An Introduction to Proof Nets

Olivier . Laurent @ens-lyon.fr

October 18, 2013

Abstract

We give some basic results of the theory of proof nets for multiplicative linear logic and multiplicative exponential linear logic with mix. The relation between proof nets and the lambda-calculus is precisely described.

1 Multiplicative Proof Nets

1.1 Multiplicative Linear Logic

We assume given a denumerable set of atoms X, Y, ... The formulas of multiplicative linear logic (MLL) are defined as:

$$A, B ::= X \mid X^{\perp} \mid A \otimes B \mid A \parr B$$

The connective $(.)^{\perp}$ is extended into an involution on all formulas by:

$$(X^{\perp})^{\perp} = X \qquad (A \otimes B)^{\perp} = A^{\perp} \parr B^{\perp} \qquad (A \parr B)^{\perp} = A^{\perp} \otimes B^{\perp}$$

For example:

$$\begin{aligned}
(X^{\perp} \otimes (X \parr Y^{\perp}))^{\perp\perp} &= (X^{\perp\perp} \parr (X \parr Y^{\perp})^{\perp})^{\perp} \\
&= (X \parr (X^{\perp} \otimes Y^{\perp\perp}))^{\perp} \\
&= (X \parr (X^{\perp} \otimes Y))^{\perp} \\
&= X^{\perp} \otimes (X^{\perp} \otimes Y)^{\perp} \\
&= X^{\perp} \otimes (X^{\perp\perp} \parr Y^{\perp}) \\
&= X^{\perp} \otimes (X \parr Y^{\perp})
\end{aligned}$$

Sequents are sequences of formulas denoted $\vdash \Gamma$. The sequent calculus rules of MLL are:

$$\frac{}{\vdash A^{\perp}, A}\ ax \qquad \frac{\vdash \Gamma, A \qquad \vdash A^{\perp}, \Delta}{\vdash \Gamma, \Delta}\ cut \qquad \frac{\vdash \Gamma, A \qquad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B}\ \otimes \qquad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}\ \parr$$

The formal definition of the system requires to add the *exchange* rule:

$$\frac{\vdash \Gamma}{\vdash \rho(\Gamma)}\ ex(\rho)$$

where ρ is a permutation. However the precise use of this rule and the careful usage of the order of formulas in sequents add a lot of useless technicalities to the results we want to present here. We will thus do in the following as if sequents were multisets of formulas even if this is not strictly speaking a valid way of defining them.

1.2 Forgetting Sequential Structure

$$
\begin{prooftree}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$\vdash A, A^\perp$}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$\vdash B, B^\perp$}
\RightLabel{$\otimes$}
\BinaryInfC{$\vdash A \otimes B, A^\perp, B^\perp$}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$\vdash C, C^\perp$}
\RightLabel{$\otimes$}
\BinaryInfC{$\vdash (A \otimes B) \otimes C, A^\perp, B^\perp, C^\perp$}
\RightLabel{$\parr$}
\UnaryInfC{$\vdash (A \otimes B) \otimes C, A^\perp \parr B^\perp, C^\perp$}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$\vdash A, A^\perp$}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$\vdash B, B^\perp$}
\RightLabel{$\otimes$}
\BinaryInfC{$\vdash A \otimes B, A^\perp, B^\perp$}
\RightLabel{cut}
\BinaryInfC{$\vdash (A \otimes B) \otimes C, A^\perp, B^\perp, C^\perp$}
\RightLabel{$\parr$}
\UnaryInfC{$\vdash (A \otimes B) \otimes C, A^\perp \parr B^\perp, C^\perp$}
\end{prooftree}
$$

$$
\begin{prooftree}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$\vdash \textcolor{red}{A, A^\perp}$}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$\vdash \textcolor{red}{B, B^\perp}$}
\RightLabel{$\otimes$}
\BinaryInfC{$\vdash \textcolor{red}{A \otimes B}, A^\perp, B^\perp$}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$\vdash \textcolor{red}{C, C^\perp}$}
\RightLabel{$\otimes$}
\BinaryInfC{$\vdash \textcolor{red}{(A \otimes B) \otimes C}, A^\perp, B^\perp, C^\perp$}
\RightLabel{$\parr$}
\UnaryInfC{$\vdash (A \otimes B) \otimes C, \textcolor{red}{A^\perp \parr B^\perp}, C^\perp$}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$\vdash \textcolor{red}{A, A^\perp}$}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$\vdash \textcolor{red}{B, B^\perp}$}
\RightLabel{$\otimes$}
\BinaryInfC{$\vdash \textcolor{red}{A \otimes B}, A^\perp, B^\perp$}
\RightLabel{cut}
\BinaryInfC{$\vdash (A \otimes B) \otimes C, A^\perp, B^\perp, C^\perp$}
\RightLabel{$\parr$}
\UnaryInfC{$\vdash (A \otimes B) \otimes C, \textcolor{red}{A^\perp \parr B^\perp}, C^\perp$}
\end{prooftree}
$$

1.3 Proof Structures

Definitions and abstract properties we are going to use about graphs can be found in Appendix A.

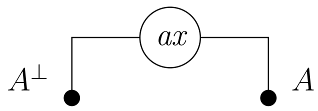
A *proof structure* is a directed multigraph e-labelled with multiplicative formulas (the label of an edge is called its *type*) and n-labelled with $\{ax, cut, \otimes, \parr, \mathsf{c}\}$ (the incoming edges of a node are called its *premisses*, the outgoing edges are its *conclusions*) such that:

- Each node labelled *ax* has exactly two conclusions (and no premisse) which are labelled A and $A^\perp$ for some A.
- Each node labelled *cut* has exactly two premisses (and no conclusion) which are labelled A and $A^\perp$ for some A. It is called a *cut*.
- Each node labelled $\otimes$ has exactly two premisses and one conclusion. These two premisses are ordered. The smallest one is called the *left* premisse of the node, the biggest one is called the *right* premisse. The left premisse is labelled A, the right premisse is labelled B and the conclusion is labelled $A \otimes B$, for some A and B.
- Each node labelled $\parr$ has exactly two ordered premisses (as for $\otimes$ nodes) and one conclusion. The left premisse is labelled A, the right premisse is labelled B and the conclusion is labelled $A \parr B$, for some A and B.
- Each node labelled c has exactly one premisse (and no conclusion). Such a premisse of c node is called a *conclusion* of the proof structure. They are simply represented as bullets in pictures.

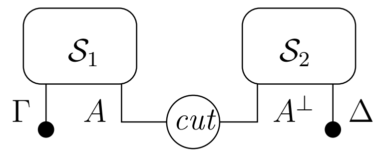
Nodes with all their conclusions connected to c nodes are called *terminal*. By definition a non-empty proof structure must contain at least one *ax* node (they are the only nodes with no premisse).

Proofs of the sequent calculus MLL can be translated into multiplicative proof structures:

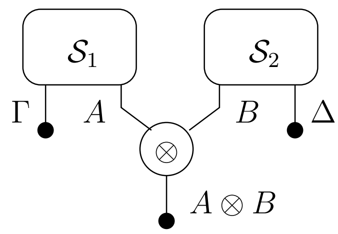
- An (ax) rule $\dfrac{}{\vdash A^\perp, A}\,ax$ is translated into an *ax* node with conclusions labelled $A^\perp$ and A which have c nodes as targets.



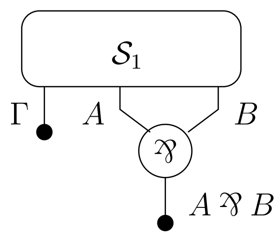
- If π_1 is translated into $\mathcal{S}_1$ and π_2 is translated into $\mathcal{S}_2$, then to the proof $\dfrac{\begin{array}{c}\pi_1\\ \vdash \Gamma, A\end{array} \qquad \begin{array}{c}\pi_2\\ \vdash A^\perp, \Delta\end{array}}{\vdash \Gamma, \Delta}\,cut$ we associate the proof structure $\mathcal{S}$ obtained from $\mathcal{S}_1$ and $\mathcal{S}_2$ by removing the c nodes with premisses e_1 labelled A and e_2 labelled $A^\perp$, and by introducing a new *cut* node with premisses e_1 and e_2.



- If π_1 is translated into $\mathcal{S}_1$ and π_2 is translated into $\mathcal{S}_2$, then to the proof $\dfrac{\begin{array}{c}\pi_1\\ \vdash \Gamma, A\end{array} \qquad \begin{array}{c}\pi_2\\ \vdash \Delta, B\end{array}}{\vdash \Gamma, \Delta, A \otimes B}\,\otimes$ we associate the proof structure $\mathcal{S}$ obtained from $\mathcal{S}_1$ and $\mathcal{S}_2$ by removing the c nodes with premisses e_1 labelled A and e_2 labelled B, and by introducing a new $\otimes$ node with premisses e_1 and e_2 and with conclusion a new edge labelled $A \otimes B$ which is itself the premisse of a new c node.



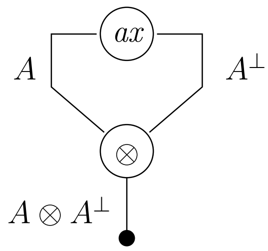
- If π_1 is translated into $\mathcal{S}_1$, then to the proof $\dfrac{\begin{array}{c}\pi_1\\ \vdash \Gamma, A, B\end{array}}{\vdash \Gamma, A \parr B}\,\parr$ we associate the proof structure $\mathcal{S}$ obtained from $\mathcal{S}_1$ by removing the c nodes with premisses e_1 labelled A and e_2 labelled B, and by introducing a new $\parr$ node with premisses e_1 and e_2 and with conclusion a new edge labelled $A \parr B$ which is itself the premisse of a new c node.

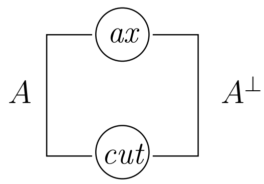


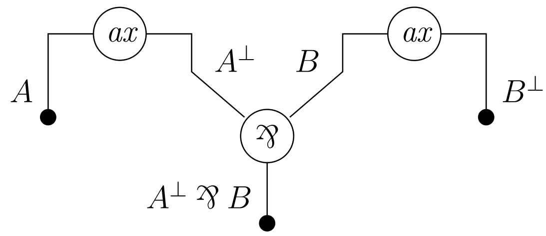
1.4 Correctness

Not any proof structure represents a proof in the sequent calculus MLL. This leads to the study of *correctness criteria* to try to delineate a sub-set of “valid” proof structures.

Here are a few examples of proof structures which do not correspond to any proof of MLL:





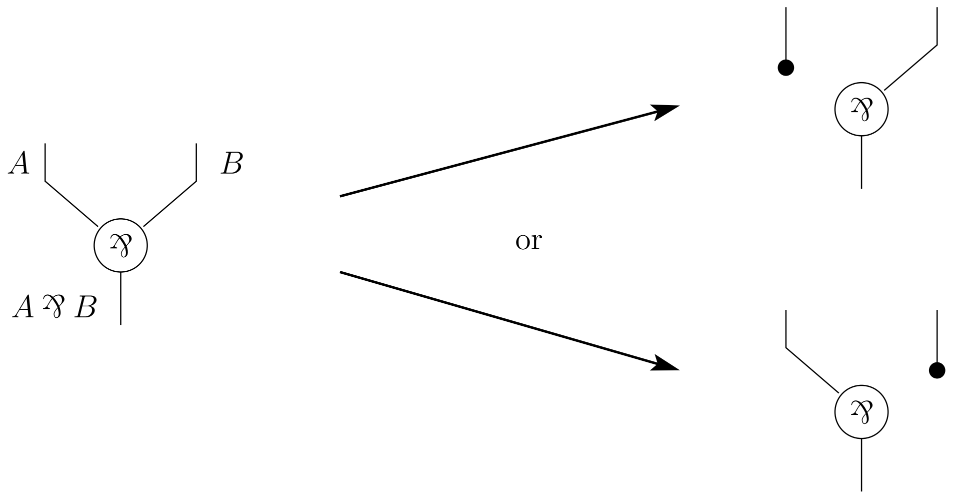


$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash A, A^\perp}\,ax \qquad \dfrac{}{\vdash B, B^\perp}\,ax}{\vdash A, A^\perp \parr B, B^\perp}\,\parr \qquad \dfrac{}{\vdash C, C^\perp}\,ax}{\vdash A, A^\perp \parr B, B^\perp \parr C, C^\perp}\,\parr}{\vdash A, (A^\perp \parr B) \otimes (B^\perp \parr C), C^\perp}\,\otimes$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash A, A^\perp}\,ax \qquad \dfrac{}{\vdash B, B^\perp}\,ax}{\vdash A \parr B, A^\perp, B^\perp}\,\parr \qquad \dfrac{}{\vdash C, C^\perp}\,ax}{\vdash C \otimes (A \parr B), C^\perp, B^\perp, A^\perp}\,\otimes}{\vdash C \otimes (A \parr B), C^\perp \parr B^\perp, A^\perp}\,\parr \qquad \dfrac{}{\vdash D, D^\perp}\,ax}{\vdash C \otimes (A \parr B), C^\perp \parr B^\perp, D \parr A^\perp, D^\perp}\,\parr \qquad \dfrac{}{\vdash E, E^\perp}\,ax}{\vdash C \otimes (A \parr B), C^\perp \parr B^\perp, E^\perp, D^\perp, (D \parr A^\perp) \otimes E}\,\otimes}{\vdash C \otimes (A \parr B), C^\perp \parr B^\perp, E^\perp \parr D^\perp, (D \parr A^\perp) \otimes E}\,\parr}{\vdash C \otimes (A \parr B), (C^\perp \parr B^\perp) \otimes (E^\perp \parr D^\perp), (D \parr A^\perp) \otimes E}\,\otimes$$

1.4.1 Correctness Criteria

Given a proof structure $\mathcal{S}$, let $\mathcal{P}$ be the set of its $\parr$ nodes, a *switching* of $\mathcal{S}$ is a function $\varphi : \mathcal{P} \to \{\text{left}, \text{right}\}$. The *switching graph* $\mathcal{S}^\varphi$ associated with φ is the labelled directed multigraph obtained from $\mathcal{S}$ by modifying the target of the $\varphi(P)$ premisse of each $\parr$ node P into a new c node.



