

Geometry of Interaction: Old and New

Thomas Seiller



October 29th, 2013

Oberseminar Logik und Sprachtheorie, Tübingen University

Linear Logic

Studying models of the λ -calculus, Girard realized that the implication $A \rightarrow B$ is decomposed as:

$$!A \multimap B$$

- ▶ $\multimap$ is a *linear* implication: it uses its argument exactly once;
- ▶ $!$ is a modality allowing to use its argument multiple times;

He then introduced linear logic to mirror in the syntax these remarks on the semantics. The restriction to linear implication induces a splitting of the conjunction and disjunction into *multiplicative* ($\otimes, \wp$) and *additive* ($\&, \oplus$) variants.

Example

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

Figure: Rules for Multiplicative Linear Logic

Example

$$\frac{A \vdash B \quad \frac{\frac{\overline{A \vdash A}^{\text{ax}} \quad \frac{\overline{B \vdash B}^{\text{ax}}}{A, B \vdash A \wedge B}^{\wedge}}{A, A \vdash A \wedge B}^{\text{cut}}}{\frac{A \vdash A \wedge B}{}^{\text{ctr}}}^{\rightarrow}}{\vdash A \rightarrow A \wedge B}$$

Example

$$\frac{\displaystyle \frac{A \vdash B}{\displaystyle \frac{\displaystyle \frac{\overline{A \vdash A}^{\text{ax}} \quad \overline{B \vdash B}^{\text{ax}}}{A, B \vdash A \otimes B}^{\otimes}}{A, A \vdash A \otimes B}^{\text{cut}}}{\displaystyle \frac{A \vdash A \otimes B}{\displaystyle \frac{A \vdash A \otimes B}{\vdash A \rightarrow A \otimes B}^{\text{ctr}}}^{\rightarrow}}^{\rightarrow}$$

Example

$$\frac{\frac{A \vdash B}{A, A \vdash A \otimes B} \text{ cut} \quad \frac{\frac{\frac{A \vdash A}{A \vdash A} \text{ ax} \quad \frac{B \vdash B}{B \vdash B} \text{ ax}}{A, B \vdash A \otimes B} \otimes}{\frac{\frac{\frac{A, A \vdash A \otimes B}{!A, !A \vdash A \otimes B} \text{ der}}{A \vdash A \otimes B} \text{ ctr}}{\vdash A \rightarrow A \otimes B} \rightarrow} \text{ ax}$$

Example

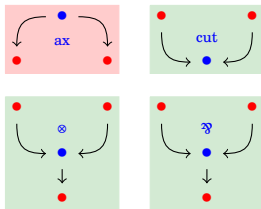
$$\frac{\frac{A \vdash B}{A, A \vdash A \otimes B} \text{ cut} \quad \frac{\frac{\frac{A \vdash A}{A \vdash A} \text{ ax} \quad \frac{B \vdash B}{B \vdash B} \text{ ax}}{A, B \vdash A \otimes B} \otimes}{\frac{\frac{\frac{A, A \vdash A \otimes B}{!A, !A \vdash A \otimes B} \text{ der}}{!A \vdash A \otimes B} \text{ ctr}}{\vdash A \rightarrow A \otimes B} \rightarrow}$$

Example

$$\frac{\frac{A \vdash B \quad \frac{\frac{\overline{A \vdash A}^{\text{ax}} \quad \frac{\overline{B \vdash B}^{\text{ax}}}{A, B \vdash A \otimes B}^{\otimes}}{A, A \vdash A \otimes B}^{\text{cut}}}{\overline{!A, !A \vdash A \otimes B}^{\text{der}}}{\overline{!A \vdash A \otimes B}^{\text{ctr}}}{\vdash !A \multimap A \otimes B}^{\multimap}$$

