Cut Elimination for Proof Nets of the Purely **M**ultiplicative and **A**dditive Fragment of **L**inear **L**ogic

Roberto Maieli

Università degli Studi "Roma Tre"
maieli@uniroma3.it

joint work with Olivier Laurent
(CNRS, Paris)

AILA, XXIII Incontro di Logica
Genova, 20th February 2008

Linear Logic and Proof Theory

- Since its inception, in 1987, linear logic (LL, Girard) has changed the proof theoretic way of dealing with cut elimination.

Linear Logic and Proof Theory

- Since its inception, in 1987, linear logic (LL, Girard) has changed the proof theoretic way of dealing with cut elimination.
- This task was traditionally carried out by means of sequent calculi with the consequence that the most part of these works were engrossed by tedious commutations of rules.

Sequent Calculus: MALL

Sequent Calculus: MALL

- *Formulas* $A, B, \ldots$ are built from *literals* by the binary connectives $\otimes$ (*tensor*), $\parr$ (*par*), $\&$ (*with*) and $\oplus$ (*plus*).

Sequent Calculus: MALL

- *Formulas* $A, B, \ldots$ are built from *literals* by the binary connectives $\otimes$ (*tensor*), $\parr$ (*par*), $\&$ (*with*) and $\oplus$ (*plus*).
- *Negation* $(.)^\perp$ extends to any formula by de Morgan laws:

$$(A \otimes B)^\perp = (B^\perp \parr A^\perp) \qquad (A \parr B)^\perp = (B^\perp \otimes A^\perp)$$
$$(A \& B)^\perp = (B^\perp \oplus A^\perp) \qquad (A \oplus B)^\perp = (B^\perp \& A^\perp)$$

Sequent Calculus: MALL

- *Formulas* $A, B, \ldots$ are built from *literals* by the binary connectives $\otimes$ (*tensor*), $⅋$ (*par*), $\&$ (*with*) and $\oplus$ (*plus*).
- *Negation* $(.)^{\perp}$ extends to any formula by de Morgan laws:

$$(A \otimes B)^{\perp} = (B^{\perp} ⅋ A^{\perp}) \qquad (A ⅋ B)^{\perp} = (B^{\perp} \otimes A^{\perp})$$
$$(A \& B)^{\perp} = (B^{\perp} \oplus A^{\perp}) \qquad (A \oplus B)^{\perp} = (B^{\perp} \& A^{\perp})$$

- *Sequents* Γ, Δ are sets of formula occurrences $A_1, \ldots, A_{n \geq 1}$, proved using the following rules (we omit $\vdash$):

Sequent Calculus: MALL

- *Formulas* $A, B, \ldots$ are built from *literals* by the binary connectives $\otimes$ (*tensor*), $\parr$ (*par*), $\&$ (*with*) and $\oplus$ (*plus*).
- *Negation* $(.)^\perp$ extends to any formula by de Morgan laws:

$$(A \otimes B)^\perp = (B^\perp \parr A^\perp) \qquad (A \parr B)^\perp = (B^\perp \otimes A^\perp)$$
$$(A \& B)^\perp = (B^\perp \oplus A^\perp) \qquad (A \oplus B)^\perp = (B^\perp \& A^\perp)$$

- *Sequents* Γ, Δ are sets of formula occurrences $A_1, \ldots, A_{n \geq 1}$, proved using the following rules (we omit $\vdash$):
  - identity:

$$\frac{}{A, A^\perp}\,\text{ax} \qquad \frac{\Gamma, A \qquad \Delta, A^\perp}{\Gamma, \Delta}\,\text{cut}$$

Sequent Calculus: MALL

- *Formulas* $A, B, \ldots$ are built from *literals* by the binary connectives $\otimes$ (*tensor*), $\parr$ (*par*), $\&$ (*with*) and $\oplus$ (*plus*).
- *Negation* $(.)^{\perp}$ extends to any formula by de Morgan laws:

$$\begin{array}{ll} (A \otimes B)^{\perp} = (B^{\perp} \parr A^{\perp}) & (A \parr B)^{\perp} = (B^{\perp} \otimes A^{\perp}) \\ (A \& B)^{\perp} = (B^{\perp} \oplus A^{\perp}) & (A \oplus B)^{\perp} = (B^{\perp} \& A^{\perp}) \end{array}$$

- *Sequents* Γ, Δ are sets of formula occurrences $A_1, \ldots, A_{n \geq 1}$, proved using the following rules (we omit $\vdash$):
  - identity: $\dfrac{}{A, A^{\perp}}\ \mathrm{ax}$ $\qquad$ $\dfrac{\Gamma, A \qquad \Delta, A^{\perp}}{\Gamma, \Delta}\ \mathrm{cut}$
  - multiplicatives: $\dfrac{\Gamma, A \qquad \Delta, B}{\Gamma, \Delta, A \otimes B}\ \otimes$ $\qquad$ $\dfrac{\Gamma, A, B}{\Gamma, A \parr B}\ \parr$

Sequent Calculus: MALL

- *Formulas* $A, B, \ldots$ are built from *literals* by the binary connectives $\otimes$ (*tensor*), $⅋$ (*par*), $\&$ (*with*) and $\oplus$ (*plus*).
- *Negation* $(.)^\perp$ extends to any formula by de Morgan laws:

$$(A \otimes B)^\perp = (B^\perp ⅋ A^\perp) \qquad (A ⅋ B)^\perp = (B^\perp \otimes A^\perp)$$
$$(A \& B)^\perp = (B^\perp \oplus A^\perp) \qquad (A \oplus B)^\perp = (B^\perp \& A^\perp)$$

- *Sequents* Γ, Δ are sets of formula occurrences $A_1, \ldots, A_{n \geq 1}$, proved using the following rules (we omit $\vdash$):
  - identity: $\dfrac{}{A, A^\perp}\,\text{ax} \qquad \dfrac{\Gamma, A \qquad \Delta, A^\perp}{\Gamma, \Delta}\,\text{cut}$
  - multiplicatives: $\dfrac{\Gamma, A \qquad \Delta, B}{\Gamma, \Delta, A \otimes B}\,\otimes \qquad \dfrac{\Gamma, A, B}{\Gamma, A ⅋ B}\,⅋$
  - additives: $\dfrac{\Gamma, A \qquad \Gamma, B}{\Gamma, A \& B}\,\& \qquad \dfrac{\Gamma, A}{\Gamma, A \oplus B}\,\oplus_1 \qquad \dfrac{\Gamma, B}{\Gamma, A \oplus B}\,\oplus_2$