A proof structure with p $\parr$ nodes induces 2^p switchings and thus 2^p switching graphs. A switching graph is not a proof structure in general since its $\parr$ nodes have only one premisse.

A *connected component* of a switching graph is a connected component of its underlying (undirected) multigraph.

Acyclicity. A multiplicative proof structure is *acyclic* if its switching graphs do not contain any undirected cycle.

An acyclic multiplicative proof structure is called a *multiplicative proof net.*

Connectedness.

Lemma 1.1 (Connected Components)
Let $\mathcal{S}$ be a proof structure, the number of connected components of its (undirected) acyclic switching graphs are the same.

Proof: If N is the number of nodes of $\mathcal{S}$, P its number of $\parr$ nodes and E its number of edges, any switching graph of $\mathcal{S}$ has $N + P$ nodes and E lines. By Lemma A.1, any such acyclic multigraph has $N + P - E$ connected components. □

A multiplicative proof net is *connected* if its switching graphs have exactly one connected component.

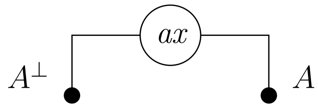
Thanks to the previous lemma, this is equivalent to checking that one switching graph is connected.

1.4.2 Soundness

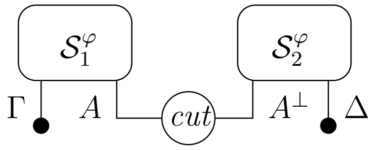
Proposition 1.1 (Soundness of Correctness)
The translation of a sequent calculus proof of MLL *is a connected multiplicative proof net.*

Proof: By induction on the structure of the MLL proof π. Let $\mathcal{S}$ be the proof structure associated with π, and we also need to consider two sub-proofs π_1 and π_2 of π with associated proof structures $\mathcal{S}_1$ and $\mathcal{S}_2$.

- The proof structure below has a unique switching graph which has no undirected cycle and a unique connected component.

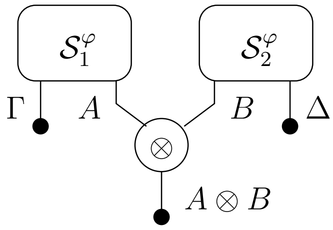


- If π is obtained from π_1 and π_2 with a (cut) rule, a switching graph $\mathcal{S}^\varphi$ of $\mathcal{S}$ is obtained by connecting through a *cut* node the switching graph $\mathcal{S}_1^\varphi$ of $\mathcal{S}_1$ and the switching graph $\mathcal{S}_2^\varphi$ of $\mathcal{S}_2$.



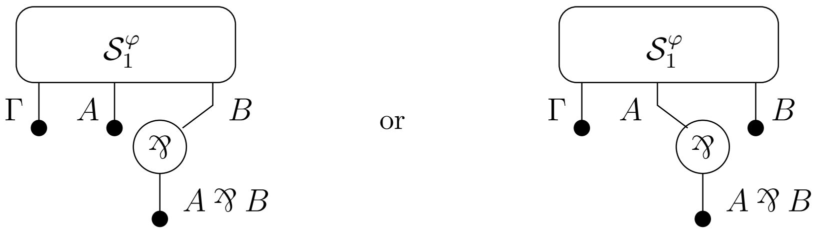
  We can deduce that no switching graph of $\mathcal{S}$ contains an undirected cycle and they all have a unique connected component.

- If π is obtained from π_1 and π_2 with a $(\otimes)$ rule, a switching graph $\mathcal{S}^\varphi$ of $\mathcal{S}$ is obtained by connecting through a $\otimes$ node the switching graph $\mathcal{S}_1^\varphi$ of $\mathcal{S}_1$ and the switching graph $\mathcal{S}_1^\varphi$ of $\mathcal{S}_2$.



  We can deduce that no switching graph of $\mathcal{S}$ contains an undirected cycle and they all have a unique connected component.

- If π is obtained from π_1 with a $(\parr)$ rule, a switching graph $\mathcal{S}^\varphi$ of $\mathcal{S}$ is obtained by putting a $\parr$ node connected to a c node instead of a c node of $\mathcal{S}_1^\varphi$.



  We can deduce that no switching graph of $\mathcal{S}$ contains an undirected cycle and they all have a unique connected component. □

1.4.3 Sequentialization

We want to associate an MLL proof with each connected proof net. This is called the sequentialization process, for it requires to turn the graph structure of proof nets into the more sequential tree structure of sequent calculus proofs.

In order to help the reuse of some of the results, we consider here a simple generalization of proof structures where *ax* nodes are replaced with *hyp* nodes:

- Each node labelled *hyp* has at least one conclusion (and no premisse). If the conclusions of an *hyp* node are labelled $A_1, \ldots, A_n$ ($n > 0$), the sequent $\vdash A_1, \ldots, A_n$ must be provable in MLL.

ax nodes are clearly a particular case of these new *hyp* nodes since $\vdash A, A^\perp$ is provable for any A in MLL by means of an (ax) rule. For the purpose of sequentialization, we associate any proof of $\vdash A_1, \ldots, A_n$ with an *hyp* node with conclusions labelled $A_1, \ldots, A_n$.

A *switching path* of a proof structure is a simple undirected path of one of its switching graphs. A switching path in a switching graph $\mathcal{S}^\varphi$ can be seen as a an undirected path of the proof structure $\mathcal{S}$ itself (this might only require to "rename" c nodes of $\mathcal{S}^\varphi$, which are not in $\mathcal{S}$, into the corresponding $⅋$ nodes). A *strong switching path* is a switching path whose first edge is not the premisse of a $⅋$ node.

Lemma 1.2 (Concatenation of Switching Paths)
If γ is a switching path, γ' is a strong switching path with $\mathsf{t}(\gamma) = \mathsf{s}(\gamma')$, and if γ and γ' are disjoint (no common edge) then their concatenation $\gamma\gamma'$ is a switching path.

If γ is strong then $\gamma\gamma'$ as well.

Proof: By hypotheses, the path $\gamma\gamma'$ is a simple undirected path. If it is not a switching path, then there exists a $⅋$ node P with premisses e_1 and e_2 and with conclusion e_0 such that:

- either $\gamma\gamma'$ contains $(e_1, +)(e_2, -)$ or $(e_2, +)(e_1, -)$ but this is not possible since it cannot occur inside γ or γ' by hypotheses and it cannot occur at the source of γ' since γ' is strong,
- or $\gamma\gamma'$ contains both $(e_1, +)(e_0, +)$ or $(e_0, -)(e_1, -)$, and $(e_2, +)(e_0, +)$ or $(e_0, -)(e_2, -)$, but this is not possible otherwise the path would contain twice e_0 while it is a simple path.

If γ is strong, then either it is empty and $\gamma\gamma' = \gamma'$ is strong or γ is not empty and the first edge of $\gamma\gamma'$ is the first edge of γ thus it is strong. $\square$

Lemma 1.3 (Cyclic Strong Switching Paths)
In a proof net, there is no strong switching path whose target is its source.

Proof: Let $\mathcal{R}^\varphi$ be a switching graph containing a strong switching path γ with target equal to its source (possibly up to identifying c nodes not in $\mathcal{R}$ with their corresponding $⅋$ node). If the source n of γ is not a $⅋$ node, γ defines a cycle in $\mathcal{R}^\varphi$, a contradiction. Otherwise, let φ' be obtained from φ by connecting the last edge of γ with n (note we may have $\varphi' = \varphi$), γ is a cycle in $\mathcal{R}^{\varphi'}$, a contradiction. $\square$

A terminal $\otimes$ node T of a proof structure is called *splitting* if by removing it (as well as its conclusion edge and the c node it is connected to) and by adding two new c nodes as targets of the premisses of T, one obtains two disjoint proof structures.

Lemma 1.4 (Blocking $⅋$)
Let $\mathcal{R}$ be a connected proof net, if T is a terminal $\otimes$ node which is not splitting and if e_1 and e_2 are its premisses, there exists a $⅋$ node P with premisses e'_1 and e'_2 such that e_1 is connected to e'_1 by a strong switching path γ_1 and e_2 is connected to e'_2 by a strong switching path γ_2, with γ_1 and γ_2 disjoint and not containing the conclusion e'_0 of P.

Such a $\parr$ node is called a *blocking* $\parr$ of the $\otimes$ node, the two paths γ_1 and γ_2 are called *blocking paths*.

PROOF: Since T is not splitting, there is an undirected path γ connecting e_1 and e_2 in $\mathcal{R}$ (without going through T). By acyclicity, γ does not belong to any switching graph of $\mathcal{R}$. We consider an arbitrary switching graph $\mathcal{R}^\varphi$ of $\mathcal{R}$. By acyclicity and connectedness of $\mathcal{R}^\varphi$, removing T from $\mathcal{R}^\varphi$ defines exactly two connected components $\mathcal{R}_1$ containing e_1 and $\mathcal{R}_2$ containing e_2. Since γ is not included in $\mathcal{R}^\varphi$, there must be a $\parr$ node P in γ such that a premisse e'_1 of P is in $\mathcal{R}_1$ and the other premisse e'_2 of P is in $\mathcal{R}_2$. Let γ_1 (resp. γ_2) be the part of γ included in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$), γ_1 (resp. γ_2) gives us a strong path from e_1 to e'_1 (resp. from e_2 to e'_2).

Now assume γ_1 contains e'_0 (or similarly for γ_2), then if φ' is obtained from φ by modifying the premisse of P connected with P, one obtains a cycle in $\mathcal{R}^{\varphi'}$ by concatenating the prefix of γ_1 ending with e'_0 and γ_2. This would thus contradict the acyclicity of $\mathcal{R}^{\varphi'}$. □

A *descent path* is a maximal (with respect to the prefix relation) directed path from a node (downwards) to a conclusion or a cut. A descent path is always a strong switching path. Such a path is unique for all nodes but *hyp* nodes which have as many descent paths as conclusions. A node has all its descent paths of length 1 if and only if it is a terminal node.

Theorem 1.1 (Sequentialization)
Any connected multiplicative proof net is the translation of a sequent calculus proof of MLL.

PROOF: By induction on the number of nodes of the connected proof net $\mathcal{R}$ plus its number of cuts, we build an MLL proof with a rule associated with each node of $\mathcal{R}$.

If $\mathcal{R}$ with conclusions Γ contains a *cut* node, we turn it into a $\otimes$ node T (with a new conclusion edge labelled $A \otimes A^\perp$ and a new associated c node). By induction hypothesis, there exists a proof π' associated with the thus obtained connected proof net. If R is the $(\otimes)$ rule of π' associated with T, we obtain π by turning R into a (cut) rule:

$$\dfrac{\begin{array}{c}\pi'\\ \vdots\end{array}}{\begin{array}{c}\vdash \Delta, A \otimes A^\perp\\ \vdots \\ \vdash \Gamma, A \otimes A^\perp\end{array}}\otimes \qquad \mapsto \qquad \dfrac{\begin{array}{c}\pi\\ \vdots\end{array}}{\begin{array}{c}\vdash \Delta\\ \vdots \\ \vdash \Gamma\end{array}}\mathit{cut}$$

We can now assume $\mathcal{R}$ is cut free.

Using Lemma 1.4, we assume a blocking $\parr$ node (with its two corresponding blocking paths) is associated with each terminal non splitting $\otimes$ node.

Since $\mathcal{R}$ is connected it cannot be empty and thus it must contain at least one *hyp* node (they are the only nodes with no premisse).

If $\mathcal{R}$ contains a terminal *hyp* node with conclusions labelled $A_1, \ldots, A_n$, then by connectedness, $\mathcal{R}$ is reduced to this *hyp* node (with its conclusions and the associated c nodes). We sequentialize it into a proof of $\vdash A_1, \ldots, A_n$ in MLL (the last rule of the proof is the rule associated with the *hyp* node).

Otherwise, we follow a non trivial (length at least 2) descent path from a non terminal *hyp* node. We reach a terminal $\parr$ or $\otimes$ node. If we reach a $\parr$ node P, we remove it (as well as its conclusions and the associated c nodes) and we replace it with two new c nodes. Let $\mathcal{R}'$ be the thus obtained connected proof net, by induction hypothesis there exists a corresponding MLL proof π'. The proof π associated with $\mathcal{R}$ is then:

$$\dfrac{\begin{array}{c}\pi' \\ \vdash \Gamma, A, B\end{array}}{\vdash \Gamma, A \parr B}\ \parr$$

The last ($\parr$) rule is the rule associated with P.

If we reach a splitting $\otimes$ node T, we remove it (as well as its conclusions and the associated c nodes) and we replace it with two new c nodes. Let $\mathcal{R}'_1$ and $\mathcal{R}'_2$ be the two thus obtained connected proof nets, by induction hypothesis there exist two corresponding MLL proofs π'_1 and π'_2. The proof π associated with $\mathcal{R}$ is then:

$$\dfrac{\begin{array}{c}\pi'_1 \\ \vdash \Gamma, A\end{array} \qquad \begin{array}{c}\pi'_2 \\ \vdash \Delta, B\end{array}}{\vdash \Gamma, \Delta, A \otimes B}\ \otimes$$

The last ($\otimes$) rule is the rule associated with T.