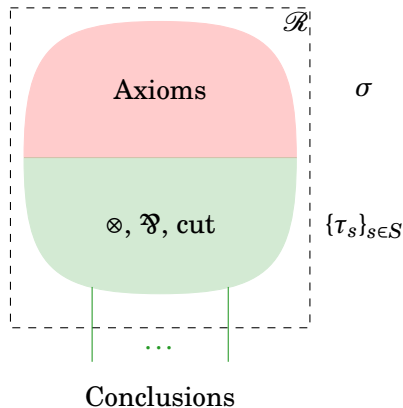
Proof Nets

Proof structures:



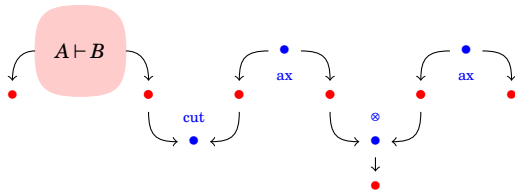
Theorem

$\mathcal{R}$ is sequentializable
iff
 $\forall s \in S, \sigma \tau_s$ is cyclic.



Example

$$\frac{\frac{\vdots}{A \vdash B} \quad \frac{\frac{\overline{A \vdash A}^{\text{ax}} \quad \overline{B \vdash B}^{\text{ax}}}{A, B \vdash A \otimes B} \otimes}{\frac{A, A \vdash A \otimes B}{A \otimes A \vdash A \otimes B} \otimes} \text{cut} \quad \frac{\vdots}{A \vdash B} \quad \frac{\overline{A \vdash A}^{\text{ax}} \quad \overline{B \vdash B}^{\text{ax}}}{A \vdash B \quad B \vdash B} \text{cut} \quad \frac{\frac{A, B \vdash A \otimes B}{A \otimes A \vdash A \otimes B} \otimes}{\vdash A \otimes A \multimap A \otimes B} \multimap$$



Geometry of Interaction: Interpretation of proofs

In a first approximation, the geometry of interaction program aims at defining a semantics of proofs that accounts for the dynamics of cut-elimination.

- ▶ A proof is interpreted by its set of axiom links;
- ▶ Cut-elimination corresponds to the computation of paths in the graph.

A remark on Proof Nets

- ▶ One can notice that the tests for A coming from the correctness criterion are in correspondence with the *proofs* of $A^\perp$.

$$\begin{array}{lcl} \text{Tests for } A & = & \text{Proofs of } A^\perp \\ \text{Proofs of } A & = & \text{Tests for } A^\perp \end{array}$$

Geometry of Interaction: Negation

Geometry of interaction aims at more than a mere interpretation of proofs. It is a complete reconstruction of logic around the dynamics of cut-elimination.

Based on the duality between tests for a A and proofs of $A^\perp$:

- ▶ one considers a set of more general objects (paraproofs) representing both proofs and tests;
- ▶ one defines a notion of orthogonality that translates the correctness criterion.

Geometry of Interaction: Formulas and Connectives

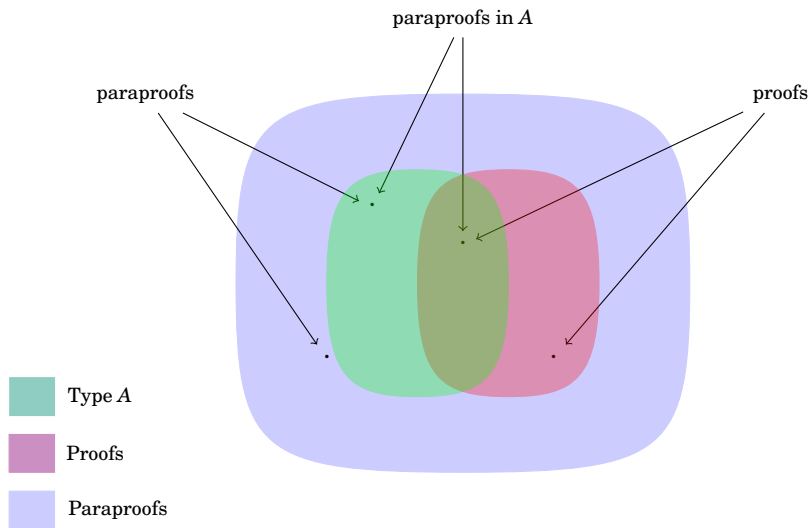
One defines formulas (or types) as sets of paraproofs closed by bi-orthogonality or, equivalently:

Definition

A formula is a set of paraproofs A such that $A = B^\perp$ for B a given set of tests.