Sequent Calculus: cut elimination is "problematic"

$$\dfrac{\dfrac{A, A^\perp \qquad A, A^\perp}{A \& A, {\color{red}A^\perp}}\,\& \qquad \dfrac{A, A^\perp}{{\color{red}A}, A^\perp \oplus B^\perp}\,\oplus_1}{A \& A, A^\perp \oplus B^\perp}\,{\color{blue}cut}$$

Sequent Calculus: cut elimination is "problematic"

$$\dfrac{\dfrac{A, A^\perp \qquad A, A^\perp}{A \& A, {\color{red}A^\perp}}\,\& \qquad \dfrac{A, A^\perp}{{\color{red}A}, A^\perp \oplus B^\perp}\,\oplus_1}{A \& A, A^\perp \oplus B^\perp}\,{\color{blue}cut}$$

reduces to:

$$\dfrac{\dfrac{A, {\color{red}A^\perp} \qquad \dfrac{A, A^\perp}{{\color{red}A}, A^\perp \oplus B^\perp}\,\oplus_1}{A, A^\perp \oplus B^\perp}\,{\color{blue}cut} \qquad \dfrac{A, {\color{red}A^\perp} \qquad \dfrac{A, A^\perp}{{\color{red}A}, A^\perp \oplus B^\perp}\,\oplus_1}{A, A^\perp \oplus B^\perp}\,{\color{blue}cut}}{A \& A, A^\perp \oplus B^\perp}\,\&$$

Sequent Calculus: cut elimination is “problematic”

$$\dfrac{\dfrac{A, A^{\perp} \qquad A, A^{\perp}}{A \& A, A^{\perp}}\,\& \qquad \dfrac{A, A^{\perp}}{A, A^{\perp} \oplus B^{\perp}}\,\oplus_1}{A \& A, A^{\perp} \oplus B^{\perp}}\,\mathit{cut}$$

reduces to:

$$\dfrac{\dfrac{A, A^{\perp} \qquad \dfrac{A, A^{\perp}}{A, A^{\perp} \oplus B^{\perp}}\,\oplus_1}{A, A^{\perp} \oplus B^{\perp}}\,\mathit{cut} \qquad \dfrac{A, A^{\perp} \qquad \dfrac{A, A^{\perp}}{A, A^{\perp} \oplus B^{\perp}}\,\oplus_1}{A, A^{\perp} \oplus B^{\perp}}\,\mathit{cut}}{A \& A, A^{\perp} \oplus B^{\perp}}\,\&$$

or to:

$$\dfrac{\dfrac{\dfrac{A, A^{\perp} \qquad A, A^{\perp}}{A \& A, A^{\perp}}\,\& \qquad A, A^{\perp}}{A \& A, A^{\perp}}\,\mathit{cut}}{A \& A, A^{\perp} \oplus B^{\perp}}\,\oplus_1$$

Sequent Calculus: cut elimination is “problematic”

$$\dfrac{\dfrac{A, A^\perp \qquad A, A^\perp}{A\&A, A^\perp}\,\& \qquad \dfrac{A, A^\perp}{A, A^\perp \oplus B^\perp}\,\oplus_1}{A\&A, A^\perp \oplus B^\perp}\,\mathit{cut}$$

reduces to:

$$\dfrac{\dfrac{A, A^\perp \qquad \dfrac{A, A^\perp}{A, A^\perp \oplus B^\perp}\,\oplus_1}{A, A^\perp \oplus B^\perp}\,\mathit{cut} \qquad \dfrac{A, A^\perp \qquad \dfrac{A, A^\perp}{A, A^\perp \oplus B^\perp}\,\oplus_1}{A, A^\perp \oplus B^\perp}\,\mathit{cut}}{A\&A, A^\perp \oplus B^\perp}\,\&$$

or to:

$$\dfrac{\dfrac{\dfrac{A, A^\perp \qquad A, A^\perp}{A\&A, A^\perp}\,\& \qquad A, A^\perp}{A\&A, A^\perp}\,\mathit{cut}}{A\&A, A^\perp \oplus B^\perp}\,\oplus_1$$

This situation has changed with the new geometrical syntax for proofs:

proof nets

Proof Nets: the idea

Proof Nets: the idea

- PNs are parallel presentations of sequential proofs

Proof Nets: the idea

- PNs are parallel presentations of sequential proofs
- PNs quotient classes of equivalent proofs, modulo permutations of derivation rules:

Proof Nets: the idea

- PNs are parallel presentations of sequential proofs
- PNs quotient classes of equivalent proofs, modulo permutations of derivation rules:

$$\dfrac{\dfrac{\dfrac{A, A^\perp}{A, B^\perp \oplus A^\perp}\oplus_2 \quad \dfrac{B, B^\perp}{B, B^\perp \oplus A^\perp}\oplus_1}{A \& B, B^\perp \oplus A^\perp}\& \qquad \dfrac{C, C^\perp}{C \parr C^\perp}\parr}{A \& B, (B^\perp \oplus A^\perp) \otimes (C \parr C^\perp)}\otimes$$

Proof Nets: the idea

- PNs are parallel presentations of sequential proofs
- PNs quotient classes of equivalent proofs, modulo permutations of derivation rules:

$$\dfrac{\dfrac{\dfrac{A, A^{\perp}}{A, B^{\perp} \oplus A^{\perp}}\oplus_2 \qquad \dfrac{B, B^{\perp}}{B, B^{\perp} \oplus A^{\perp}}\oplus_1}{A \& B, B^{\perp} \oplus A^{\perp}}\& \qquad \dfrac{C, C^{\perp}}{C \parr C^{\perp}}\parr}{A \& B, (B^{\perp} \oplus A^{\perp}) \otimes (C \parr C^{\perp})}\otimes$$

$$\dfrac{\dfrac{\dfrac{A, A^{\perp}}{A, B^{\perp} \oplus A^{\perp}}\oplus_2 \qquad \dfrac{C, C^{\perp}}{C \parr C^{\perp}}\parr}{A, (B^{\perp} \oplus A^{\perp}) \otimes (C \parr C^{\perp})}\otimes \qquad \dfrac{\dfrac{B, B^{\perp}}{B, B^{\perp} \oplus A^{\perp}}\oplus_1 \qquad \dfrac{C, C^{\perp}}{C \parr C^{\perp}}\parr}{B, (B^{\perp} \oplus A^{\perp}) \otimes (C \parr C^{\perp})}\otimes}{A \& B, (B^{\perp} \oplus A^{\perp}) \otimes (C \parr C^{\perp})}\&$$