If we reach a non splitting $\otimes$ node, we start building a sequence of terminal non splitting $\otimes$ nodes in the following way: if we arrive to the $\otimes$ node T by its premisse e_1, we go to the associated blocking $\parr$ node P through the strong splitting path starting from the other premisse e_2 of T, we then follow the descent path from P, we reach a terminal node. If it is a non splitting $\otimes$ node, we extend our sequence, otherwise we are in one of the above cases. It is thus enough to prove that this sequence cannot be infinite. Since $\mathcal{R}$ contains a finite number of nodes, such an infinite sequence must correspond to a cyclic path in $\mathcal{R}$. We now show this infinite sequence would induce a cycle in a switching graph of $\mathcal{R}$ contradicting correctness.

Let γ be the path used to build the sequence, we look at it and we stop when we arrive to a node belonging to a blocking path of a previously met $\otimes$ node (not necessarily the blocking path of this $\otimes$ node we went through) or to a descent path of a previously met $\parr$ node. If we stop on a descent path on a node n, then the suffix of the considered path starting from the conclusion of n contradicts Lemma 1.3. If we stop on a node n in a blocking path γ_1^b of a $\otimes$ node T whose other blocking path is γ_2^b and whose blocking $\parr$ node is P, and if n was visited from its conclusion to its premisse in γ_1^b, we consider the concatenation $\gamma_2^b \gamma_{Pn} \bar{\gamma}_n^b$ (γ_{Pn} being the sub-path of γ from P to n and γ_n^b the sub-path of γ_1^b from T to n). By Lemma 1.2, this is a strong switching path, and we contradict Lemma 1.3. If n was visited from a premisse to its conclusion in γ_1^b, the situation is similar to the case where n was in a descent path. $\square$

It is then natural to try to analyse the kernel of the translation by understanding when two different sequent calculus proofs are mapped to the same proof structure. One can prove that it is the case if and only if one can transform one of the two proofs to the other by some permutations of the order of application of rules.

In a cut-free sequent calculus proof or proof structure, the formulas used in the *ax* rules or nodes are occurrences of sub-formulas of the conclusions of the proof or proof structure. Two proofs are mapped to the same proof structure if and only if the pairing of such occurrences of formulas given by *ax* rules are the same in the two proofs.

1.5 Cut Elimination

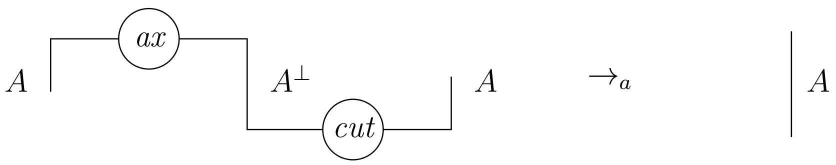
If we propose proof nets as an alternative to sequent calculus to study proofs in (multiplicative) linear logic, we need to be able to deal with cut elimination in this new syntax without referring to the sequent calculus.

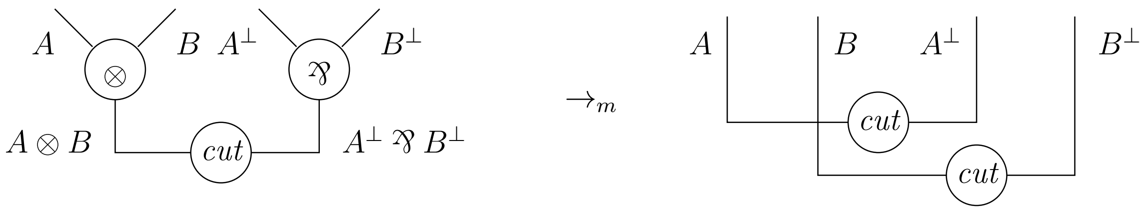
Cut elimination in proof nets is defined as a graph rewriting procedure, which acts through local transformations of the proof net.

We first define the transformation on proof structure, but we will also immediately restrict to the case of proof nets.

1.5.1 Reductions Steps

We consider two reductions steps:





In the a step, the two edges of type A in the redex are supposed to be distinct.

Examples

One can check that in a proof net any cut belongs to a redex: if the sources of the premisses of the cut are not *ax* nodes, they must be $\otimes$ or ⅋ nodes and, due to the typing constraints, they cannot be both $\otimes$ nodes or both ⅋ nodes. Moreover, by acyclicity, the sources of the two premisses of a cut cannot be the same *ax* node. As a consequence normal forms for the reduction of multiplicative proof structures which are proof nets are exactly cut-free proof nets.

1.5.2 Preservation of Correctness

Lemma 1.5 (Preservation of Acyclicity)
If $\mathcal{R}$ is a multiplicative proof net and $\mathcal{R} \to \mathcal{R}'$ then $\mathcal{R}'$ is a proof net.

Proof: We consider the two steps:

- Through an a step, a switching graph of the reduct can be turned into a switching graph of the redex by replacing the edge of type A with a path of length 3 going through the *ax* node and through the *cut* node. One of these two switching graphs is then acyclic if and only if the other one is.

- Through an m step, a switching graph $\mathcal{S}$ of the reduct gives rise to two switching graphs $\mathcal{S}_1$ and $\mathcal{S}_2$ in the redex depending of the $\{\text{left}, \text{right}\}$ choice for the $\parr$ node which disappears through the reduction. Assume there is a cycle in $\mathcal{S}$. It must go through at least one of the cuts otherwise it is a cycle in $\mathcal{S}_1$ and $\mathcal{S}_2$. If it goes through the cut between A and $A^\perp$ thus the premisses of this cut are connected in $\mathcal{S}$ (without using the cut) and then we have a cycle in $\mathcal{S}_1$, a contradiction. Similarly it cannot be a cycle using only the cut between B and $B^\perp$. If it uses both cuts, the premisses A and B are connected in $\mathcal{S}$ and we have a cycle in both $\mathcal{S}_1$ and $\mathcal{S}_2$, or the premisses A and $B^\perp$ are connected in $\mathcal{S}$ and we have a cycle in $\mathcal{S}_2$. □

Remember that, thanks to Lemma 1.1, all the switching graphs of a multiplicative proof net have the same number of connected components.

Lemma 1.6 (Preservation of Connected Components)
If $\mathcal{R}$ is a multiplicative proof net and $\mathcal{R} \to \mathcal{R}'$ then the number of connected components of the switching graphs of $\mathcal{R}'$ is the same as for the switching graphs of $\mathcal{R}$.

PROOF: The switching graphs are acyclic in both $\mathcal{R}$ and $\mathcal{R}'$ (see Lemma 1.5). We can thus use Lemma A.1. We consider the two reduction steps. In each case, we loose two nodes and two edges thus the number of connected components is not modified.

In particular a reduct of a connected multiplicative proof net is a connected proof net.

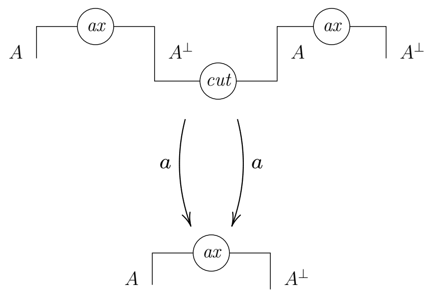
1.5.3 Properties

If we consider cut elimination as a computational process on proof nets, the two key properties we want to prove about it are termination and uniqueness of the result. If the existence of a terminating reduction strategy (weak normalization) allowing to reach a cut-free proof net from any proof net is enough from the point of few of logical consistency, it is more satisfactory from a computational point of view to prove that any reduction will eventually terminate (strong normalization).

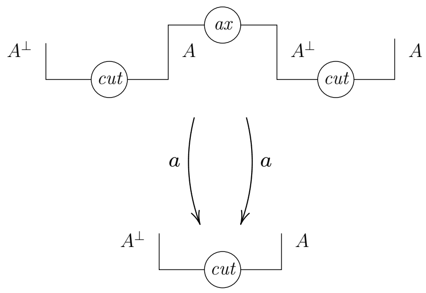
Lemma 1.7 (Sub-Confluence)
The reduction of multiplicative proof nets is sub-confluent.

PROOF: There are two kinds of critical pairs:

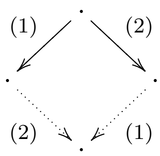
- a/a (shared *cut*)



- a/a (shared ax)



In all the other situations, two different reductions from a given proof net commute:



since they cannot overlap. $\square$

Proposition 1.2 (Confluence)
The reduction of multiplicative proof nets is confluent.

Proof: By Proposition B.1 and Lemma 1.7. $\square$

Proposition 1.3 (Strong Normalization)
The reduction of multiplicative proof nets is strongly normalizing.

Proof: The reduction of multiplicative proof nets is s-decreasing where s is the number of nodes of the proof net. We conclude with Proposition B.2. $\square$

1.6 The Mix Rules

We consider an extension of LL which will make the study of exponential proof nets easier.

The two mix rules are the nullary mix rule (*void*) and the binary mix rule (*mix*).

$$\frac{}{\vdash}\,void \qquad\qquad \frac{\vdash \Gamma \qquad \vdash \Delta}{\vdash \Gamma, \Delta}\,mix$$

Lemma 1.8 (Sociability of (*void*))
If π is a proof in MLL *with (void) and (mix) rules, by applying (possibly many times) the transformation:*

$$\dfrac{\dfrac{}{\vdash}\,void \qquad \begin{array}{c}\vdots\\ \vdash \Gamma\end{array}}{\vdash \Gamma}\,mix \qquad \mapsto \qquad \begin{array}{c}\vdots\\ \vdash \Gamma\end{array}$$

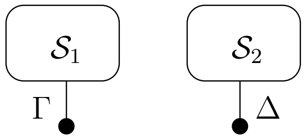
we obtain either the proof $\dfrac{}{\vdash}\,void$ *or a proof without the (void) rule.*

Proof: The transformation described can only be applied a finite number of times (the number of rules strictly decreases). Assume we apply it as many times as possible. If the obtained proof contains an occurrence of the (*void*) rule, it is the only rule of the proof since the only possible rule below it is (*mix*) (it must admit the empty sequent $\vdash$ as a premisse) but then the transformation can be applied one more time, a contradiction. $\square$

We can interpret these two rules as proof structures constructions.

The (*void*) rule is translated into the empty proof structure.

The (*mix*) rule applied to two proofs π_1 and π_2 which translate into the proof structures $\mathcal{S}_1$ and $\mathcal{S}_2$ leads to the disjoint union of $\mathcal{S}_1$ and $\mathcal{S}_2$.



One can note that the transformation given on Lemma 1.8 does not modify the associated proof structure.

Proposition 1.4 (Soundness and Sequentialization with (*void*))
A multiplicative proof structure is the translation of a sequent calculus proof of MLL *with (void) if and only if it is acyclic and its switching graphs have at most one connected component.*

PROOF: By Lemma 1.8, for soundness it is enough to apply Proposition 1.1 and to see that the empty proof structure (obtained from the (*void*) rule) has empty switching graphs thus is acyclic and with no connected component.

Concerning sequentialization, since the only multigraph with no connected component is the empty one, the only multiplicative proof structure with switching graphs with no connected component is the empty one which is the translation of the (*void*) rule. For acyclic and connected multiplicative proof structures, we apply Theorem 1.1. □

Lemma 1.9 (Connection)
If $\mathcal{R}$ is a cut-free multiplicative proof net which is a connected directed multigraph and which is not a connected proof net, there exists a $⅋$ node in $\mathcal{R}$ such that $\mathcal{R}'$ obtained by transforming it into a $\otimes$ node is still a multiplicative proof net.

Proposition 1.5 (Soundness and Sequentialization with (*mix*))
A multiplicative proof structure is the translation of a sequent calculus proof of MLL *with (mix) if and only if it is acyclic and its switching graphs have at least one connected component.*

Note that asking the switching graphs to have at least one connected component is equivalent to ask them not to be empty and thus it is equivalent to ask the proof structure itself not to be empty.

PROOF: The only if part is given by Proposition 1.1 for the MLL rules, and the translation of the (*mix*) rule does not introduce cycles and is not empty (since the translations of the premisses satisfy these properties).

We turn to the if part, and consider an acyclic proof structure $\mathcal{S}$. As for the proof of Theorem 1.1, we can transform *cut* nodes into $\otimes$ nodes and focus on the cut-free case. We go by induction on the number of connected components of the proof structure as a directed multigraph (not of its switching graphs). The proof structure has at least one connected component since its switching graphs do.