Connectives are defined on paraproofs, and this definition induces a construction on types.

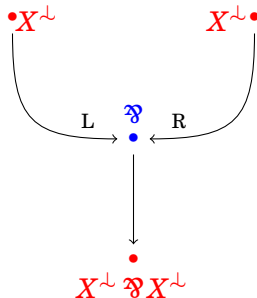
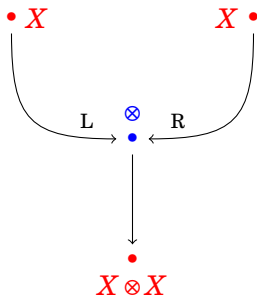
Principle



Remarks on Proof Nets

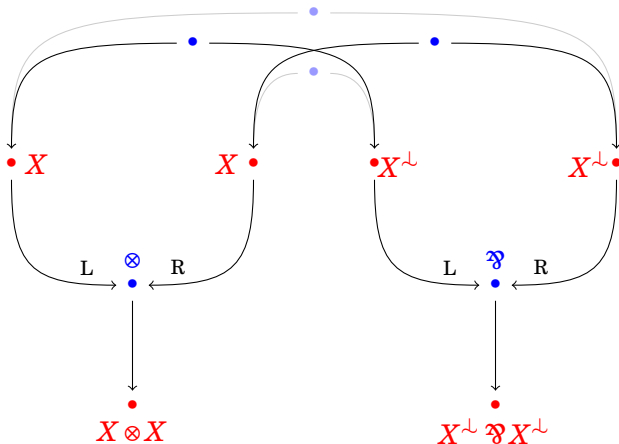
Proofs as sets of axioms

Different proofs of a given formula differ only by their axiom links.



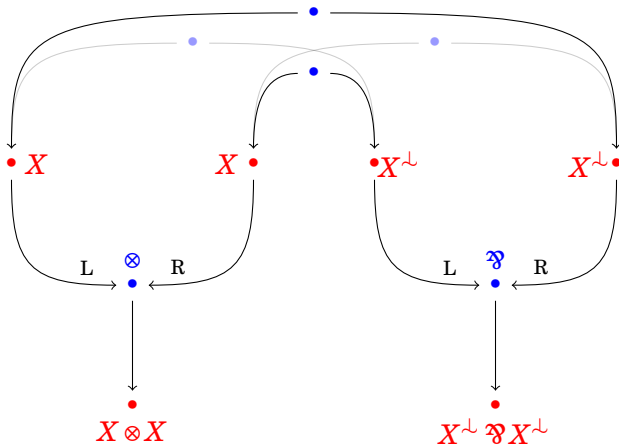
Proofs as sets of axioms

Different proofs of a given formula differ only by their axiom links.



Proofs as sets of axioms

Different proofs of a given formula differ only by their axiom links.



Given a proof structure $\mathcal{R}$, we therefore distinguish between:

- ▶ the axiomatic part of $\mathcal{R}$ — denoted by $\mathcal{R}_{ax}$ — which is the set of axiom bricks of $\mathcal{R}$;
- ▶ the typing part of $\mathcal{R}$ — denoted by $\mathcal{R}_{type}$ — which is the set of $\otimes$, $\wp$ and cut bricks of $\mathcal{R}$;

We will therefore consider a proof structure as a couple of:

- ▶ $\mathcal{R}_{ax}$: an "untyped proof" (like a pure lambda-term);
- ▶ $\mathcal{R}_{type}$: a type (it is a formula tree);

Correctness Criteria

Given a proof structure $\mathcal{R}$:

1. we define a finite set S of objects depending on the typing part $\mathcal{R}_{type}$ of $\mathcal{R}$: trips (Longtrips criterion), graphs (DR);
2. we show that $\mathcal{R}$ is sequentializable if and only if all elements of S interact properly with the axiomatic part $\mathcal{R}_{ax}$ of $\mathcal{R}$: the resulting trip is long (Longtrips), the resulting graph is connected and acyclic (DR).