Proof Nets: state of the art

PNs of the Multiplicative fragment of LL are the perfect setting:

Proof Nets: state of the art

PNs of the Multiplicative fragment of LL are the perfect setting:

1. a proof net is a canonical representative of a proof of the sequent calculus,

Proof Nets: state of the art

PNs of the Multiplicative fragment of LL are the perfect setting:

1. a proof net is a canonical representative of a proof of the sequent calculus,
2. the cut elimination procedure is purely local: the reduction of a cut is given by only modifying the nodes connected to it.

Proof Nets: state of the art

PNs of the Multiplicative fragment of LL are the perfect setting:

1. a proof net is a canonical representative of a proof of the sequent calculus,
2. the cut elimination procedure is purely local: the reduction of a cut is given by only modifying the nodes connected to it.

A lot of work has been done to extend (1) and (2) to MALL.

Proof Nets: state of the art

PNs of the Multiplicative fragment of LL are the perfect setting:

1. a proof net is a canonical representative of a proof of the sequent calculus,
2. the cut elimination procedure is purely local: the reduction of a cut is given by only modifying the nodes connected to it.

A lot of work has been done to extend (1) and (2) to MALL.

In 1996, Girard proposed a new syntax for MALL PNs:

Proof Nets: state of the art

PNs of the Multiplicative fragment of LL are the perfect setting:

1. a proof net is a canonical representative of a proof of the sequent calculus,
2. the cut elimination procedure is purely local: the reduction of a cut is given by only modifying the nodes connected to it.

A lot of work has been done to extend (1) and (2) to MALL.

In 1996, Girard proposed a new syntax for MALL PNs:

- without additive boxes (sequentiality)

Proof Nets: state of the art

PNs of the Multiplicative fragment of LL are the perfect setting:

1. a proof net is a canonical representative of a proof of the sequent calculus,
2. the cut elimination procedure is purely local: the reduction of a cut is given by only modifying the nodes connected to it.

A lot of work has been done to extend (1) and (2) to MALL.

In 1996, Girard proposed a new syntax for MALL PNs:

- without additive boxes (sequentiality)
- allowing super-positions (slices)

Proof Nets: state of the art (continues)

... but Girard's proposal was not as good as for MLL:

Proof Nets: state of the art (continues)

... but Girard's proposal was not as good as for MLL:

1. w.r.t. canonicity of the representation: there exist proofs which de-sequentialize into two possible proof nets with no way to discriminate them.

Proof Nets: state of the art (continues)

... but Girard's proposal was not as good as for MLL:

1. w.r.t. canonicity of the representation: there exist proofs which de-sequentialize into two possible proof nets with no way to discriminate them.
   This problem has been solved by D. Hughes and R. van Glabbeek (2003)

Proof Nets: state of the art (continues)

... but Girard's proposal was not as good as for MLL:

1. w.r.t. canonicity of the representation: there exist proofs which de-sequentialize into two possible proof nets with no way to discriminate them.
   This problem has been solved by D. Hughes and R. van Glabbeek (2003)
2. Girard's cut elimination is:

Proof Nets: state of the art (continues)

... but Girard's proposal was not as good as for MLL:

1. w.r.t. canonicity of the representation: there exist proofs which de-sequentialize into two possible proof nets with no way to discriminate them.
   This problem has been solved by D. Hughes and R. van Glabbeek (2003)
2. Girard's cut elimination is:
   – not local;

Proof Nets: state of the art (continues)

... but Girard's proposal was not as good as for MLL:

1. w.r.t. canonicity of the representation: there exist proofs which de-sequentialize into two possible proof nets with no way to discriminate them.
   This problem has been solved by D. Hughes and R. van Glabbeek (2003)
2. Girard's cut elimination is:
   – not local;
   – not confluent;

Proof Nets: state of the art (continues)

... but Girard's proposal was not as good as for MLL:

1. w.r.t. canonicity of the representation: there exist proofs which de-sequentialize into two possible proof nets with no way to discriminate them.
   This problem has been solved by D. Hughes and R. van Glabbeek (2003)
2. Girard's cut elimination is:
   – not local;
   – not confluent;
   – lazy one: only cuts not involving additive contractions (ready cuts) are reducible.

Proof Nets: state of the art (continues)

... but Girard's proposal was not as good as for MLL:

1. w.r.t. canonicity of the representation: there exist proofs which de-sequentialize into two possible proof nets with no way to discriminate them.
   This problem has been solved by D. Hughes and R. van Glabbeek (2003)
2. Girard's cut elimination is:
   – not local;
   – not confluent;
   – lazy one: only cuts not involving additive contractions (ready cuts) are reducible.
   As a consequence, not all proof-nets are normalizable.

Proof Nets: state of the art (continues)

... but Girard's proposal was not as good as for MLL:

1. w.r.t. canonicity of the representation: there exist proofs which de-sequentialize into two possible proof nets with no way to discriminate them.
   This problem has been solved by D. Hughes and R. van Glabbeek (2003)
2. Girard's cut elimination is:
   – not local;
   – not confluent;
   – lazy one: only cuts not involving additive contractions (ready cuts) are reducible.
   As a consequence, not all proof-nets are normalizable.

Our goal here is:

Proof Nets: state of the art (continues)

... but Girard's proposal was not as good as for MLL:

1. w.r.t. canonicity of the representation: there exist proofs which de-sequentialize into two possible proof nets with no way to discriminate them.
   This problem has been solved by D. Hughes and R. van Glabbeek (2003)
2. Girard's cut elimination is:
   – not local;
   – not confluent;
   – lazy one: only cuts not involving additive contractions (ready cuts) are reducible.
   As a consequence, not all proof-nets are normalizable.

