and the proof $\delta_1$ of $\Theta[\{A, B\}] \vdash X$ (resp. $\Theta[\langle A; B\rangle] \vdash X$) yielding a proof of $\Theta[\Delta] \vdash X$ and then applying $R'$ to this proof and possibly the proof $\delta_2$ of the sequent $\Psi \vdash U$.*

*Proof.* This is a case study, according to the rules above the product elimination. Observe that the product elimination rule only move upwards when both cancelled hypotheses are in the same premise and when their respective places in the order match the product elimination requirements. Here is one case.

$$\dfrac{\Delta \vdash A \otimes B \qquad \dfrac{\Gamma[\{A,B\}] \vdash D \quad \Phi \vdash D \backslash C}{\langle \Gamma[\{A,B\}]; \Phi \rangle \vdash C}\,[\backslash_e]}{\langle \Gamma[\Delta]; \Phi \rangle \vdash C}\,[\otimes_e] \Rightarrow \dfrac{\dfrac{\Delta \vdash A \otimes B \quad \Gamma[\{A,B\}] \vdash D}{\Gamma[\Delta] \vdash D}\,[\otimes_e] \qquad \Phi \vdash D \backslash C}{\langle \Gamma[\Delta]; \Phi \rangle \vdash C}\,[\backslash_e]$$

We write $S_j$ for the occurrence of sequent inside a proof, $|S_j|$ for the corresponding sequent, and $|S_j|^r$ for the formula on its right hand side.

In a proof $\delta$ , $B(S_0)$ the **principal branch** issued from an occurrence $S_0$ of a sequent $|S_0|$ is the smallest upwards path containing $S_0$ and closed under the following operations:

1. if $S \in B(S_0)$ is obtained by a unary rule $R$ from an occurrence $S'$ of a sequent, then $S' \in B(S_0)$ as well;
2. if $S \in B(S_0)$ is obtained by a product elimination rule $\odot_e$ (resp. $\otimes_e$) as depicted in figure 1 then the **connective-marked premise** which is $S'$ with $|S'| = \Gamma[\langle A, B \rangle] \vdash C$ (resp. $|S'| = \Gamma[\{A, B\}] \vdash C$) is in $B(S_0)$ as well.
3. if $S \in B(S_0)$ is obtained by an implication elimination rule $\backslash_e$ (resp. $/_e$, $\multimap_e$) as depicted in figure 1 then the **connective-marked premise** $S'$ with $|S'| = \Delta \vdash A \setminus C$ (resp. $|S'| = \Delta \vdash A \,/\, C$, $|S'| = \Delta \vdash A \multimap C$) is in $B(S_0)$ as well.

Any initial subpath of a principal branch $B(S_0)$ of length $n$ from $S_0$ to $S_n$ with $|S_0|^r = |S_n|^r$, such that $|S_0|$ is the connective-marked premise of some elimination rule $R_e$ and $S_n$ is the conclusion of some introduction rule $R_i$ is called an **n-extended redex**. Observe that the rules $R_i$ and $R_e$ necessarily introduce and eliminate the same connective and are said to be **conjoined**. A *0-extended redex* is called a **redex**, and there are seven redexes in PCIMLL see figure 3. A **normal proof** is defined as a proof without any $k$-extended-redex, for every $k$. Observe that there is at most one $k$-extented redex in a principal branch, because it's an initial path of $S(B_0)$ and the rule above $S_n$ is an introduction.

Given an implication elimination rule $R$ (resp. a product elimination rule $R'$), with connective-marked premise $S_0$, the integer $e(R)$ (resp. $g(R')$) is $k$ if there is some (hence one) $k$-extended redex in $B(S_0)$ called the $k$-extended redex above $R$ (resp. above $R'$) and 0 otherwise. Given a proof $\delta$ let $PER(\delta)$ and $IER(\delta)$ be respectively the of occurrences of product elimination rule in $\delta$ and of implication elimination rule in $\delta$. We define $e(\delta)$ as $\min_{R \in PER(\delta)} e(R)$ and $g(\delta)$ as $\min_{R \in IER(\delta)} g(R)$ and $r(\delta)$ as the number of rules in $\delta$. The measure of $\delta$ denoted by $|\delta|$ is the triple $(r(\delta), e(\delta), g(\delta)$. A proof is normal if and only if its measure is $r(\delta, 0, 0)$.