The correctness criteria can therefore be seen as a way of checking if the untyped proof $\mathcal{R}_{ax}$ can be given the type corresponding to $\mathcal{R}_{type}$.

Cut-Elimination

We can choose the following reduction "strategy" (it is in fact a family of strategies):

- ▶ First eliminate all $\otimes$ - $\wp$ cuts: this step does nothing more than associating (identifying) atoms of the cut formulas;
- ▶ Then eliminate ax - ax cuts: this is where the real computation takes place, the atoms of the cut formulas are erased and sequences of axioms and cuts are replaced by axioms.

Example

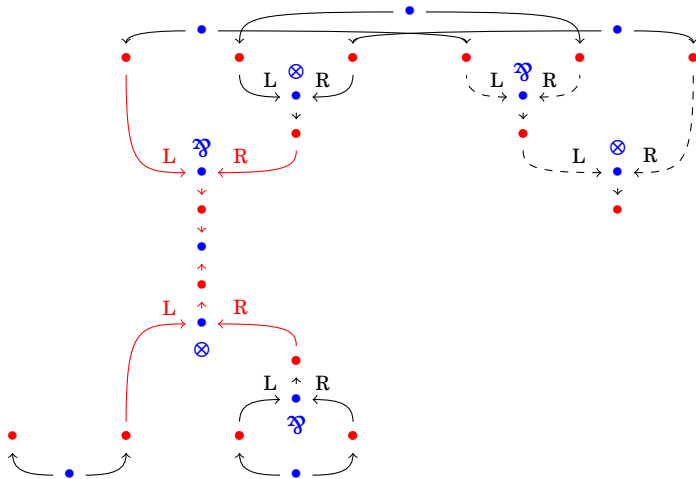


Figure: Example

Example

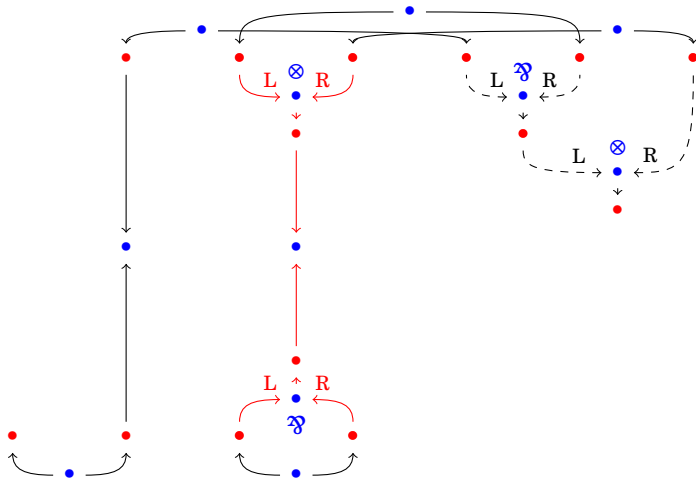


Figure: Example

Example

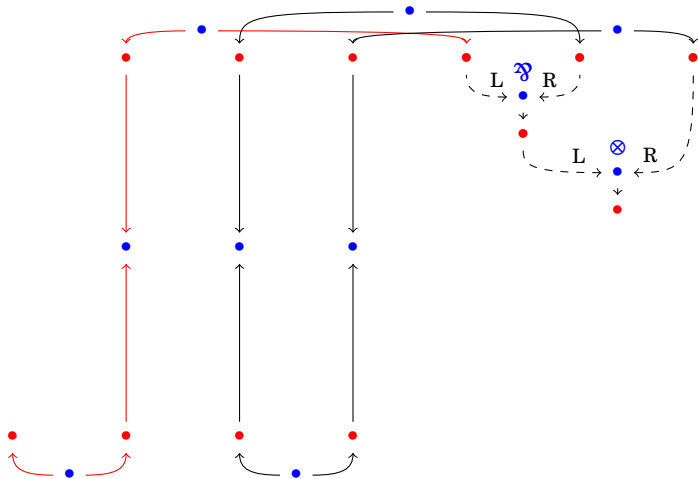


Figure: Example

Example

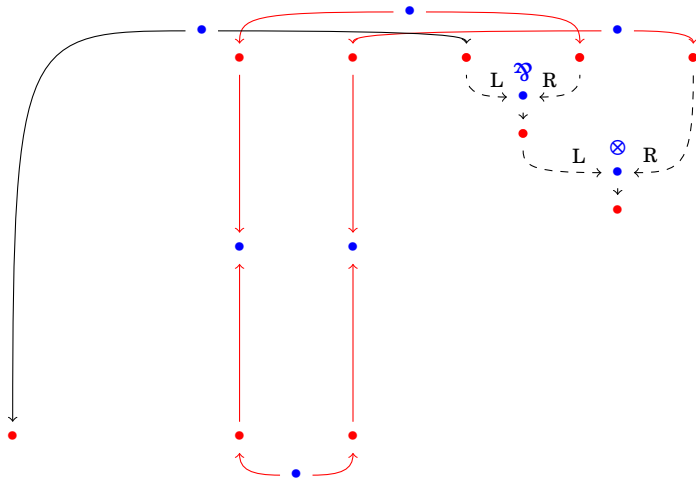