Our goal here is:

– to provide an answer to the locality of the cut elimination.

Proof Nets: state of the art (continues)

... but Girard's proposal was not as good as for MLL:

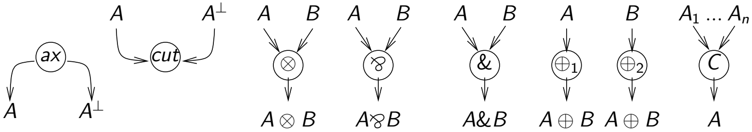
1. w.r.t. canonicity of the representation: there exist proofs which de-sequentialize into two possible proof nets with no way to discriminate them.
   This problem has been solved by D. Hughes and R. van Glabbeek (2003)
2. Girard's cut elimination is:
   – not local;
   – not confluent;
   – lazy one: only cuts not involving additive contractions (ready cuts) are reducible.
   As a consequence, not all proof-nets are normalizable.

Our goal here is:
– to provide an answer to the locality of the cut elimination.
– to allow a new kind of sharing nodes which neither exists in JYG nor in HvG

MALL Proof Structures: links

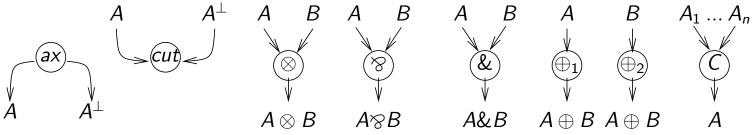
MALL Proof Structures: links

- A **(pre-)proof structure** π is an oriented graph such that each edge is labelled by a MALL formula and built on the set of following nodes ($A = A_1 = ... = A_n$ in the C node).



MALL Proof Structures: links

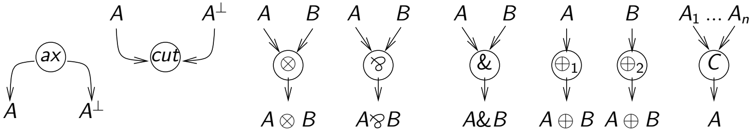
- A **(pre-)proof structure** π is an oriented graph such that each edge is labelled by a MALL formula and built on the set of following nodes ($A = A_1 = ... = A_n$ in the C node).



- fixed a node, an entering edge is called **premise** while its (possibly) emergent edges are called *conclusions*.

MALL Proof Structures: links

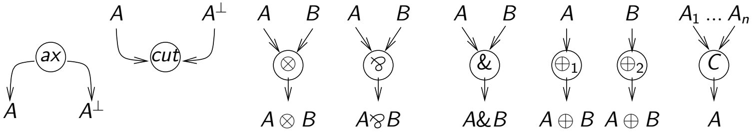
- A **(pre-)proof structure** π is an oriented graph such that each edge is labelled by a MALL formula and built on the set of following nodes ($A = A_1 = \ldots = A_n$ in the C node).



- fixed a node, an entering edge is called **premise** while its (possibly) emergent edges are called *conclusions*.
- pending edges are called **conclusions** of π

MALL Proof Structures: links

- A **(pre-)proof structure** π is an oriented graph such that each edge is labelled by a MALL formula and built on the set of following nodes ($A = A_1 = ... = A_n$ in the C node).



- fixed a node, an entering edge is called **premise** while its (possibly) emergent edges are called *conclusions*.
- pending edges are called **conclusions** of π
- a **link** is the graph made by a node together with its premise(s) and its (possibly) conclusion(s).

MALL Proof Structures: weights

MALL Proof Structures: weights

- a set of Boolean variables denoted by $p, q, \ldots,$

MALL Proof Structures: weights

- a set of Boolean variables denoted by $p, q, \ldots$,
- a **monomial weight** $w, v, \ldots$ is a product "." (conjunction) of variables or negation of variables.

MALL Proof Structures: weights

- a set of Boolean variables denoted by $p, q, \ldots$,
- a **monomial weight** $w, v, \ldots$ is a product "." (conjunction) of variables or negation of variables.
- ϵ_p, for a variable p or its negation $\overline{p}$;

MALL Proof Structures: weights

- a set of Boolean variables denoted by $p, q, \ldots$,
- a **monomial weight** $w, v, \ldots$ is a product "." (conjunction) of variables or negation of variables.
- ϵ_p, for a variable p or its negation $\overline{p}$;
- 1, for the empty product;

MALL Proof Structures: weights

- a set of Boolean variables denoted by $p, q, \ldots$,
- a **monomial weight** $w, v, \ldots$ is a product "." (conjunction) of variables or negation of variables.
- ϵ_p, for a variable p or its negation $\overline{p}$;
- 1, for the empty product;
- 0, for a product where both p and $\bar{p}$ appear;

MALL Proof Structures: weights

- a set of Boolean variables denoted by $p, q, \ldots$,
- a **monomial weight** $w, v, \ldots$ is a product "." (conjunction) of variables or negation of variables.
- ϵ_p, for a variable p or its negation $\overline{p}$;
- 1, for the empty product;
- 0, for a product where both p and $\bar{p}$ appear;
- two weights, v and w, are **disjoint** when $v.w = 0$.

MALL Proof Structures: weights

- a set of Boolean variables denoted by $p, q, \ldots$,
- a **monomial weight** $w, v, \ldots$ is a product "." (conjunction) of variables or negation of variables.
- ϵ_p, for a variable p or its negation $\overline{p}$;
- 1, for the empty product;
- 0, for a product where both p and $\bar{p}$ appear;
- two weights, v and w, are **disjoint** when $v.w = 0$.
- the special notation $\epsilon_p @ v$ when we want to focus on the **prefix weight** ϵ_p within a weight $\epsilon_p.v$ (v is the **suffix**).