- If there is 1 component, we go by induction on the number k of connected components of the switching graphs of $\mathcal{S}$ (which is the same for all switching graphs thanks to Lemma 1.1). If $k = 1$, we apply Theorem 1.1. If $k > 1$, using Lemma 1.9, we obtain a multiplicative proof net $\mathcal{S}'$ by turning a $⅋$ node into a $\otimes$ node T. By induction hypothesis, there is a sequent calculus proof π' associated with $\mathcal{S}'$. By focussing on the ($\otimes$) rule corresponding to T in π', we can decompose it into:

$$\dfrac{\begin{array}{cc}\pi_1' & \pi_2' \\ \vdash \Gamma, A & \vdash B, \Delta\end{array}}{\vdash \Gamma, A \otimes B, \Delta}\otimes$$

$$\begin{array}{c}\vdots \\ \vdash \Xi\end{array}$$

One can see the following proof is a sequentialization of $\mathcal{S}$:

$$\dfrac{\dfrac{\begin{array}{cc}\pi_1' & \pi_2' \\ \vdash \Gamma, A & \vdash B, \Delta\end{array}}{\vdash \Gamma, A, B, \Delta}\,mix}{\vdash \Gamma, A \parr B, \Delta}\parr$$

$$\begin{array}{c}\vdots \\ \vdash \Xi\end{array}$$

- If there are $n+1$ $(n>0)$ components, we add a $\parr$ node P between two conclusions of $\mathcal{S}$ belonging to different connected components. The obtained proof structure $\mathcal{S}'$ has n components, so by induction hypothesis we obtain a sequent calculus proof π' corresponding to $\mathcal{S}'$. π' contains an occurrence of $(\parr)$ rule corresponding to P. By reversibility of the $(\parr)$ rule, we transform π' into π'' by moving down this occurrence of rule so that it becomes the last rule of the proof. If π''' is the premisse of the last rule of π'', one can check π''' is a sequentialization of $\mathcal{S}$. □

Proposition 1.6 (Soundness and Sequentialization with (*void*) and (*mix*))
A multiplicative proof structure is the translation of a sequent calculus proof of MLL *with (void) and (mix) if and only if it is acyclic.*

Proof: By Lemma 1.8, soundness is obtained from Propositions 1.4 and 1.5

Concerning sequentialization, almost as in Proposition 1.4, either the proof structure is empty and it is the translation of the (*void*) rule, or we apply Proposition 1.5 □

2 Multiplicative Exponential Proof Nets

We introduce now the exponential connectives which provides linear logic with real expressive power. The rewriting theory of proof nets becomes much richer.

2.1 Multiplicative Exponential Linear Logic with Mix

The formulas of multiplicative exponential linear logic (MELL) are defined as:

$$A, B ::= X \mid X^{\perp} \mid A \otimes B \mid A \parr B \mid !A \mid ?A$$

The connective $(.)^{\perp}$ is extended into an involution on all formulas by:

$$(!A)^{\perp} = ?A^{\perp} \qquad (?A)^{\perp} = !A^{\perp}$$

For MELL, we consider the rules of MLL as well as the two mix rules, together with:

$$\dfrac{}{\vdash ?A}\,?w_0 \qquad \dfrac{\vdash \Gamma, ?A, ?A}{\vdash \Gamma, ?A}\,?c \qquad \dfrac{\vdash \Gamma, A}{\vdash \Gamma, ?A}\,?d \qquad \dfrac{\vdash ?\Gamma, A}{\vdash ?\Gamma, !A}\,!$$

Due to the presence of mix rules, our presentation of the weakening rule $(?w_0)$ is equivalent to the more traditional one $\dfrac{\vdash \Gamma}{\vdash \Gamma, ?A}\,?w$. The two rules are inter-derivable.

$$\dfrac{\vdash \Gamma \qquad \dfrac{}{\vdash ?A}\,?w_0}{\vdash \Gamma, ?A}\,mix \qquad\qquad \dfrac{\dfrac{}{\vdash}\,void}{\vdash ?A}\,?w$$

2.2 Proof Structures

boxes $\mathcal{B}$, *main door*, with explicit $?p$ nodes (*auxiliary doors*)

content of a box

The $?$-*tree* of an edge of type $?\_$ is defined inductively by:

- If the edge is conclusion of an *ax* node, its ?-tree is empty.
- If the edge is conclusion of a $?d$ node, its ?-tree is this $?d$ node.
- If the edge is conclusion of a $?w$ node, its ?-tree is this $?w$ node.
- If the edge is conclusion of a $?c$ node, its ?-tree is this $?c$ node together with the ?-trees of the two premisses of the $?c$ node.
- If the edge is conclusion of a $?p$ node, its ?-tree is this $?p$ node together with the ?-tree of the its premisse.

The *size* of a ?-tree is its number of nodes.

descent path (bis): from a node downwards to a conclusion or to a cut or to a premisse of ! node (that is we do not continue down through an ! node)

2.3 Correctness Criterion

acyclicity

sequentialization

2.4 Cut Elimination

2.4.1 Reductions Steps

A *numbered proof net* is a proof net together with a strictly positive natural number, as well as a strictly natural number associated with each box. All these natural numbers are called *labels* of the numbered proof net. Numbered proof nets will mainly be a tool to prove properties of the normalization of proof nets. We define reduction steps on numbered proof nets, but the corresponding notion for proof nets can simply be obtained by forgetting labels.

- a: $n \mapsto n+1$
- m: $n \mapsto n+1$
- d: $n, m \mapsto n+m+1$
- c: $n, m \mapsto n, m, m$
- w: $n, m \mapsto n$
- p: $n, m, k \mapsto n, m, k$

Lemma 2.1 (Preservation of Correctness)
If $\mathcal{R}$ is a proof net and $\mathcal{R} \to \mathcal{R}'$ then $\mathcal{R}'$ is a proof net.

2.4.2 Properties

The goal of this section is to prove the strong normalization and the confluence of the reduction of proof nets.

Lemma 2.2 (Numbered Congruence)
If $\mathcal{R}$ is a proof net containing $\mathcal{R}_0$ as a sub proof net a depth 0, if $\mathcal{R}_0$ equipped with label m reduces to $\mathcal{R}'_0$ with label m' then $\mathcal{R}$ reduces to $\mathcal{R}'$ where $\mathcal{R}'$ is obtained from $\mathcal{R}$ by replacing $\mathcal{R}_0$ with $\mathcal{R}'_0$ and the label of $\mathcal{R}'$ is $n + m' - m$ (where n is the label of $\mathcal{R}$).

Proposition 2.1 (Local Confluence)
The reduction of numbered proof nets is locally confluent.

Proof:

- a/a (shared *cut*)

$$\begin{tikzcd} n \arrow[d, bend right, "a"'] \arrow[d, bend left, "a"] \\ n+1 \end{tikzcd}$$

- a/a (shared *ax*)

$$\begin{tikzcd} n \arrow[d, bend right, "a"'] \arrow[d, bend left, "a"] \\ n+1 \end{tikzcd}$$

- d/in

$$\begin{tikzcd} & n, m \arrow[dl, "d"'] \arrow[dr, "\text{in}"] & \\ n+m+1 \arrow[dr, dotted, "\text{in}"'] & & n, m' \arrow[dl, dotted, "d"] \\ & n+m'+1 & \end{tikzcd}$$

- c/in

$$\begin{tikzcd} & & n, m \arrow[dll, "c"'] \arrow[dr, "\text{in}"] & \\ n, m, m \arrow[dr, dotted, "\text{in}"'] & & & n, m' \arrow[ddl, dotted, "c"] \\ & n, m', m \arrow[dr, dotted, "\text{in}"'] & & \\ & & n, m', m' & \end{tikzcd}$$

- w/in

$$\begin{tikzcd} n, m \arrow[dd, bend right, "w"'] \arrow[dr, "\text{in}"] & \\ & n, m' \arrow[dl, dotted, "w"] \\ n & \end{tikzcd}$$

- p/in (left side)

$$\begin{tikzcd}
 & n,m,k \arrow[dl, "p"'] \arrow[dr, "\mathrm{in}"] & \\
n,m,k \arrow[dr, dotted, "\mathrm{in}"'] & & n,m',k \arrow[dl, dotted, "p"] \\
 & n,m',k &
\end{tikzcd}$$

- p/in (right side)

$$\begin{tikzcd}
 & n,m,k \arrow[dl, "p"'] \arrow[dr, "\mathrm{in}"] & \\
n,m,k \arrow[dr, dotted, "\mathrm{in}"'] & & n,m,k' \arrow[dl, dotted, "p"] \\
 & n,m,k' &
\end{tikzcd}$$

- d/p

$$\begin{tikzcd}
n,m,k \arrow[r, "p"] \arrow[d, bend right, "d"'] & n,m,k \arrow[dl, dotted, "d"] \\
n+m+1,k &
\end{tikzcd}$$

- c/p

$$\begin{tikzcd}
 & & n,m,k \arrow[dl, "c"'] \arrow[ddr, "p"] & \\
 & n,m,m,k \arrow[dl, dotted, "c"'] & & \\
n,m,m,k,k \arrow[dr, dotted, "p"'] & & & n,m,k \arrow[ddl, dotted, "c"] \\
 & n,m,m,k,k \arrow[dr, dotted, "p"'] & & \\
 & & n,m,m,k,k &
\end{tikzcd}$$

- w/p

$$\begin{tikzcd}
 & n,m,k \arrow[dl, "w"'] \arrow[dr, "p"] & \\
n,k \arrow[dr, dotted, "w"'] & & n,m,k \arrow[dl, dotted, "w"] \\
 & n &
\end{tikzcd}$$

- p/p

$$\begin{tikzcd}
 & & n,m,k,l \arrow[dll, "p"'] \arrow[dr, "p"] & \\
n,m,k,l \arrow[dr, dotted, "p"'] & & & n,m,k,l \arrow[ddl, dotted, "p"] \\
 & n,m,k,l \arrow[dr, dotted, "p"'] & & \\
 & & n,m,k,l &
\end{tikzcd}$$

$\square$

Proposition 2.2 (Weak Normalization)
The reduction of proof nets is weakly normalizing.

PROOF: We define a *size* associated with each cut of a proof net $\mathcal{R}$. It is a pair of natural numbers (s,t) where s is the size of the cut formula (*i.e.* the size of the types of the premisses of the *cut* node) and t is the size of the ?-tree above the ? premisse of the cut if any, and $t = 0$ otherwise. These pairs are ordered lexicographically. The *cut size* of the proof net $\mathcal{R}$ is the multiset of the sizes of its cuts. Thanks to the multiset ordering, the cut sizes are well ordered.

We now prove that it is always possible to reduce a cut in a proof net $\mathcal{R}$ in a way which makes its size strictly decrease. By Proposition B.2, this proves the weak normalization property.

A cut is of *exponential type* if the types of its premisses are $!A$ and $?A^\perp$ for some A. Note the source of the premisse with type $!A$ of a cut of exponential type must be an *ax* node or an ! node.

- If $\mathcal{R}$ contains an a redex for which the cut is not of exponential type, we reduce it. A cut disappears and the sizes of the other cuts are not modified.
- If $\mathcal{R}$ contains an m redex, we reduce it. If $A \otimes B$ and $A^\perp \parr B^\perp$ are the types of the premisses of the cut, we replace a cut of size $(s_A + s_B + 1, 0)$ by two cuts of sizes $(s_A, \_)$ and $(s_B, \_)$ (and the sizes of the other cuts are not modified), thus the cut size of the proof net strictly decreases.
- If $\mathcal{R}$ has only cuts of exponential types, we consider the following relation on cuts: $c \prec c'$ if one of the following two properties holds:
  - The $!A$ premisse of c has an *ax* node as source and there is a descent path from the $?A^\perp$ conclusion of this *ax* node to c'.
  - The $!A$ premisse of c has an ! node with box $\mathcal{B}$ as source and there is a descent path from an auxiliary door of $\mathcal{B}$ to c'.

  We are going to show that $\prec$ is an acyclic relation on the cuts of $\mathcal{R}$. Let us consider a minimal cycle $c_0 \prec c_1 \prec \cdots \prec c_n$ with $n > 0$ and $c_n = c_0$, it induces a path in $\mathcal{R}$ (enriched with the edges from the main door of each box to its auxiliary doors): from each c_i we go to the ? premisse of c_{i+1} by going to the ! premisse of c_i reaching the main door of the box $\mathcal{B}_i$ (or an *ax* node) then we go to an auxiliary door of $\mathcal{B}_i$ (or to the ? conclusion of the *ax* node) and we follow the descent path until the ? premisse of c_{i+1} (we cannot reach its ! premisse since descent paths stop when going down on the premisse of an ! node). In the case of a minimal cycle, the induced path is a simple undirected path, and all the cuts under consideration must have the same depth since the depth always decreases along the $\prec$ relation. Moreover each $\parr$ node is crossed from one of its premisses to its conclusion. By considering a switching graph which contains all the c_i's (they live in the same boxes) and which connects the $\parr$ nodes of the path with the premisse contained in the path, we would obtain a cycle which contradicts the acyclicity of the proof net.

  Let us now consider the set $\mathcal{C}$ of all cuts which are maximal for the $\prec$ relation (it is finite and not empty since the set of cuts is finite and the relation $\prec$ is acyclic), and let c be a cut of $\mathcal{C}$ of maximal depth, we reduce c. The reduction of c does not modify the size of any other cut since:
  - If c is maximal for $\prec$, has a box $\mathcal{B}$ above its ! premisse, then any cut in $\mathcal{B}$ which is maximal for $\prec$ is maximal in $\mathcal{R}$, so if there is a cut in $\mathcal{B}$ there is a maximal

cut in $\mathcal{B}$ for $\prec$ with bigger depth than c (this contradicts the choice of c, thus the content of $\mathcal{B}$ is cut free).

  – The reduction of c does not modify the type of any other cut.
  – The reduction of c can only modify the ?-trees of cuts c' such that $c \prec c'$ (and there is no such c' thanks to the choice of c).

If the reduction step is an a or w step, a cut disappears, thus the cut size strictly decreases. If the reduction step is a d step, a cut of size $(s+1, 1)$ is replaced by a cut of size $(s, \_)$, thus the cut size strictly decreases. If the reduction step is a c or p step, a cut of size (s, t) is replaced by 2 or 1 cut(s) of size(s) (s, t') with $t' < t$, thus the cut size strictly decreases. □

We define some sub-reduction relations:

- The $\to_{am}$ reduction is the reduction of proof nets obtained by considering only $\to_a$ and $\to_m$ steps.
- The $\to_{\not w}$ reduction is the reduction of proof nets restricted to non w steps.
- The $\to_{\not c}$ reduction is the reduction of proof nets restricted to non c steps.