**Proposition 2** *A $k$-extended-redex $S_0 \cdots S_k$ containing an implication elimination contains another $k'$-extended-redex with $k' < k$.*

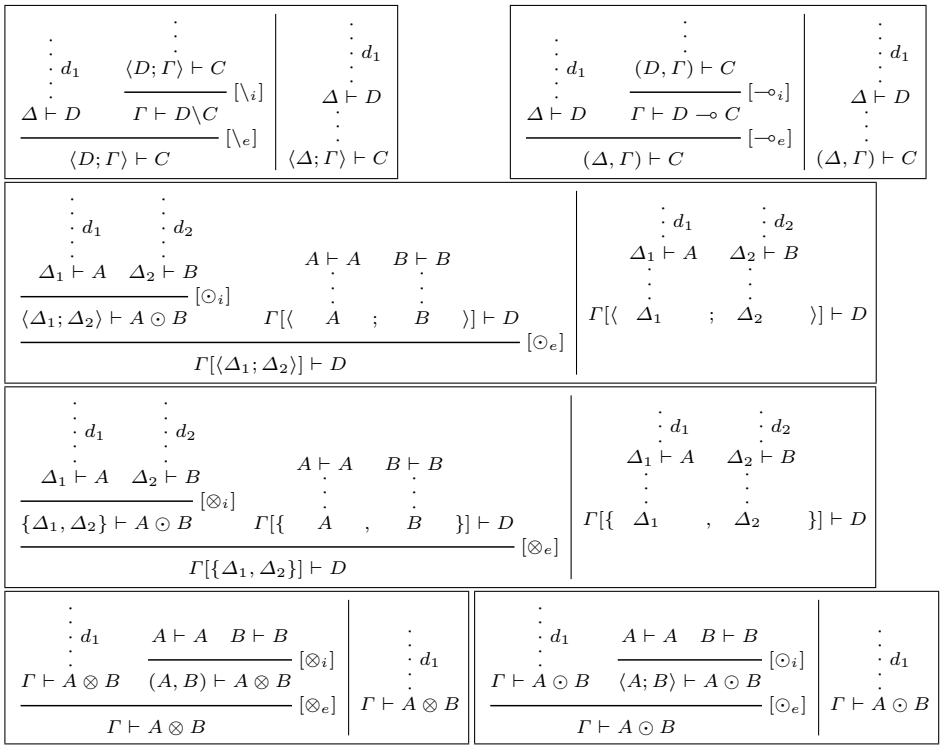

**Fig. 2.** The seven redexes of PCIMLL

*Hence, a $k$-extended-redex which does not contain any $k'$-extended-redex with $k' < k$ and in particular the extended redex associated with $R$ such that $e(R) = e(\pi)$ or $g(R) = g(\pi)$, only contains entropy rules and product elimination rules.*

**Proposition 3** *Product elimination and entropy can move below implication elimination rules.*

**Theorem 4 (Normalisation)** *Every proof $\pi$ in PCMLL has a normal form.*

*Proof.* We proceed by induction on the measure of the proof. We assume that for all proof $\pi'$ of size $|\pi'| < \langle n, e, g\rangle$ has a normal form and let us show that any proof $\pi$ with $|\pi| = \langle n, e, g\rangle$ can be turned into a normal proof. As every redex turns a proof into a proof with less rules, we can assume that $\pi$ has no redex.

If $e(\pi) \neq 0$, then there exists an implication-elimination rule $S$ with a $e(\pi)$-extended redex. Since it's a minimal redex, this extended redex only contain product elimination rules and entropy rules by proposition 2. Consequently, by proposition 3 we can exchange $S$ and the rule immediately above $S$ because it's an entropy rule or a product elimination rule. The proof that we thus obtain

$\pi'$ satifies $n(\pi') = n(\pi)$ and $e(\pi') < e(\pi)$ — in $\pi'$ $e(S') = e(S) - 1$. Hence, by induction $\pi'$ as a normal form and $\pi$ as well.