MALL Proof Structures: weights

- a set of Boolean variables denoted by $p, q, \ldots$,
- a **monomial weight** $w, v, \ldots$ is a product "." (conjunction) of variables or negation of variables.
- ϵ_p, for a variable p or its negation $\overline{p}$;
- 1, for the empty product;
- 0, for a product where both p and $\bar{p}$ appear;
- two weights, v and w, are **disjoint** when $v.w = 0$.
- the special notation $\epsilon_p@v$ when we want to focus on the **prefix weight** ϵ_p within a weight $\epsilon_p.v$ (v is the **suffix**).
- a weight w **depends on a variable** p when ϵ_p appears in w;

MALL Proof Structures: basic definition

A MALL **proof structure** (PS), is a pair $\langle \pi, E \rangle$ where:

MALL Proof Structures: basic definition

A MALL **proof structure** (PS), is a pair $\langle \pi, E \rangle$ where:

- $E = \{\epsilon_p @ w = 0 \mid \epsilon_p$ is a prefix $\wedge\ w$ is a weight ϵ_p-free$\}$;

MALL Proof Structures: basic definition

A MALL **proof structure** (PS), is a pair $\langle \pi, E \rangle$ where:

- $E = \{\epsilon_p @ w = 0 \mid \epsilon_p$ is a prefix $\wedge$ w is a weight ϵ_p-free$\}$;
- π is a pre-proof structure s.t.:

MALL Proof Structures: basic definition

A MALL **proof structure** (PS), is a pair $\langle \pi, E \rangle$ where:

- $E = \{\epsilon_p @ w = 0 \mid \epsilon_p$ is a prefix $\wedge$ w is a weight ϵ_p-free$\}$;
- π is a pre-proof structure s.t.:
  1. we associate a Boolean variable p, called *eigen weight*, to each & node of π ($\&_p$ notation);

MALL Proof Structures: basic definition

A MALL **proof structure** (PS), is a pair $\langle \pi, E \rangle$ where:

- $E = \{\epsilon_p @ w = 0 \mid \epsilon_p$ is a prefix $\wedge\ w$ is a weight ϵ_p-free$\}$;
- π is a pre-proof structure s.t.:
  1. we associate a Boolean variable p, called *eigen weight*, to each & node of π ($\&_p$ notation);
     eigen weights are not supposed to be different;

MALL Proof Structures: basic definition

A MALL **proof structure** (PS), is a pair $\langle \pi, E \rangle$ where:

- $E = \{\epsilon_p @ w = 0 \mid \epsilon_p$ is a prefix $\wedge$ w is a weight ϵ_p-free$\}$;
- π is a pre-proof structure s.t.:
  1. we associate a Boolean variable p, called *eigen weight*, to each & node of π ($\&_p$ notation);
     eigen weights are not supposed to be different;
  2. to each node we associate a nonzero weight w of the Boolean algebra generated over the set of variables that are:

MALL Proof Structures: basic definition

A MALL **proof structure** (PS), is a pair $\langle \pi, E \rangle$ where:

- $E = \{\epsilon_p @ w = 0 \mid \epsilon_p$ is a prefix $\wedge\ w$ is a weight ϵ_p-free$\}$;
- π is a pre-proof structure s.t.:
  1. we associate a Boolean variable p, called *eigen weight*, to each & node of π ($\&_p$ notation);
     eigen weights are not supposed to be different;
  2. to each node we associate a nonzero weight w of the Boolean algebra generated over the set of variables that are:
     - eigen weights occurring in π or

MALL Proof Structures: basic definition

A MALL **proof structure** (PS), is a pair $\langle \pi, E \rangle$ where:

- $E = \{\epsilon_p @ w = 0 \mid \epsilon_p$ is a prefix $\wedge$ w is a weight ϵ_p-free$\}$;
- π is a pre-proof structure s.t.:
  1. we associate a Boolean variable p, called *eigen weight*, to each & node of π ($\&_p$ notation);
     eigen weights are not supposed to be different;
  2. to each node we associate a nonzero weight w of the Boolean algebra generated over the set of variables that are:
     - eigen weights occurring in π or
     - prefix weights of the equations in E

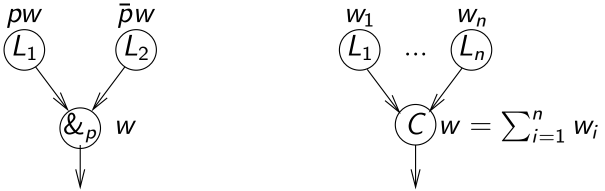
MALL Proof Structures: basic definition

A MALL **proof structure** (PS), is a pair $\langle \pi, E \rangle$ where:

- $E = \{\epsilon_p @ w = 0 \mid \epsilon_p$ is a prefix $\wedge\ w$ is a weight ϵ_p-free$\}$;
- π is a pre-proof structure s.t.:
  1. we associate a Boolean variable p, called *eigen weight*, to each & node of π ($\&_p$ notation);
     eigen weights are not supposed to be different;
  2. to each node we associate a nonzero weight w of the Boolean algebra generated over the set of variables that are:
     - eigen weights occurring in π or
     - prefix weights of the equations in E
  3. all weights are *modulo* E;

MALL Proof Structures: weights assignment

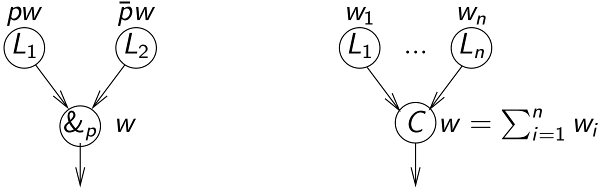
3. two nodes have the same weight, if they have a common edge, except when the edge is the premise of a & or C node:



ϵ_p does not occur in w $\quad$ $\forall i \forall j, w_i w_j = 0 \; (1 \leq i, j \leq n)$

MALL Proof Structures: weights assignment

5. two nodes have the same weight, if they have a common edge, except when the edge is the premise of a & or C node:



ϵ_p does not occur in w $\quad$ $\forall i \forall j, w_i w_j = 0 \; (1 \le i, j \le n)$

6. every conclusion node has weight 1;

MALL Proof Structures: technical condition

7. if w is a weight depending on p and s.t.