Lemma 2.3 (Strong *am* Normalization)
The $\to_{am}$ reduction of proof nets is strongly normalizing.

Proof: We use Proposition B.2, since the number of nodes of proof nets is strictly decreasing along an a or m reduction step. □

Lemma 2.4 (Strong *w* Normalization)
The $\to_w$ reduction of proof nets is strongly normalizing.

Proof: We use Proposition B.2, since the number of nodes of proof nets is strictly decreasing along a w reduction step. □

Lemma 2.5 (Sub-Commutation of *am* and non *c*)
The reduction relations $\to_{am}$ and $\to_{\not c}$ sub-commute.

Proof: This easily comes by looking at the proof of Proposition 2.1. □

Lemma 2.6 (Quasi-Commutation of *w* over non *w*)
The $\to_w$ reduction of proof nets quasi-commutes over the $\to_{\not w}$ reduction.

Proof: Assume we have $\mathcal{R} \to_w \mathcal{R}' \to_{\not w} \mathcal{R}''$. If the $\to_w$ and the $\to_{\not w}$ steps do not overlap, we directly have commutation and by first applying the $\to_{\not w}$ step, one obtains $\mathcal{R} \to_{\not w} \mathcal{R}''' \to_w \mathcal{R}''$.

$$\begin{tikzcd} & \mathcal{R} \arrow[dl, "w"'] \arrow[dr, dotted, "\not w"] & \\ \mathcal{R}' \arrow[dr, "\not w"'] & & \mathcal{R}''' \arrow[dl, dotted, "w"] \\ & \mathcal{R}'' & \end{tikzcd}$$

The only possible overlapping is when the $\to_{\not w}$ step acts on a box containing the $\to_w$ step, but then by looking at the _/in cases of the proof of Proposition 2.1, we can see we can close the diagrams in an appropriate way:

$$\begin{tikzcd}[column sep=small]
& \cdot \arrow[dl, "w"'] \arrow[dr, dotted, "d"] & \\
\cdot \arrow[dr, "d"'] & & \cdot \arrow[dl, dotted, "w"] \\
& \cdot &
\end{tikzcd}
\qquad
\begin{tikzcd}[column sep=small]
& \cdot \arrow[dl, "w"'] \arrow[dr, dotted, "c"] & \\
\cdot \arrow[ddr, "c"'] & & \cdot \arrow[d, dotted, "w"] \\
& & \cdot \arrow[dl, dotted, "w"] \\
& \cdot &
\end{tikzcd}
\qquad
\begin{tikzcd}[column sep=small]
& \cdot \arrow[dl, "w"'] \arrow[dr, dotted, "p"] & \\
\cdot \arrow[dr, "p"'] & & \cdot \arrow[dl, dotted, "w"] \\
& \cdot &
\end{tikzcd}$$

□

Lemma 2.7 (Weak non w Normalization)
The $\to_{\not w}$ reduction of numbered proof nets is weakly normalizing.

PROOF: We can use the same proof as for Proposition 2.2, by using the following remarks.

We consider the *non w cut size* of a proof net to be the multiset of the sizes of the non w cuts of $\mathcal{R}$.

Reducing a cut of non exponential type makes the non w cut size strictly decrease.

If $c \prec c'$ (for the $\prec$ relation of the proof of Proposition 2.2) then c' cannot be a w cut. Thus if there are non w cuts in $\mathcal{R}$, the set $\mathcal{C}$ contains non w cuts. We now choose c to be of maximal depth among the non w elements of $\mathcal{C}$, and we reduce c. The only difference with the proof of Proposition 2.2 is that the box above c might contain some w cuts. We then see that the non w cut size strictly decreases. □

Lemma 2.8 (Increasing non w Reduction)
The reduction $\to_{\not w}$ on numbered proof nets is μ-increasing where, for a numbered proof net $\mathcal{R}$, $\mu(\mathcal{R}) = l^2 + p$ with:

- *l is the sum of all the labels of $\mathcal{R}$,*
- *p is the sum of the depths of the boxes of $\mathcal{R}$.*

PROOF: We analyse each non w step $\mathcal{R} \to_{\not w} \mathcal{R}'$, we note l' the sum of the labels of $\mathcal{R}'$ and p' the sum of the depths of the boxes of $\mathcal{R}'$.

- a: $\mu(\mathcal{R}') > \mu(\mathcal{R})$ ($l' = l + 1$ and $p' = p$).
- m: $\mu(\mathcal{R}') > \mu(\mathcal{R})$ ($l' = l + 1$ and $p' = p$).
- d: Let n be the label at the current depth in $\mathcal{R}$ and the same for n' in $\mathcal{R}'$, if m is the label of the box, we have $n' = n + m + 1$ and the other labels are not modified thus $l' = l + 1$. Let D be the depth of $\mathcal{R}$, the depth of the opened box is at most D. Let B be the number of boxes in $\mathcal{R}$, there are at most $B - 1$ boxes inside the opened box in $\mathcal{R}$. The opened box disappears, the depth of the boxes inside it decreases by 1, and the depth of the other boxes is not modified. We thus have $p' \geq p - D - (B - 1)$. Since all the labels are strictly positive numbers, we have $l > B \geq D$. We can deduce:

$$\mu(\mathcal{R}') = l'^2 + p' > (l + 1)^2 + p - 2l = l^2 + 2l + 1 + p - 2l = \mu(\mathcal{R}) + 1$$

- c: The label of the duplicated box is duplicated (as well as for the labels of all the boxes included in it) and the other labels are not modified thus $l' > l$. New boxes are created (the duplicated one and the new boxes in the copy) and the depth of the other boxes is not modified thus $p' \geq p$ and $\mu(\mathcal{R}') > \mu(\mathcal{R})$.

- p: We have $l' = l$. The depth of the right box, as well as the depth of all the boxes included in it, increases by 1. The depth of all the other boxes is not modified thus $p' > p$ and $\mu(\mathcal{R}') > \mu(\mathcal{R})$. □

Theorem 2.1 (Strong Normalization)
The reduction of proof nets is strongly normalizing.

Proof: We first prove the strong normalization of the $\rightarrow_{\not w}$ reduction by means of Proposition B.5: we have Lemmas 2.8 and 2.7, and we can check in the proof of Proposition 2.1 that diagrams with $a \rightarrow_{\not w} b$ and $a \rightarrow_{\not w} c$ can be closed into $b \rightarrow^*_{\not w} d$ and $c \rightarrow^*_{\not w} d$ (that is there is not need for w steps in closing the diagram).

We now apply Proposition B.8 to $\rightarrow_w$ and $\rightarrow_{\not w}$, using Lemmas 2.6 and 2.4. □

Theorem 2.2 (Confluence)
The reduction of proof nets is confluent.

Proof: By Newman's Lemma (Proposition B.4) using Proposition 2.1 and Theorem 2.1. □

2.5 Generalized ? Nodes

We now consider a modified syntax for the exponential connectives in proof nets. The goal is to make more canonical the representation of ?-trees in proof nets. We want a syntax able to realize the fact that the differences between the following ?-trees do not matter:

$$\dfrac{?A \qquad \dfrac{?A \quad ?A}{?A}}{?A} \qquad \text{vs} \qquad \dfrac{\dfrac{?A \quad ?A}{?A} \qquad ?A}{?A}$$

$$?A \qquad \text{vs} \qquad \dfrac{?A \qquad \dfrac{}{?A}}{?A} \qquad \text{vs} \qquad \dfrac{\dfrac{}{?A} \qquad ?A}{?A}$$

Among the different kinds of nodes we used for exponential proof nets, we replace $?d$, $?c$, $?w$ and $?p$ nodes by two new kinds of nodes:

- Nodes labelled p have exactly one premisse and one conclusion. The label of the premisse is the same as the label of the conclusion.
- Nodes labelled ? have an arbitrary number $n \geq 0$ of premisses and one conclusion. The labels of the premisses are the same formula A and the label of the conclusion is $?A$.

In a proof structure, we add the constraint that a p node must be above a p node or above a ? node. In particular it cannot be above a conclusion node.

It is not possible to represent arbitrary proofs of the sequent calculus MELL in this new syntax. We need the slight restriction that the principal connectives of the formulas introduced by (ax) rules is not ? or !. Note however there is an easy transformation of proofs ensuring this property:

$$\dfrac{}{\vdash !A, ?A^\perp}\,ax \qquad \mapsto \qquad \dfrac{\dfrac{\dfrac{}{\vdash A, A^\perp}\,ax}{\vdash A, ?A^\perp}\,?d}{\vdash !A, ?A^\perp}\,!$$

This is an instance of the general notion of axioms expansion of proofs of MELL.

Instead of translating sequent calculus proofs, we will define a translation of the previous proof nets (with ax nodes not introducing formulas with principal connective ? or !) into the new syntax.

translation $(.)^?$ from proof nets to proof nets with ? nodes (just for information): replace maximal ?-trees by a ? node with chains of p nodes above it

correctness

reduction

translation $(.)^{cw}$ into proof nets: use degenerate binary trees (left comb trees)

Lemma 2.9 (Translation of Correctness)
Let $\mathcal{S}$ be a proof structure with ? nodes, $\mathcal{S}$ is acyclic if and only if $\mathcal{S}^{cw}$ is acyclic.

Proposition 2.3 (Simulation)
The translation $(.)^{cw}$ is an injective strict simulation which preserves normal forms from proof nets with ? nodes into proof nets.

Lemma 2.10 (Preservation of Correctness)
Let $\mathcal{R}$ be a proof net with ? nodes which reduces into $\mathcal{R}'$, $\mathcal{R}'$ is a proof net.

PROOF: By Lemma 2.9, $\mathcal{R}^{cw}$ is acyclic. By Proposition 2.3, $\mathcal{R}^{cw} \to^+ \mathcal{R}'^{cw}$, thus by Lemma 2.1 $\mathcal{R}'^{cw}$ is acyclic. By Lemma 2.9 again, $\mathcal{R}'$ is acyclic. □

Proposition 2.4 (Strong Normalization)
The reduction of proof nets with ? nodes is strongly normalizing.

PROOF: By Propositions B.6 and 2.3 and Theorem 2.1. □

Proposition 2.5 (Confluence)
The reduction of proof nets with ? nodes is confluent.

PROOF: By Proposition B.1 and Theorem 2.2, proof nets have the unique normal form property. By Propositions B.7 and 2.3, proof nets with ? nodes have the unique normal form property. By Propositions B.3, B.2 and 2.4, proof nets with ? nodes are confluent. □

3 Translation of the Lambda-Calculus

3.1 The Lambda-Calculus inside Linear Logic

Given a denumerable set of λ-variables x, y, ..., the *terms* of the λ-calculus (or λ-*terms*) are:

$$t, u ::= x \mid \lambda x.t \mid t\,u$$

where λ is a binder for x in $\lambda x.t$ and terms are considered up to α-renaming of bound variables.

We assume given a denumerable set of ground types α, β, ... The simple types of the λ-calculus are:

$$\tau, \sigma ::= \alpha \mid \tau \to \sigma$$

Typing judgements are of the shape $\Gamma \vdash t : \tau$ where Γ is a finite partial function from λ-variables to simple types. The typing rules of the simply typed λ-calculus are:

$$\frac{}{\Gamma, x : \tau \vdash x : \tau}\ var$$

$$\frac{\Gamma, x : \tau \vdash t : \sigma}{\Gamma \vdash \lambda x.t : \tau \to \sigma}\ abs \qquad\qquad \frac{\Gamma \vdash t : \tau \to \sigma \qquad \Gamma \vdash u : \tau}{\Gamma \vdash t\,u : \sigma}\ app$$

We assume given a bijection $(.)^\bullet$ from the ground types of the simply typed λ-calculus to the atoms of linear logic. We extend it to any simple type by:

$$(\tau \to \sigma)^\bullet = ?\tau^{\bullet\perp} \parr \sigma^\bullet$$

$$L, M ::= X \mid ?L^\perp \parr M$$

3.2 Directed Proof Nets

$$D, E ::= X \mid D \parr E \mid ?U$$
$$U, V ::= X^\perp \mid U \otimes V \mid !D$$

$L \subsetneq D$ and $L^\perp \subsetneq U$
with generalized ? nodes: appropriate definition of the orientation of edges
mention cut-free correctness

3.3 The Translation

into directed proof nets using only sub-formulas of L (or dual) and only D conclusions

3.3.1 Definition

Pre-translation $(.)^\circ$

$$\dfrac{\dfrac{\dfrac{}{\vdash L^\perp, L}\,ax}{\vdash ?L^\perp, L}\,?d}{\vdash ?\Gamma^\perp, ?L^\perp, L}\,?w$$

$$\dfrac{\vdash ?\Gamma^\perp, ?L^\perp, M}{\vdash ?\Gamma^\perp, ?L^\perp \parr M}\,\parr$$

$$\dfrac{\dfrac{\vdash ?\Gamma^\perp, ?L^\perp \parr M \qquad \dfrac{\dfrac{\vdash ?\Gamma^\perp, L}{\vdash ?\Gamma^\perp, !L}\,! \qquad \dfrac{}{\vdash M^\perp, M}\,ax}{\vdash ?\Gamma^\perp, !L \otimes M^\perp, M}\,\otimes}{\vdash ?\Gamma^\perp, ?\Gamma^\perp, M}\,cut}{\vdash ?\Gamma^\perp, M}\,?c$$

By looking at the proof of Lemma 2.5, one can see $\to_{am}$ is sub-confluent thus it satisfies the unique normal form property (Proposition B.1). Moreover $\to_{am}$ is strongly normalizing (Lemma 2.3), thus we can define the *multiplicative normal form* $\mathrm{NF}_{am}(\mathcal{R})$ of a proof net $\mathcal{R}$ as its unique $\to_{am}$ normal form.