Figure: Example

Example

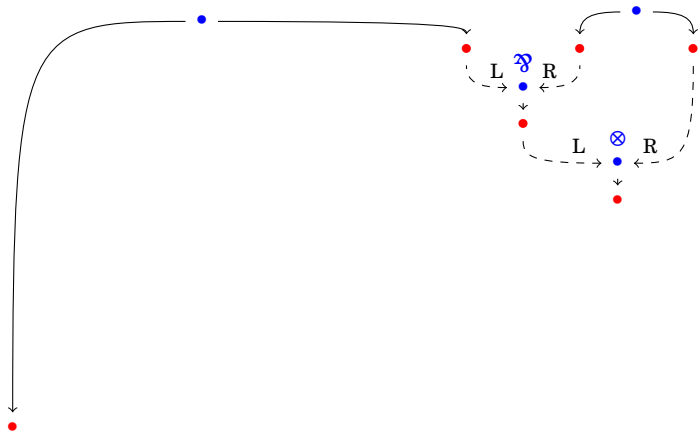


Figure: Example

Paths in Proof Nets



Vincent Danos and Laurent Regnier,
Proof Nets and the Hilbert Space,
in: *Advances in Linear Logic*,
London Math. Soc. Lecture Note Series, vol. 222, CUP 1995.

Paths in proof nets

- ▶ We will be interested in paths in proof nets. For this, we will consider *reverted edges*: if e of source s and target t is an edge in a graph, we define the reverted edge e^* of source t and target s . The label of a reverted edge s^* is the same as the label of the edge e . The edge $(e^*)^*$ is defined to be equal to e .

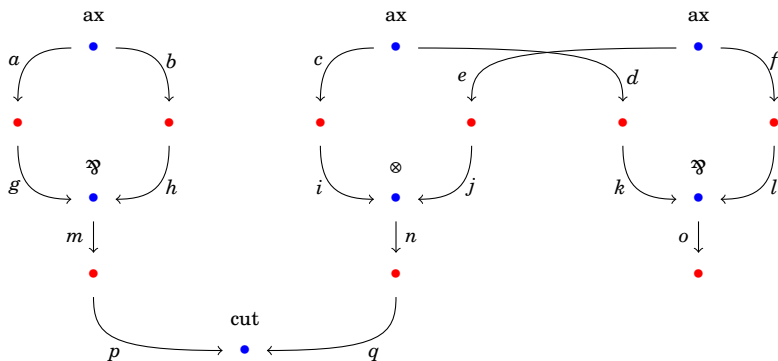
Definition

A path $(e_i)_{i=0}^n$ in a proof net $\mathcal{R}$ is a sequence of edges and reverted edges of $\mathcal{R}$ such that the source of e_i is equal to the target of e_{i+1} .