MALL Proof Structures: technical condition

7. if w is a weight depending on p and s.t.
   - it belongs to a node of π, or

MALL Proof Structures: technical condition

7. if w is a weight depending on p and s.t.
   - it belongs to a node of π, or
   - it occurs in an equation of E

MALL Proof Structures: technical condition

7. if w is a weight depending on p and s.t.
   - it belongs to a node of π, or
   - it occurs in an equation of E

   then

$$w \leq (\sum_{i=1}^{n} w_i + \sum_{j=1}^{m} v_j) \mod E \tag{1}$$

   where :

MALL Proof Structures: technical condition

7. if w is a weight depending on p and s.t.
   - it belongs to a node of π, or
   - it occurs in an equation of E

   then

$$w \leq (\sum_{i=1}^{n} w_i + \sum_{j=1}^{m} v_j) \mod E \quad (1)$$

   where :

   - w_i, $1 \leq i \leq n$, is the weight of a node $\&_p$;

MALL Proof Structures: technical condition

7. if w is a weight depending on p and s.t.
   - it belongs to a node of π, or
   - it occurs in an equation of E

   then

$$w \leq (\sum_{i=1}^{n} w_i + \sum_{j=1}^{m} v_j) \mod E \quad (1)$$

   where :

   - w_i, $1 \leq i \leq n$, is the weight of a node $\&_p$;
   - v_j, $1 \leq j \leq m$, is the suffix of an equation $\epsilon_p @ v_j = 0$ of E;

MALL Proof Structures: technical condition

7. if w is a weight depending on p and s.t.
   - it belongs to a node of π, or
   - it occurs in an equation of E

   then

$$w \leq (\sum_{i=1}^{n} w_i + \sum_{j=1}^{m} v_j) \mod E \quad (1)$$

   where :

   - w_i, $1 \leq i \leq n$, is the weight of a node $\&_p$;
   - v_j, $1 \leq j \leq m$, is the suffix of an equation $\epsilon_p @ v_j = 0$ of E;
   - $(\sum_{i=1}^{n} w_i + \sum_{j=1}^{m} v_j)$ is a monomial weight $\mod E$;

MALL Proof Structures: technical condition

7. if w is a weight depending on p and s.t.
   - it belongs to a node of π, or
   - it occurs in an equation of E

   then

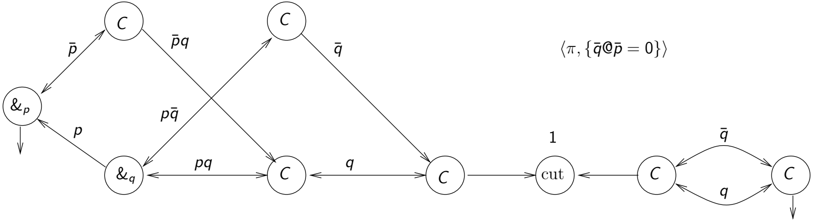
$$w \leq (\sum_{i=1}^{n} w_i + \sum_{j=1}^{m} v_j) \mod E \qquad (1)$$

   where :

   - w_i, $1 \leq i \leq n$, is the weight of a node $\&_p$;
   - v_j, $1 \leq j \leq m$, is the suffix of an equation $\epsilon_p @ v_j = 0$ of E;
   - $(\sum_{i=1}^{n} w_i + \sum_{j=1}^{m} v_j)$ is a monomial weight $\mod E$;
   - all weights $w_1, ..., w_n, v_1, ...v_m$ are pairwise disjoint.

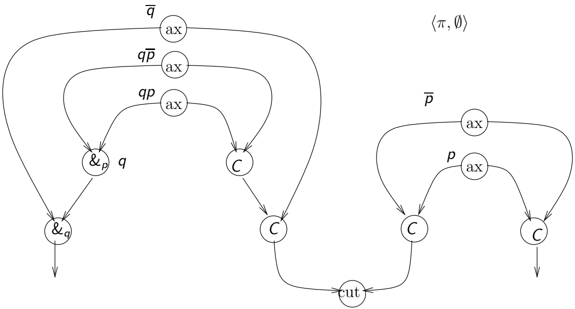
MALL Proof Structures: example 1

The following pair $\langle \pi, \{(\bar{q}@\bar{p} = 0)\}\rangle$ is a proof structure:



MALL Proof Structures: example 2

The following pair $\langle \pi, \emptyset \rangle$ is not a proof structure



it violates the *technical condition* of PS definition: there exists a (axiom) node whose weight is $\bar{p}$ but $\bar{p} \not\leq q$, where q is the weight of the unique $\&_p$ node.

The Request of a Correctness Criterion

The Request of a Correctness Criterion

- we are interested on those proof structures that correspond to proofs of the sequent calculus;

The Request of a Correctness Criterion

- we are interested on those proof structures that correspond to proofs of the sequent calculus;
- those proof structures will be called **proof nets**

The Request of a Correctness Criterion

- we are interested on those proof structures that correspond to proofs of the sequent calculus;
- those proof structures will be called **proof nets**
- there exists a **Correction Crietrion** that detect these PNs

The Request of a Correctness Criterion

- we are interested on those proof structures that correspond to proofs of the sequent calculus;
- those proof structures will be called **proof nets**
- there exists a **Correction Crietrion** that detect these PNs
- cut elimination can be defined directly on PSs

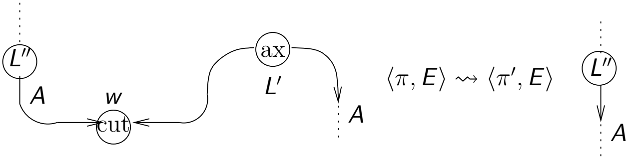
The Request of a Correctness Criterion

- we are interested on those proof structures that correspond to proofs of the sequent calculus;
- those proof structures will be called **proof nets**
- there exists a **Correction Crietrion** that detect these PNs
- cut elimination can be defined directly on PSs
- then we have to show that the Correction Crietrion is preserved by the cut elimination

Cut Elimination

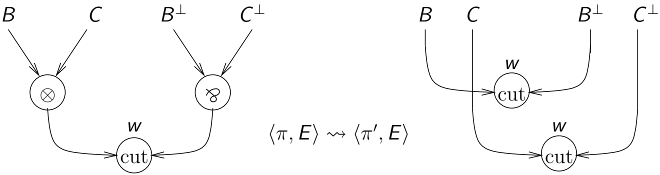
Cut Elimination: *ax*-step

If L' (resp., L'') is an axiom node of π, then $\langle \pi, E \rangle \rightsquigarrow \langle \pi', E \rangle$, where π' is obtained by removing in π both formulas A and $A^{\perp}$ (as well as L) and giving a new conclusion to L'' (resp., L'), the other conclusion of L' (resp., L'')