We define the translation $t^\bullet$ of λ-term t by $t^\bullet = \mathrm{NF}_{am}(t^\circ)$.

3.3.2 Simulations

Substitution Lemma for $(.)^\circ$
$(.)^\circ$ is a strict simulation of β-reduction
translation $(.)^\bullet$ of a β-redex: $(\lambda y.t)\,u$

$$\dfrac{\vdash ?\Gamma^\perp, ?L^\perp, M \qquad \dfrac{\vdash ?\Gamma^\perp, L}{\vdash ?\Gamma^\perp, !L}\,!}{\vdash ?\Gamma^\perp, M}\,cut$$

cuts correspond to β-redexes through $(.)^\bullet$
$(.)^\bullet$ is an injective strict simulation of β-reduction which preserves normal forms
strong normalization and confluence of the simply typed λ-calculus

3.3.3 Image

We already mentioned that proof nets obtained from λ-term by means of the $(.)^{\bullet}$:

- only contain edges labelled with sub-formulas of formulas generated by the grammar L (or of their dual),
- and only contain exponential cuts.

One can remark as well that all conclusions are labelled with formulas of the shape L or $?L^{\perp}$.

A proof net satisfying these three conditions is called a *λ-proof net.*

Theorem 3.1 (Sequentialization)
Any λ-proof net is the image of a λ-term through the translation $(.)^{\bullet}$.

3.3.4 Kernel

The *σ-reduction* is the congruence on λ-terms generated by:

$$\begin{array}{lr} ((\lambda y.t)\,u)\,v \to_{\sigma} (\lambda y.(t\,v))\,u & y \notin v \\ (\lambda y.\lambda x.t)\,u \to_{\sigma} \lambda x.((\lambda y.t)\,u) & x \notin u \end{array}$$

The *σ-equivalence* is the equivalence relation generated by the σ-reduction.

Lemma 3.1 (Strong Normalization)
The σ-reduction is strongly normalizing.

The σ-reduction is not locally confluent, as one can see with the following example:

$$\begin{tikzcd} & (\lambda y.\lambda z.x)\,u\,v \arrow[dl, "\sigma"'] \arrow[dr, "\sigma"] & \\ (\lambda y.((\lambda z.x)\,v))\,u \arrow[d, "\sigma"', "/" marking] & & (\lambda z.((\lambda y.x)\,u))\,v \arrow[d, "\sigma", "/" marking] \\ {} & & {} \end{tikzcd}$$

with $y \notin v$ and $z \notin u$.

A λ-term is called a *canonical form* if it is of the shape:

$$\overrightarrow{\lambda z}.\overrightarrow{\beta(y,u)}.(x\,\overrightarrow{v})$$

where $\beta(y,u).t = (\lambda y.t)\,u$ and all the $\overrightarrow{u}$s and $\overrightarrow{v}$s are themselves canonical forms.

Note that β-normal forms are exactly canonical forms without β-redex.

Lemma 3.2 (σ-Normal Forms)
A λ-term is a σ-normal form if and only if it is a canonical form.

Proof: We prove, by induction on its size, that any λ-term t which is a σ-normal form is a canonical form. We can always write t in a unique way as $t = \overrightarrow{\lambda z}.(\_\,\overrightarrow{v})$ where $\_ = x$ or $\_ = \beta(y,u).t'$. In the first case, t is a canonical form (the $\overrightarrow{v}$s are themselves σ-normal forms thus canonical forms by induction hypothesis). In the second case, by induction hypothesis, t' is a canonical form (and u as well), moreover it does not start with a λ (otherwise we have a σ-redex in t). If the sequence $\overrightarrow{v}$ is not empty, we have a σ-redex in t as well. We can conclude that $t = \overrightarrow{\lambda z}.\beta(y,u).t'$ with u and t' in canonical form and t' not starting with a λ, which makes t a canonical form.

Conversely, there is no σ-redex in a canonical form. $\square$

Theorem 3.2 (σ-Equivalence)
Let t and t' be two λ-terms, $t^{\bullet} = t'^{\bullet}$ if and only if $t \simeq_{\sigma} t'$.

3.4 Untyped Lambda-Calculus

The untyped λ-calculus can be seen as the result of quotienting the types of the simply typed λ-calculus by means of an equation $o = o \to o$. Any variable can then be seen as typed with type o and the typing rules become:

$$\frac{}{\Gamma, x : o \vdash x : o}\ var \qquad \frac{\Gamma, x : o \vdash t : o}{\Gamma \vdash \lambda x.t : o}\ abs \qquad \frac{\Gamma \vdash t : o \qquad \Gamma \vdash u : o}{\Gamma \vdash t\,u : o}\ app$$

The information provided by these rules is mainly a super-set of the list of free variables of the term.

One can similarly quotient formulas of linear logic by means of the equation $o = !o \multimap o$, that is $o = ?o^{\perp} ⅋ o$. This entails that the set of the sub-formulas of formulas generated from the atom o by the unique construction $?o^{\perp} ⅋ o$ and of their dual (up to the quotient) contains four elements: o, $\iota = o^{\perp}$, $!o$ and $?\iota$. It is then possible to translate λ-terms as proof net with edges labelled with these four formulas.

4 Further Reading

We suggest an incomplete list of related papers.

4.1 Historical Papers

- The original paper on linear logic which introduces proof nets [Gir87]. The correctness criterion used there is the long trip criterion and the proof technique for sequentialization is based on the theory of empires.
- The definition of the acyclic-connected correctness criterion we use here [DR89].
- The definition of the σ-equivalence on λ-terms [Reg94].

4.2 Sequentialization

- A sequentialization proof based on the acyclic-connected criterion and using empires [Gir91].
- [Dan90]
- [BdW95]
- The sequentialization proof we used here [Lau13].

4.3 Rewriting Properties

- [Ter03]
- [Dan90]
- [PTdF10]

4.4 Extensions of the Syntax

- Units [BCST96, Hug13]
- Quantifiers [Gir91]
- Additive connectives [Gir96, HvG05]

4.5 Relations with the Lambda-Calculus

- [Reg94]
- [DCKP03]

4.6 Complexity

- [Gue11]

A Graphs

Given a set E, $\mathcal{P}^{1,2}(E)$ is the set of all its subsets containing one or two elements. ε is the empty sequence. If s is a finite sequence of elements of a set E and if e is in E then $s \cdot e$ is the finite sequence obtained by adding e at the end of s.

A.1 Multigraphs

A *multigraph* is a triple $(\mathcal{N}, \mathcal{L}, \mathsf{e})$ where $\mathcal{N}$ is the set of *nodes*, $\mathcal{L}$ is the set of *lines*, and e (the *endpoints*) is a function from $\mathcal{L}$ to $\mathcal{P}^{1,2}(\mathcal{N})$. A *loop* is an edge with only one endpoint.

A multigraph is *finite* if it has finitely many nodes and lines.

A *path* in a multigraph $(\mathcal{N}, \mathcal{L}, \mathsf{e})$ is a pair $\sigma = (n_{\mathsf{s}}, \sigma_{\mathsf{l}})$ where n_{s} is a node (the *source* of the path) and σ_{l} is a sequence of lines $(l_i)_{0 \leq i < N}$ (with $N \in \mathbb{N}$) obtained inductively by:

- for any node n, (n, ε) is a path with *target* n, called an *empty path* and noted ε_n,
- if $(n_{\mathsf{s}}, \sigma_{\mathsf{l}})$ is a path with target n_{t} and l is a line with $\mathsf{e}(l) = \{n_{\mathsf{t}}, n'_{\mathsf{t}}\}$ then $(n_{\mathsf{s}}, \sigma_{\mathsf{l}} \cdot l)$ is a path with *target* n'_{t}.

The *length* of the path is N. A *cycle* in a multigraph is a path of length at least 1 from a node to itself (*i.e.* the source and the target are the same node).

Two nodes are *connected* if there is a path from one to the other. A *connected component* of a multigraph is a maximal non-empty subset of its nodes which are all connected together. A multigraph is *connected* if any two nodes are connected, that is if it has exactly one connected component.

Lemma A.1 (Acyclic Connected Components)
In a finite acyclic multigraph, the number of connected components is the number of nodes minus the number of lines.

Proof: By induction on the number of nodes.

- The empty multigraph has no node, no line and no connected component.
- Assume the multigraph contains at least one node. Let n be a node, if it has p lines attached to it, we remove the node and all these lines, we loose one node, p lines and we create $p-1$ connected components (we cannot create more than $p-1$ connected components, and if we create strictly less than $p-1$ connected components, there was a cycle in the multigraph). We can then apply the induction hypothesis. □

Lemma A.2 (Acyclicity and Connectedness)
A multigraph with k lines and $k+1$ nodes is acyclic if and only if it is connected.

Proof: If the multigraph is acyclic, we apply Lemma A.1. If the multigraph is connected, we go by induction on the number of nodes:

- If there is 1 node, there is no edge and the multigraph is acyclic.
- If there are at least $k \geq 2$ nodes, there are $k-1$ edges. By connectedness each node has at least one edge attached to it. If each node has at least two edges attached to it (including loops counted twice), there must be at least k edges (otherwise the average number of endpoints of edges at each node is strictly less than two). This entails that we have a node n which is exactly once the endpoint of an edge. We remove the node and the edge and we apply the induction hypothesis. □

A.2 Directed Multigraphs

A *directed multigraph* is a quadruple $\mathcal{G} = (\mathcal{N}, \mathcal{E}, \mathsf{s}, \mathsf{t})$ where $\mathcal{N}$ is the set of *nodes*, $\mathcal{E}$ is the set of *edges*, and s (the *source*) and t (the *target*) are functions from $\mathcal{E}$ to $\mathcal{N}$.

A directed multigraph is *finite* if it has finitely many nodes and edges.

Let n be a node and e be an edge, if $\mathsf{s}(e) = n$ then e is called an *outgoing edge* of n. If $\mathsf{t}(e) = n$ then e is called an *incoming edge* of n.

A (possibly infinite) *undirected path* in a directed multigraph $(\mathcal{N}, \mathcal{E}, \mathsf{s}, \mathsf{t})$ is a pair $\gamma = (n_{\mathsf{s}}, \gamma_{\mathsf{e}})$ where n_{s} is a node (the *source* of the path, also noted $\mathsf{s}(\gamma)$) and γ_{e} is a (possibly infinite) sequence of pairs $(e_i, \epsilon_i)_{0\leq i<N}$ (with $N \in \mathbb{N}\cup\{\infty\}$) where e_i is an edge and ϵ_i is a *sign* in $\{-,+\}$ such that $\mathsf{s}_\gamma(e_0) = n_{\mathsf{s}}$ and, for any $0 < i < N$, $\mathsf{t}_\gamma(e_{i-1}) = \mathsf{s}_\gamma(e_i)$ (where $\mathsf{s}_\gamma(e_i) = \mathsf{s}(e_i)$ and $\mathsf{t}_\gamma(e_i) = \mathsf{t}(e_i)$ if $\epsilon_i = +$, and $\mathsf{s}_\gamma(e_i) = \mathsf{t}(e_i)$ and $\mathsf{t}_\gamma(e_i) = \mathsf{s}(e_i)$ if $\epsilon_i = -$). The *length* of the undirected path is N. It is *finite* if N is finite. The *target* $\mathsf{t}(\gamma)$ of a finite undirected path $\gamma = (n_{\mathsf{s}}, (e_i, \epsilon_i)_{0\leq i<N})$ is $\mathsf{t}_\gamma(e_{N-1})$ if $N \geq 1$, and n_{s} if $N = 1$. A node n is *internal* to an undirected path γ of length l, if $n = \mathsf{s}_\gamma(e_i)$ with $i > 0$, or $n = \mathsf{t}_\gamma(e_i)$ with $i + 1 < l$. An undirected path is *simple* if it does not contain twice the same edge.

A directed multigraph is (weakly) *connected* if any two nodes are connected by an undirected path.

A finite undirected path γ of length at least 1 is an *undirected cycle* if $\mathsf{t}(\gamma) = \mathsf{s}(\gamma)$.

If $\gamma = (n_{\mathsf{s}}, \gamma_{\mathsf{e}})$ is a finite undirected path with target n_{t} and $\gamma' = (n_{\mathsf{t}}, \gamma'_{\mathsf{e}})$ is an undirected path, their *concatenation* $\gamma\gamma'$ is the undirected path $(n_{\mathsf{s}}, \gamma_{\mathsf{e}}{\cdot}\gamma'_{\mathsf{e}})$. We have $\gamma\varepsilon_{n_{\mathsf{t}}} = \gamma$ and $\varepsilon_{n_{\mathsf{t}}}\gamma' = \gamma'$. γ is a *prefix* of $\gamma\gamma'$.

If $\gamma = (n_{\mathsf{s}}, (e_i, \epsilon_i)_{0\leq i<N})$ is an undirected path and $0 \leq k \leq l < N$, one defines the *sub-paths* $\gamma_{k,l} = (\mathsf{s}_\gamma(e_k), (e_i, \epsilon_i)_{k\leq i<l})$ and the *suffixes* $\gamma_k = (\mathsf{s}_\gamma(e_k), (e_i, \epsilon_i)_{k\leq i<N})$ of γ.

If $\gamma = (n_{\mathsf{s}}, (e_i, \epsilon_i)_{0\leq i<N})$ is a finite undirected path, its *reverse* is the finite undirected path $\bar{\gamma} = (\mathsf{t}(\gamma), (e_{N-i-1}, \bar{\epsilon}_{N-i-1})_{0\leq i<N})$ with $\bar{+} = -$ and $\bar{-} = +$.