If $e(\pi) = 0$ and $g(\pi) \neq 0$ then there exists a product-elimination rule $S$ with a $g(\pi)$-extended redex. Since it's a minimal redex, this extended redex only contain product elimination rules and entropy rules by proposition 2. Consequently, we can exchange $S$ and the rule immediately above $S$ because it's an entropy rule or a product elimination rule. The proof that we thus obtain $\pi'$ satifies $n(\pi') = n(\pi)$ and still does not have any implication extended redex by checking that the modification does not create new principal branch — in $n(\pi') = n(\pi)$ $g(\pi') = g(\pi) = 0$. Hence, by induction $\pi'$ as a normal form and $\pi$ as well.

If $e(\pi) = g(\pi) = 0$, $\pi$ is already normal.

**Theorem 5 (Subformulae property)** *The sub-formula property holds for PCMLL: in a normal proof $\pi$ of a sequent $\Gamma \vdash C$, every formulæ of every sequent of $\pi$ is a sub-formula of a formula of $\Gamma$ or of the formula $C$.*

*Proof.* We proceed by induction on the height of a normal proof showing the result and that whenever the last rule is some implication elimination, then $C$ is a subformula of some formula of $\Gamma$. The entropy rule is ignored, both as a last rule and for measuring the height of a proof. Axioms enjoy this property. Assuming the property holds for smaller proofs, and that our normal proofs ends with an introduction rule the result is clear.

Assume the last rule is $\backslash_e$ — the other implication rules are handled *mutatis mutandis*. We successively inspect all possible rules above the connective-marked premise $\Gamma \vdash A \setminus B$:

- *axiom* in this case the formula also appear in $\Gamma$ as wished.
- $\backslash_i$ impossible there would exist a redex while the proof is in normal form.
- all other introductions can not lead to $\Gamma \vdash A \setminus B$ because of the $\setminus$ in $A \setminus B$.
- $\backslash_e$, $/_e$ and $\multimap_e$: using the induction hypothesis.
- $\otimes_e$ and $\odot_e$ which preserve the target of the sequent, namely $A \backslash B$ we consider the principal branch $B(\Gamma \vdash A \setminus B)$. After a sequence of $\otimes_e$ and $\odot_e$ rules, yielding to a sequent $\Delta \vdash A \setminus B$ we either find an axiom $A \setminus B \vdash A \setminus B$ or any of the afore mentioned rules. In the first case $A \setminus B$ is itself a formula of $\Gamma$ and in the second one the induction hypothesis justifies the result.

Assume the last rule is $\otimes_e$ with the connective marked sequent being $\Gamma \vdash A \otimes B$, the other begin $\Theta[\{A, B\}] \vdash C$ and the conclusion $\Theta[\Gamma] \vdash C$. The only point to be checked is that $A \otimes B$, which does not appear in the conclusion sequent, is a subformula of a formula of $\Gamma$. After a sequence of $\otimes_e$ and $\odot_e$ rules, yielding to a sequent $\Delta \vdash A \otimes B$ we either find an axiom $A \otimes B \vdash A \otimes B$ or a proper rule $R$. In the first case $A \otimes B$ is itself a formula of $\Gamma$. In the second case $R$ cannot be any implication introduction but it can neither be a product introduction since this initial part of the principal branch would be an extented redex. Hence it has to be an arrow elimination rule and thus $A \otimes B$ is a subformula of $\Delta$, hence of $\Gamma$.

Observe that we have such results for a calculus with standard rules, as opposed to [11].

## 4 Normalisation for Lambek calculus with product

Lambek caculus with product is PCIMLL restricted to $\backslash$, / and $\odot$. Context only use $\langle ; \rangle$; as they are defined up to associativity, context are sequences of formulæ and the *entropy* rule can safely be left out.