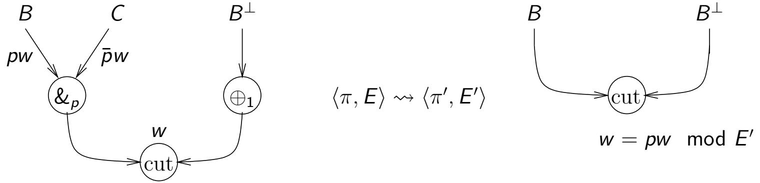
Cut Elimination: ($\otimes/\parr$)-step

If L' is a $\otimes$ node with premises B and C and L'' is a $\parr$ node with premises $B^\perp$ and $C^\perp$, then $\langle \pi, E \rangle \leadsto \langle \pi', E \rangle$, where π' is obtained by removing in π the formulas A and $A^\perp$ as well as the cut node L with L' and L'' and adding two new cut nodes with premises, respectively, $B, B^\perp$ and $C, C^\perp$



Cut Elimination: $(\oplus_i/\&)$-step

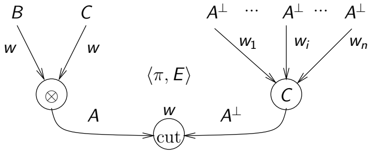
If L' is a $\&_p$ node with weight w and B and C as premises whose weights are, respectively, pw and $\bar{p}w$, and L'' is a $\oplus_1$ node with premise $B^\perp$ in π, then $\langle \pi, E\rangle \rightsquigarrow \langle \pi', E'\rangle$ as below



- $E' = E \cup \{\bar{p}@w = 0\}$;
- π' is what remains still nonzero, $\mod E'$, w.r.t. π.

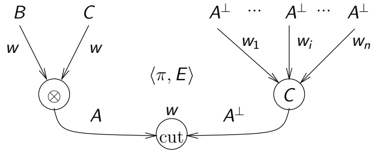
Cut Elimination: $(\otimes/C)$-step

If L' is a C node and L'' is a $\otimes$ node

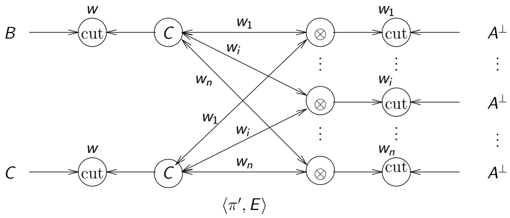


Cut Elimination: $(\otimes/C)$-step

If L' is a C node and L'' is a $\otimes$ node

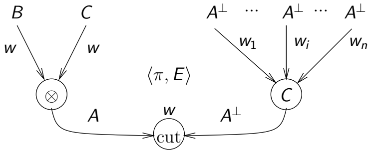


then $\langle \pi, E \rangle \rightsquigarrow \langle \pi', E \rangle$ as follows:

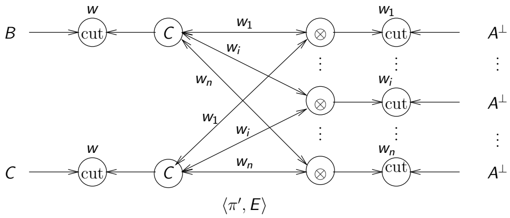


Cut Elimination: $(\otimes/C)$-step

If L' is a C node and L'' is a $\otimes$ node



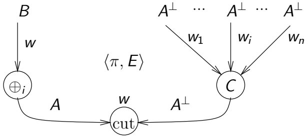
then $\langle \pi, E \rangle \rightsquigarrow \langle \pi', E \rangle$ as follows:



Case $(⅋/C)$-step is analogous (replace $\otimes$s with ⅋s).

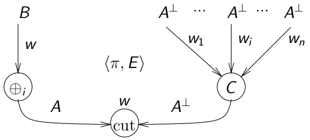
Cut Elimination: $(\oplus_i / C)$-step

If L' is a C node and L'' is a $\oplus_i$ node

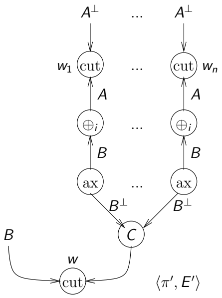


Cut Elimination: $(\oplus_i/C)$-step

If L' is a C node and L'' is a $\oplus_i$ node

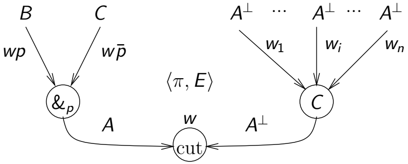


then $\langle\pi, E\rangle \rightsquigarrow \langle\pi', E\rangle$ as follows:



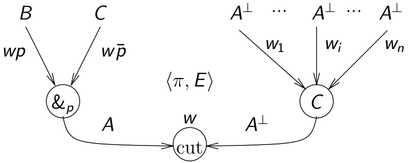
Cut Elimination: $(\&/C)$-step

If L' is a $\&_p$ node and L'' is a C node

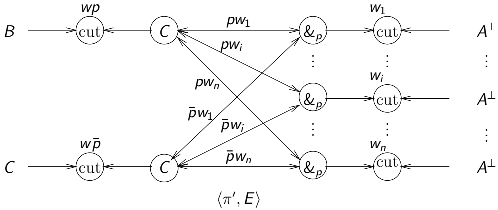


Cut Elimination: $(\&/C)$-step

If L' is a $\&_p$ node and L'' is a C node

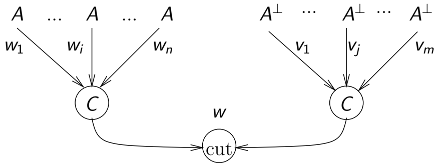


then $\langle \pi, E \rangle \rightsquigarrow \langle \pi', E \rangle$ as follows



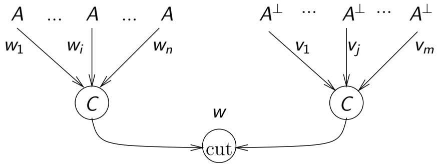
Cut Elimination: (C/C)-step

If both L' and L'' are C nodes

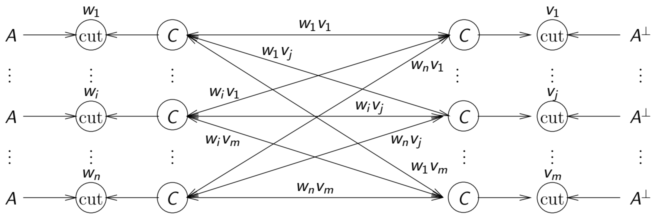


Cut Elimination: (C/C)-step

If both L' and L'' are C nodes



then $\langle \pi, E \rangle \rightsquigarrow \langle \pi', E \rangle$ as follows:



Stability under the Cut Elimination

Stability under the Cut Elimination

Theorem (Stability of PS)

If a PS $\langle \pi, E \rangle$ reduces in one step to $\langle \pi', E' \rangle$, then $\langle \pi', E' \rangle$ is a PS too.