If $\mathcal{G} = (\mathcal{N}, \mathcal{E}, \mathsf{s}, \mathsf{t})$ is a directed multigraph, its *underlying multigraph* is the multigraph $\underline{\mathcal{G}} = (\mathcal{N}, \mathcal{E}, \mathsf{e})$ where $\mathsf{e}(e) = \{\mathsf{s}(e), \mathsf{t}(e)\}$. There is an undirected path from n to n' in $\mathcal{G}$ if and only if there is a path from n to n' in $\underline{\mathcal{G}}$. There is an undirected cycle in $\mathcal{G}$ if and only if there is cycle in $\underline{\mathcal{G}}$. $\mathcal{G}$ is connected if and only if $\underline{\mathcal{G}}$ is connected.

A *directed path* in a directed multigraph is an undirected path with $+$ signs only. A *directed cycle* in a directed multigraph is directed path which is an undirected cycle. A *directed acyclic multigraph* is a directed multigraph with no directed cycle.

Lemma A.3 (Directed Acyclic Pre-Order)
In a directed acyclic multigraph, the relation $n \preccurlyeq n'$ if there exists a (finite) directed path from n to n' is a pre-order relation.

PROOF: Thanks to the empty path ε_n, we have $n \preccurlyeq n$, and thanks to concatenation, if $n \preccurlyeq n'$ and $n' \preccurlyeq n''$ then $n \preccurlyeq n''$. □

Lemma A.4
A family of acyclic undirected paths containing the paths of length 0, closed under sub-path and under concatenation induces a pre-order on the nodes of the graph.

A.3 Labelled Multigraphs

Given a set of *labels* $\mathbb{L}$, a multigraph (resp. directed multigraph) is *e-labelled* with $\mathbb{L}$ if it comes with a function from $\mathcal{L}$ (resp. $\mathcal{E}$) to $\mathbb{L}$. It is *n-labelled* with $\mathbb{L}$ if it comes with a function from $\mathcal{N}$ to $\mathbb{L}$.

B Abstract Reduction Systems

We present some basic results about rewriting theory in the setting of abstract reduction systems. The material presented here is strongly inspired from [Ter03].

B.1 Definitions and Notations

An *abstract reduction system* (ARS) $\mathcal{A}$ is a pair $(A, \to)$ where A is a set and $\to$ is a binary relation on A (*i.e.* a subset of $A \times A$).

Given an ARS $\mathcal{A} = (A, \to)$, we use the following notations:

- $a \to b$ if $(a, b) \in \to$. b is called a 1-*step reduct* of a.
- $a \leftarrow b$ if $b \to a$.
- $a \to^{=} b$ if $a = b$ or $a \to b$ ($\to^{=}$ is the reflexive closure of $\to$).
- $a \to^{+} b$ if there exists a finite sequence $(a_k)_{0 \le k \le N}$ $(N \ge 1)$ of elements of A such that $a = a_0$, $a_N = b$ and for $0 \le k \le N-1$, $a_k \to a_{k+1}$ ($\to^{+}$ is the transitive closure of $\to$).
- $a \to^{*} b$ if $a = b$ or $a \to^{+} b$ ($\to^{*}$ is the reflexive transitive closure of $\to$). b is called a *reduct* of a.
- $a \simeq b$ if there exists a finite sequence $(a_k)_{0 \le k \le N}$ $(N \ge 0)$ of elements of A such that $a = a_0$, $a_N = b$ and for $0 \le k \le N-1$, $a_k \to a_{k+1}$ or $a_k \leftarrow a_{k+1}$ ($\simeq$ is the reflexive symmetric transitive closure of $\to$).
- If a is an element of A, the *restriction* of $\mathcal{A}$ to a is the ARS $\mathcal{A} \upharpoonright_a = (A \upharpoonright_a, \to \cap (A \upharpoonright_a \times A \upharpoonright_a))$ where $A \upharpoonright_a = \{b \in A \mid a \to^{*} b\}$ (*i.e.* the set of all reducts of a).

A sequence $(a_k)_{0 \le k < N}$ (with $N \in \mathbb{N}$ such that $N \ge 1$, or $N = \infty$) of elements of A, such that $a_{k-1} \to a_k$ for each $0 < k < N$, is called a *reduction sequence* (starting from a_0 and ending on a_{N-1}, if $N \neq \infty$). When $N \in \mathbb{N}$, the reduction sequence is *finite* and its length is $N - 1$. We use the notation $a \to^{k} b$ if there exists a finite reduction sequence of length k starting from a and ending on b.

B.2 Confluence

An ARS $(A, \to)$ has the *diamond property* if for any a, b and c in A with $a \to b$ and $a \to c$, there exists some d in A such that both $b \to d$ and $c \to d$. Thus diagrammatically:

$$\begin{tikzcd} & a \arrow[dl] \arrow[dr] & \\ b \arrow[dr, dotted] & & c \arrow[dl, dotted] \\ & d & \end{tikzcd}$$

An ARS $(A, \to)$ is *sub-confluent* if for any a, b and c in A with $a \to b$ and $a \to c$, there exists some d in A such that both $b \to^{=} d$ and $c \to^{=} d$. Thus diagrammatically:

$$\begin{tikzcd} & a \arrow[dl] \arrow[dr] & \\ b \arrow[dr, dotted, "="'] & & c \arrow[dl, dotted, "="] \\ & d & \end{tikzcd}$$

An ARS $(A, \to)$ is *locally confluent* if for any a, b and c in A with $a \to b$ and $a \to c$, there exists some d in A such that both $b \to^* d$ and $c \to^* d$. Thus diagrammatically:

$$\begin{tikzcd}
& a \arrow[dl] \arrow[dr] & \\
b \arrow[dr, dotted, "*"'] & & c \arrow[dl, dotted, "*"] \\
& d &
\end{tikzcd}$$

An ARS $(A, \to)$ is *confluent* if for any a, b and c in A with $a \to^* b$ and $a \to^* c$, there exists some d in A such that both $b \to^* d$ and $c \to^* d$. Thus diagrammatically:

$$\begin{tikzcd}
& a \arrow[dl, "*"'] \arrow[dr, "*"] & \\
b \arrow[dr, dotted, "*"'] & & c \arrow[dl, dotted, "*"] \\
& d &
\end{tikzcd}$$

A *normal form* in an ARS $(A, \to)$ is an element a of A such that there is no b in A with $a \to b$ (*i.e.* a has no reduct, but itself).

An ARS $(A, \to)$ has the (weak) *unique normal form property* if for any a in A and any two normal forms b and c in A with $a \to^* b$ and $a \to^* c$, we have $b = c$. Thus diagrammatically:

$$\begin{tikzcd}
& a \arrow[dl, "*"'] \arrow[dr, "*"] & \\
b \arrow[rr, dotted, equal] \arrow[d, "/" marking] & & c \arrow[d, "/" marking] \\
{} & & {}
\end{tikzcd}$$

Proposition B.1 (Confluence Properties)
For any ARS*,*

- *diamond property* $\Longrightarrow$ *sub-confluence* $\Longrightarrow$ *confluence* $\Longrightarrow$ *local confluence,*
- *confluence* $\Longrightarrow$ *unique normal form.*

PROOF: We prove the four implications:

- If $a \to b$ and $a \to c$, the diamond property gives some d such that $b \to d$ and $c \to d$, thus $b \to^= d$ and $c \to^= d$.
- By induction on the length of the reduction sequence from a to b. The following figure might help.

$$\begin{tikzcd}
& & a \arrow[dl, "*"'] \arrow[ddrr, "*"] & & \\
& \cdot \arrow[dl] \arrow[dr] & & & \\
b \arrow[dr, "="'] & & \cdot \arrow[dl, "="] \arrow[dr, "*"'] & & c \arrow[dl, "*"] \\
& \cdot \arrow[dr, "*"'] & & \cdot \arrow[dl, "*"] & \\
& & d & &
\end{tikzcd}$$

  - If $a = b$, we have $b \to^* c$ and $c \to^* c$.

- If $a \to b$, we use an induction on the length of the reduction sequence from a to c:
  - If $a = c$, we have $b \to^* b$ and $c \to^* b$.
  - If $a \to c$, by sub-confluence, there exists d such that $b \to^= d$ and $c \to^= d$.
  - If $a \to c'$ and $c' \to^* c$, by sub-confluence, we have some d' such that $b \to^= d'$ and $c' \to^= d'$. If $c' = d'$ we have $b \to^* c$ and $c \to^* c$. If $c' \to d'$, by induction hypothesis, there exists d such that $d' \to^* d$ and $c \to^* d$ (thus $b \to^* d$).
- If $a \to^* a'$ and $a' \to b$, by induction hypothesis we have d' such that $a' \to^* d'$ and $c \to^* d'$. By the case above, there exists d such that $b \to^* d$ and $d' \to^* d$. We then conclude with $c \to^* d$.

- If $a \to b$ and $a \to c$, confluence gives some d such that $b \to^* d$ and $c \to^* d$.
- If $a \to^* b$ and $a \to^* c$ with b and c normal forms, confluence gives some d such that $b \to^* d$ and $c \to^* d$. But since b and c are normal forms, we must have $b = d$ and $c = d$. □

B.3 Normalization

An ARS $(A, \to)$ is *weakly normalizing* if for any a in A there exists a normal form b in A such that $a \to^* b$ (b is a reduct of a).

An ARS $(A, \to)$ is *strongly normalizing* if it induces no infinite reduction sequence. That is if the relation $\leftarrow$ is well founded.

Lemma B.1 (Transitive Strong Normalization)
If $(A, \to)$ is strongly normalizing then $(A, \to^+)$ is strongly normalizing.

Proof: If there is an infinite reduction sequence for $\to^+$, by expanding the definition of $\to^+$, one obtains an infinite reduction sequence for $\to$. □

An ARS $(A, \to)$ is *μ-decreasing* if μ is a function from A to a set with a well founded relation $<$ such that whenever $a \to b$, we have $\mu(a) > \mu(b)$.

An ARS $(A, \to)$ is *μ-increasing* if μ is a function from A to $\mathbb{N}$ such that whenever $a \to b$, we have $\mu(a) < \mu(b)$.

An ARS $(A, \to)$ is *weakly μ-decreasing* if μ is a function from A to a set with a well founded relation $<$ such that, for any a in A which is not a normal form, there exists some b in A such that $a \to b$ and $\mu(a) > \mu(b)$.

Proposition B.2 (Normalization Properties)
For any ARS, μ-decreasing $\Longrightarrow$ strong normalization $\Longrightarrow$ weakly μ-decreasing $\Longrightarrow$ weak normalization.

Proof: Let $\mathcal{A} = (A, \to)$ be an ARS.

- Assume, by contradiction, there exists an infinite reduction sequence $(a_k)_{0 \leq k < \infty}$ in $\mathcal{A}$. Then $(\mu(a_k))_{0 \leq k < \infty}$ is an infinite decreasing sequence with respect to a well founded relation, a contradiction.
- Since $\mathcal{A}$ is strongly normalizing, it is *id*-decreasing where *id* is the identity function. If a is not a normal form, let b be any 1-step reduct of a, we have $id(a) \to id(b)$.
- Given an a in A, by induction on $k \in \mathbb{N}$, we build the following sequence: $a_0 = a$ and, if a_k is not a normal form, a_{k+1} is such that $a_k \to a_{k+1}$ and $\mu(a_k) > \mu(a_{k+1})$. Since this sequence cannot be infinite (otherwise $(\mu(a_k))_{0 \leq k < \infty}$ is an infinite decreasing sequence), we reach an element a_K which is a normal form and such that $a = a_0 \to^* a_K$. This is a use of the axiom of dependent choices. □

Proposition B.3 (Weak Normalization and Confluence)
For any ARS, *weak normalization* $\wedge$ *unique normal form* $\Longrightarrow$ *confluence.*

Proof: If $a \to^* b$ and $a \to^* c$, by weak normalization, there exist two normal forms b' and c' such that $b \to^* b'$ and $c \to^* c'$, thus $a \to^* b'$ and $a \to^* c'$. By uniqueness of the normal form, we have $b' = c'$. $\square$

Proposition B.4 (Newman's Lemma)
For any ARS, *strong normalization* $\wedge$ *local confluence* $\Longrightarrow$ *confluence.*

Proof: Let $\mathcal{A} = (A, \to)$ be a strongly normalizing and locally confluent ARS. Since the relation $\leftarrow$ is well founded, we can reason by induction on it. We prove this way that for any a, $\mathcal{A} \restriction_a$ is confluent. If a is a normal form the result is immediate. Otherwise we assume that for any 1-step reduct a' of a, $\mathcal{A} \restriction_{a'}$ is confluent. Assume $a \to^* b$ and $a \to^* c$. If $a = b$ or $a = c$ the result is immediate. If $a \to b' \to^* b$ and $a \to c' \to^* c$, by local confluence, we have d' such that $b' \to^* d'$ and $c' \to^* d'$. By confluence of $\mathcal{A} \restriction_{b'}$, there exists d'' such that $b \to^* d''$ and $d' \to^* d''$, thus $c' \to^* d''$. By confluence of $\mathcal{A} \restriction_{c'}$, there exists d such that $d'' \to^* d$ and $c \to^* d$, thus $b \to^* d$ and we conclude.

$$
\begin{tikzcd}
 & & & a \arrow[dl] \arrow[dr] & & & \\
 & & b' \arrow[dl, "*"'] \arrow[dr, "*"] & & c' \arrow[dl, "*"'] \arrow[dr, "*"] & & \\
 & b \arrow[dr, "*"'] & & d' \arrow[dl, "*"] & & c \arrow[ddll, "*"] & \\
 & & d'' \arrow[dr, "*"'] & & & & \\
 & & & d & & &
\end{tikzcd}
$$

$\square$

Proposition B.5 (Increasing Normalization)
For any ARS, *local confluence* $\wedge$ μ*-increasing* $\wedge$ *weak normalization* $\Longrightarrow$ *strong normalization.*

Proof: Let $\mathcal{A} = (A, \to)$ be an ARS, we prove by induction on k that $a \to^* b$ with b normal form and $\mu(b) - \mu(a) \leq k$ implies there is no infinite reduction sequence starting from a.