Theorem 4 implies that proofs in the $L_{\odot}$ have a normal form enjoying the subformula property. In this particular case proofs have a unique normal proof which is reached when the algorithm underlying the normalisation proof is concluded by moving as high as possible the $\odot_e$ as explained in proposition 1. The uniqueness of the normal proof is easily established from local confluence of the redex.

## 5 Conclusion and future work

Motivated by concurrency and computational linguistics we have been defining PCIMLL in natural deduction and proved normalisation. For Lambek calculus with product, a subcalculus of PCIMLL, we also characterized the unique normal proof. Regarding the full calculus, we characterized the normal proofs up to the permutation of commutative product elimination rules among them.

Next we'll look forward a proof net syntax for PCIMLL, which also allows to easily compute lambda terms. Despite the existence of proof nets for MLL andfor the Lambek calculus (of which PCMLL is the superimposition), and for intuitionistic NL of Abrusci and Ruet, because of the more flexible entropy rule that we are using, there is not yet any proof net calculus for PCNLL. This work in aprticular on normal proofs which split the entropy rules into canonical steps according to the rules used above and below can be viewed as a first step in this direction.

With respect to computational linguistic application, our results tighten the already explored connection between minimalist grammars and linear calculi in a natural deduction format, which is responsible for the automatic computing of semantic recipes. It also opens the possibility to have concurrent abstract machines for minimalist grammars, since their coding uses the same formulæ as Petri nets coding. The linear deduction also enables a use of other semantics like ludics.

**Acknowledgement** We would like to apologize to the reviewers for the mistakes and the poor style of the version that was sent to them.

## References

[1] Samson Abramsky. Computational interpretations of linear logic. *Theoretical Computer Science*, 111:3–57, 1993.

[2] V. Michele Abrusci. Phase semantics and sequent calculus for pure noncommutative classical linear propositional logic. *The Journal of Symbolic Logic*, 56(4):1403–1451, December 1991.

[3] V. Michele Abrusci and Paul Ruet. Non-commutative logic I: The multiplicative fragment. *Annals of pure and applied logic*, 101(1):29–64, 1999.

[4] Denis Bechet, Philippe de Groote, and Christian Retoré. A complete axiomatisation of the inclusion of series-parallel partial orders. In H. Comon, editor, *Rewriting Techniques and Applications, RTA'97*, volume 1232 of *LNCS*, pages 230–240. Springer Verlag, 1997.

[5] Philippe de Groote. Partially commutative linear logic: sequent calculus and phase semantics. In Vito Michele Abrusci and Claudia Casadio, editors, *Third Roma Workshop: Proofs and Linguistics Categories – Applications of Logic to the analysis and implementation of Natural Language*, pages 199–208. Bologna:CLUEB, 1996.

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

[7] Alession Guglielmi. A system of interaction and structure. *ACM Transaction on Computational Logic*, 8(1):1–64, January 2007.

[8] J.. S. Hodas and D. Miller. Logic programming in a fragment of intuitionistic linear logic. *Information and computation*, pages 327–365, 1994.

[9] Joachim Lambek. The mathematics of sentence structure. *American mathematical monthly*, pages 154–170, 1958.

[10] Alain Lecomte and Christian Retoré. Extending Lambek grammars: a logical account of minimalist grammars. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics, ACL 2001*, pages 354–361, Toulouse, July 2001. ACL.

[11] Sara Negri. A normalizing system of natural deduction for intuitionistic linear logic. *Archive for Mathematical Logic*, 2002.

[12] Christian Retoré. Pomset logic: a non-commutative extension of classical linear logic. In Philippe de Groote and James Roger Hindley, editors, *Typed Lambda Calculus and Applications, TLCA'97*, volume 1210 of *LNCS*, pages 300–318, 1997.

[13] Christian Retoré. A description of the non-sequential execution of petri nets in partially commutative linear logic. In Jan van Eijck, Vincent van Oostrom, and Albert Visser, editors, *Logic Colloquium 99*, Lecture Notes in Logic, pages 152–181. ASL and A. K. Peters, 2004.

[14] Edward Stabler. Derivational minimalism. *Logical Aspect of Computational Linguistic*, 1328, 1997.