Stability under the Cut Elimination

Theorem (Stability of PS)

If a PS $\langle \pi, E \rangle$ *reduces in one step to* $\langle \pi', E' \rangle$*, then* $\langle \pi', E' \rangle$ *is a PS too.*

Theorem (Stability of Correctness Criterion)

If a PS $\langle \pi, E \rangle$ *is correct and it reduces in one step to* $\langle \pi', E' \rangle$*, then* $\langle \pi', E' \rangle$ *is still a correct PS.*

Strong Cut Elimination

Theorem

We can always strongly reduce a proof net $\langle \pi, E \rangle$ into a proof net $\langle \pi', E' \rangle$ that is cut-free, by iterating the reduction steps.

Strong Cut Elimination

Theorem

We can always strongly reduce a proof net $\langle \pi, E \rangle$ into a proof net $\langle \pi', E' \rangle$ that is cut-free, by iterating the reduction steps.

Proof.

The proof is by lexicographic induction on the *word*

$$\sharp 1, ..., \sharp n$$

Strong Cut Elimination

Theorem

We can always strongly reduce a proof net $\langle \pi, E \rangle$ *into a proof net* $\langle \pi', E' \rangle$ *that is cut-free, by iterating the reduction steps.*

Proof.

The proof is by lexicographic induction on the *word*

$$\sharp 1, ..., \sharp n$$

- n is the number of variables (eigen or prefix weights) of $\langle \pi, E \rangle$;

Strong Cut Elimination

Theorem

We can always strongly reduce a proof net $\langle \pi, E \rangle$ *into a proof net* $\langle \pi', E' \rangle$ *that is cut-free, by iterating the reduction steps.*

Proof.

The proof is by lexicographic induction on the *word*

$$\sharp 1, ..., \sharp n$$

- n is the number of variables (eigen or prefix weights) of $\langle \pi, E \rangle$;
- $\sharp i$, with $1 \leq i \leq n$, is the sum of the logical complexities of all cuts whose *depth* is i.

Strong Cut Elimination

Theorem

We can always strongly reduce a proof net $\langle \pi, E \rangle$ *into a proof net* $\langle \pi', E' \rangle$ *that is cut-free, by iterating the reduction steps.*

Proof.

The proof is by lexicographic induction on the *word*

$$\sharp 1, ..., \sharp n$$

- n is the number of variables (eigen or prefix weights) of $\langle \pi, E \rangle$;
- $\sharp i$, with $1 \leq i \leq n$, is the sum of the logical complexities of all cuts whose *depth* is i.
- the depth $\delta(L)$ of a node L is $\max(|w_1|, |w_2|)$, if

Strong Cut Elimination

Theorem

We can always strongly reduce a proof net $\langle \pi, E \rangle$ *into a proof net* $\langle \pi', E' \rangle$ *that is cut-free, by iterating the reduction steps.*

Proof.

The proof is by lexicographic induction on the *word*

$$\sharp 1, ..., \sharp n$$

- n is the number of variables (eigen or prefix weights) of $\langle \pi, E \rangle$;
- $\sharp i$, with $1 \leq i \leq n$, is the sum of the logical complexities of all cuts whose *depth* is i.
- the depth $\delta(L)$ of a node L is $\max(|w_1|, |w_2|)$, if
  - w_1 and w_2 are equivalent (modulo E) weights of L and

Strong Cut Elimination

Theorem

We can always strongly reduce a proof net $\langle \pi, E \rangle$ into a proof net $\langle \pi', E' \rangle$ that is cut-free, by iterating the reduction steps.

Proof.

The proof is by lexicographic induction on the *word*

$$\sharp 1, ..., \sharp n$$

- n is the number of variables (eigen or prefix weights) of $\langle \pi, E \rangle$;
- $\sharp i$, with $1 \leq i \leq n$, is the sum of the logical complexities of all cuts whose *depth* is i.
- the depth $\delta(L)$ of a node L is $\max(|w_1|, |w_2|)$, if
  - w_1 and w_2 are equivalent (modulo E) weights of L and
  - $|w_j|$, for $j = 1, 2$, is the length of w_j.

□

Confluence

Theorem (local confluence)

Let $\langle \pi, E \rangle$ *be a proof net with two cut nodes,* L_1 *and* L_2*, and let*

Confluence

Theorem (local confluence)

Let $\langle \pi, E \rangle$ *be a proof net with two cut nodes,* L_1 *and* L_2*, and let*

- α *be the cut reduction* $\langle \pi, E \rangle \rightsquigarrow_{L_1} \langle \pi_1, E_1 \rangle$ *and*

Confluence

Theorem (local confluence)

Let $\langle \pi, E \rangle$ be a proof net with two cut nodes, L_1 and L_2, and let

- α *be the cut reduction* $\langle \pi, E \rangle \rightsquigarrow_{L_1} \langle \pi_1, E_1 \rangle$ *and*
- β *be the cut reduction* $\langle \pi, E \rangle \rightsquigarrow_{L_2} \langle \pi_2, E_2 \rangle$,

Confluence

Theorem (local confluence)

Let $\langle \pi, E \rangle$ be a proof net with two cut nodes, L_1 and L_2, and let

- α *be the cut reduction* $\langle \pi, E \rangle \rightsquigarrow_{L_1} \langle \pi_1, E_1 \rangle$ *and*
- β *be the cut reduction* $\langle \pi, E \rangle \rightsquigarrow_{L_2} \langle \pi_2, E_2 \rangle$,

then there exists a proof net $\langle \pi^*, E^* \rangle$ which $\langle \pi_i, E_i \rangle$, for $1 \leq i \leq 2$, reduces to in at most one step.

conclusions