- If a is a normal form, the result is immediate.
- If $k = 0$, a is a normal form.
- If $k > 0$ and $a \neq b$, we can decompose the reduction sequence from a to b into $a \to c \to^* b$. We have $\mu(c) > \mu(a)$ thus $\mu(b) - \mu(c) < k$ with $c \to^* b$ and, by induction hypothesis, there is no infinite reduction sequence starting from c.

  Let d be a 1-step reduct of a, by local confluence, there exists some e such that both $c \to^* e$ and $d \to^* e$. We apply Propositions B.4 and B.1 to $\mathcal{A} \restriction_c$ to deduce the unique normal form property for the reducts of c. By weak normalization, let f be a normal from of e, we necessarily have $f = b$ (unique normal form of c) thus $d \to^* b$, $\mu(b) - \mu(d) < k$ (since $\mu(d) > \mu(a)$) and, by induction hypothesis, there is no infinite reduction sequence starting from d.

$$
\begin{tikzcd}
a \arrow[r] \arrow[d] & c \arrow[r, "*"] \arrow[d, "*"] & b \arrow[d, equal] \arrow[r, "/" marking] & {} \\
d \arrow[r, "*"'] & e \arrow[r, "*"'] & f \arrow[r, "/" marking] & {}
\end{tikzcd}
$$

This means no infinite reduction sequence starts from a 1-step reduct of a, thus no infinite reduction sequence starts from a.

Now, by weak normalization, for any a in A there is some normal form b such that $a \to^* b$, thus there is no infinite reduction sequence starting from a. $\square$

B.4 Simulation

Let $\mathcal{A} = (A, \to_A)$ and $\mathcal{B} = (B, \to_B)$ be two ARSs, a function φ from A to B is a *simulation* if for every a and a' in A, $a \to_A a'$ entails $\varphi(a) \to_B^* \varphi(a')$. It is a *strict simulation* if $a \to_A a'$ entails $\varphi(a) \to_B^+ \varphi(a')$.

Proposition B.6 (Anti Simulation of Strong Normalization)
If φ is a strict simulation from $\mathcal{A}$ to $\mathcal{B}$ and $\mathcal{B}$ is strongly normalizing, then $\mathcal{A}$ is strongly normalizing as well.

Proof: By Lemma B.1, $(B, \to_B^+)$ is strongly normalizing, thus $\leftarrow^+$ is a well founded relation. We can conclude with Proposition B.2 since $\mathcal{A}$ is then φ-decreasing. $\square$

Proposition B.7 (Anti Simulation of Unique Normal Form)
*If φ is a simulation from $\mathcal{A}$ to $\mathcal{B}$ which preserves normal forms (*i.e. *if a is a normal form in $\mathcal{A}$ then $\varphi(a)$ is a normal form in $\mathcal{B}$) and is injective on normal forms (*i.e. *no two different normal forms of $\mathcal{A}$ have the same image through φ), then the unique normal form property for $\mathcal{B}$ entails the unique normal form property for $\mathcal{A}$.*

Proof: Assume b and c are normal forms with $a \to_A^* b$ and $a \to_A^* c$, then $\varphi(a) \to_B^* \varphi(b)$ and $\varphi(a) \to_B^* \varphi(c)$ with $\varphi(b)$ and $\varphi(c)$ normal forms. This entails $\varphi(b) = \varphi(c)$ by unique normal form for $\mathcal{B}$, and finally $b = c$ since φ is injective on normal forms. $\square$

B.5 Commutation

In this section, we consider two ARSs $\mathcal{A} = (A, \to_{\mathcal{A}})$ and $\mathcal{B} = (A, \to_{\mathcal{B}})$ on the same set A.

The ARS $\mathcal{A} \bowtie \mathcal{B}$ is defined as $(A, \to_{\mathcal{A}\bowtie\mathcal{B}})$ with $\to_{\mathcal{A}\bowtie\mathcal{B}} = \to_{\mathcal{A}} \cup \to_{\mathcal{B}}$. Note that $\to_{\mathcal{A}\bowtie\mathcal{B}}^* = (\to_{\mathcal{A}}^* \cup \to_{\mathcal{B}}^*)^*$

We say that $\mathcal{A}$ and $\mathcal{B}$ *sub-commute* if for any a, b and c in A such that $a \to_{\mathcal{A}} b$ and $a \to_{\mathcal{B}} c$, there exists d such that $b \to_{\mathcal{B}}^= d$ and $c \to_{\mathcal{A}}^= d$. Thus diagrammatically:

$$\begin{tikzcd} & a \arrow[dl, "\mathcal{A}"'] \arrow[dr, "\mathcal{B}"] & \\ b \arrow[dr, dotted, "\mathcal{B}"', "="] & & c \arrow[dl, dotted, "\mathcal{A}", "="'] \\ & d & \end{tikzcd}$$

With this definition, an ARS is sub-confluent if it sub-commutes with itself.

We say that $\mathcal{A}$ and $\mathcal{B}$ *locally commute* if for any a, b and c in A such that $a \to_{\mathcal{A}} b$ and $a \to_{\mathcal{B}} c$, there exists d such that $b \to_{\mathcal{B}}^* d$ and $c \to_{\mathcal{A}}^* d$. Thus diagrammatically:

$$\begin{tikzcd} & a \arrow[dl, "\mathcal{A}"'] \arrow[dr, "\mathcal{B}"] & \\ b \arrow[dr, dotted, "\mathcal{B}"', "*"] & & c \arrow[dl, dotted, "\mathcal{A}", "*"'] \\ & d & \end{tikzcd}$$

With this definition, an ARS is locally confluent if it locally commutes with itself.

We say that $\mathcal{A}$ *quasi-commutes over* $\mathcal{B}$, if for any a, b and c in A such that $a \to_{\mathcal{A}} b$ and $b \to_{\mathcal{B}} c$, there exists d such that $a \to_{\mathcal{B}} d$ and $d \to^*_{\mathcal{A} \bowtie \mathcal{B}} c$. Thus diagrammatically:

$$\begin{tikzcd} & a \arrow[dl, "\mathcal{A}"'] \arrow[dr, dotted, "\mathcal{B}"] & \\ b \arrow[dr, "\mathcal{B}"'] & & d \arrow[dl, dotted, "\mathcal{A}\bowtie\mathcal{B}", "*"'] \\ & c & \end{tikzcd}$$

Lemma B.2 (Transitive Quasi-Commutation)
If $\mathcal{A} = (A, \to_{\mathcal{A}})$ quasi-commutes over $\mathcal{B} = (A, \to_{\mathcal{B}})$ then $\mathcal{A}^* = (A, \to^*_{\mathcal{A}})$ quasi-commutes over $\mathcal{B}$.

Proof: We prove, by induction on k, that $a \to^k_{\mathcal{A}} b$ and $b \to_{\mathcal{B}} c$ implies there exists d such that $a \to_{\mathcal{B}} d$ and $d \to^*_{\mathcal{A} \bowtie \mathcal{B}} c$.

- If $k = 0$, $a = b$ and we can choose $d = c$.
- If $k > 0$, we have $a \to^{k-1}_{\mathcal{A}} a' \to_{\mathcal{A}} b$. By quasi-commutation of $\mathcal{A}$ over $\mathcal{B}$, there exists some d' such that $a' \to_{\mathcal{B}} d'$ and $d' \to^*_{\mathcal{A} \bowtie \mathcal{B}} c$. By induction hypothesis, there exists some d such that $a \to_{\mathcal{B}} d$ and $d \to^*_{\mathcal{A} \bowtie \mathcal{B}} d'$, thus $d \to^*_{\mathcal{A} \bowtie \mathcal{B}} c$. □

Proposition B.8 (Commutation of Strong Normalization)
If $\mathcal{A} = (A, \to_{\mathcal{A}})$ and $\mathcal{B} = (A, \to_{\mathcal{B}})$ are two ARSs, and $\mathcal{A}$ quasi-commutes over $\mathcal{B}$ then if $\mathcal{A}$ and $\mathcal{B}$ are strongly normalizing then $\mathcal{A} \bowtie \mathcal{B}$ is strongly normalizing.

Proof: By contradiction, we assume to have an infinite reduction sequence $(a_k)_{0 \leq k < \infty}$ for $\to_{\mathcal{A} \bowtie \mathcal{B}}$. Since $\mathcal{A}$ is strongly normalizing, for any k, there exists $k' \geq k$ such that $a_{k'} \to_{\mathcal{B}} a_{k'+1}$. By induction on $N \geq 1$, we build a finite reduction sequence $(b_n)_{0 \leq n < N}$ for $\to_{\mathcal{B}}$, such that there exists N' such that $b_{N-1} \to^*_{\mathcal{A} \bowtie \mathcal{B}} a_{N'}$.

- If $N = 1$, $b_0 = a_0$.
- If $N > 1$, we have already built $(b_n)_{0 \leq n < N-1}$ by induction hypothesis, and there exists N'' such that $b_{N-2} \to^*_{\mathcal{A} \bowtie \mathcal{B}} a_{N''}$. There exists $N''' \geq N''$ such that $a_{N'''} \to_{\mathcal{B}} a_{N'''+1}$, thus by decomposing $b_{N-2} \to^*_{\mathcal{A} \bowtie \mathcal{B}} a_{N''} \to^*_{\mathcal{A} \bowtie \mathcal{B}} a_{N'''} \to_{\mathcal{B}} a_{N'''+1}$, we can find some c and d such that $b_{N-2} \to^*_{\mathcal{A}} c \to_{\mathcal{B}} d \to^*_{\mathcal{A} \bowtie \mathcal{B}} a_{N'''+1}$. By Lemma B.2, there exists b_{N-1} such that $b_{N-2} \to_{\mathcal{B}} b_{N-1}$ and $b_{N-1} \to^*_{\mathcal{A} \bowtie \mathcal{B}} d$, thus $b_{N-1} \to^*_{\mathcal{A} \bowtie \mathcal{B}} a_{N'''+1}$ (and we choose $N' = N''' + 1$).

By the axiom of dependent choices, we can build an infinite reduction sequence for $\to_{\mathcal{B}}$ which contradicts the strong normalization of $\mathcal{B}$. □

References

[BCST96] Richard Blute, Robin Cockett, Robert Seely, and Todd Trimble. Natural deduction and coherence for weakly distributive categories. *Journal of Pure and Applied Algebra*, 113:229–296, 1996.

[BdW95] Gianluigi Bellin and Jacques Van de Wiele. Subnets of proof-nets in MLL−. In Jean-Yves Girard, Yves Lafont, and Laurent Regnier, editors, *Advances in Linear Logic*, volume 222 of *London Mathematical Society Lecture Note Series*, pages 249–270. Cambridge University Press, 1995.

[Dan90] Vincent Danos. *La Logique Linéaire appliquée à l'étude de divers processus de normalisation (principalement du λ-calcul)*. Thèse de doctorat, Université Paris VII, 1990.

[DCKP03] Roberto Di Cosmo, Delia Kesner, and Emmanuel Polonovski. Proof nets and explicit substitutions. *Mathematical Structures in Computer Science*, 13(3):409–450, June 2003.

[DR89] Vincent Danos and Laurent Regnier. The structure of multiplicatives. *Archive for Mathematical Logic*, 28:181–203, 1989.

[Gir87] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[Gir91] Jean-Yves Girard. Quantifiers in linear logic II. In Corsi and Sambin, editors, *Nuovi problemi della logica e della filosofia della scienza*, pages 79–90, Bologna, 1991. CLUEB.

[Gir96] Jean-Yves Girard. Proof-nets: the parallel syntax for proof-theory. In Aldo Ursini and Paolo Agliano, editors, *Logic and Algebra*, volume 180 of *Lecture Notes In Pure and Applied Mathematics*, pages 97–124, New York, 1996. Marcel Dekker.

[Gue11] Stefano Guerrini. A linear algorithm for MLL proof net correctness and sequentialization. *Theoretical Computer Science*, 412(20):1958–1978, 2011.

[Hug13] Dominic Hughes. Simple multiplicative proof nets with units. *Annals of Pure and Applied Logic*, 2013. To appear. Available at `http://arxiv.org/abs/math/0507003`.

[HvG05] Dominic Hughes and Rob van Glabbeek. Proof nets for unit-free multiplicative-additive linear logic. *ACM Transactions on Computational Logic*, 6(4):784–842, 2005.

[Lau13] Olivier Laurent. Sequentialization of multiplicative proof nets. Unpublished note. Available at `http://perso.ens-lyon.fr/olivier.laurent/seqmll.pdf`, April 2013.

[PTdF10] Michele Pagani and Lorenzo Tortora de Falco. Strong normalization property for second order linear logic. *Theoretical Computer Science*, 411(2):410–444, 2010.

[Reg94] Laurent Regnier. Une équivalence sur les lambda-termes. *Theoretical Computer Science*, 126:281–292, 1994.

[Ter03] Terese. *Term Rewriting Systems*, volume 55 of *Cambridge tracts in theoretical computer science*. Cambridge University Press, 2003.