Remark

*Notice our paths are written **from right to left!***

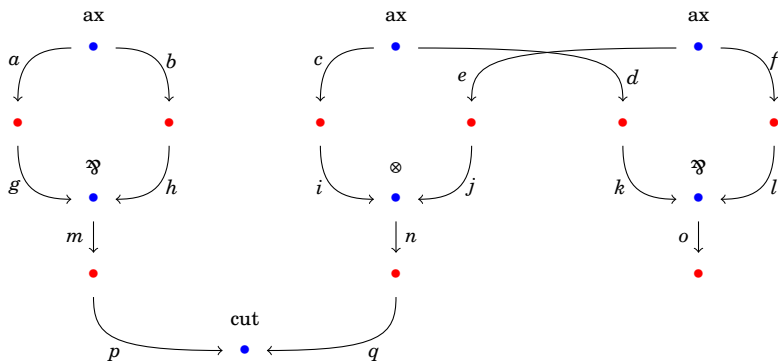
Paths in proof nets



- Path (bouncing):

$$dd^*k^*o^*$$

Paths in proof nets



- Path (twisting):

$$j^* i c d^* k^* o^*$$

Straight paths

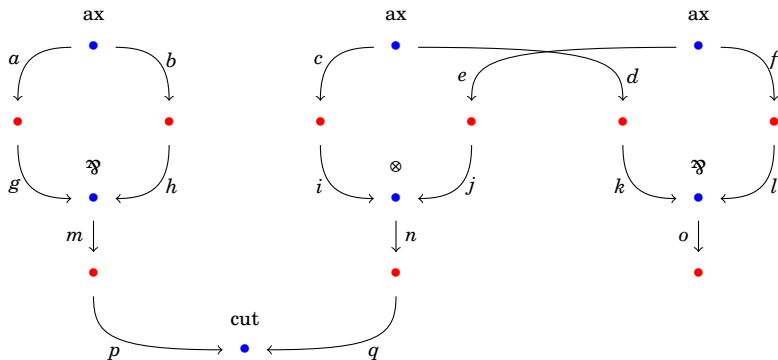
Definition

A path $(e_i)_{i=0}^n$ in $\mathcal{R}$ is *straight* if it is:

- ▶ *non-bouncing*: if for all $0 \leq i \leq n-1$, $e_{i+1} \neq e_i^*$;
- ▶ *non-twisting*: if it does not contain $\alpha^* \beta$ where α, β are distinct premises of a $\otimes$ or $\wp$ vertex;

We define $\mathcal{P}(\mathcal{R})$ to be the set of conclusion-to-conclusion straight paths in $\mathcal{R}$ to which we add a formal 0 element.

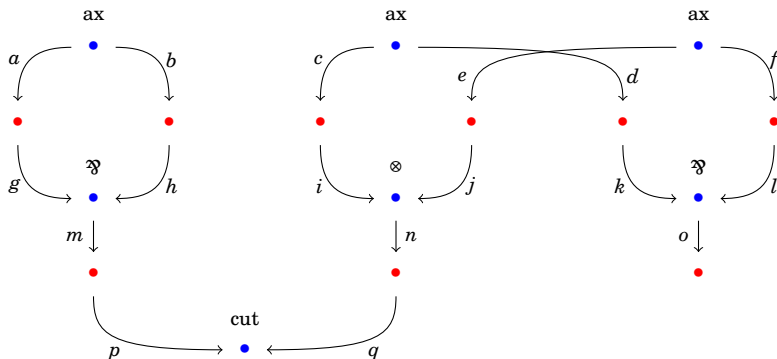
Straight Paths



► Path (straight):

$$lfe^*j^*n^*q^*$$

Straight Paths



- Path (straight, conclusion-to-conclusion):

$$olfe^*j^*n^*q^*pmhba^*g^*m^*p^*qnicd^*k^*o^*$$

Cut-Elimination

We first recall the cut-elimination steps on proof nets in order to introduce notations that will be used in the next paragraph. In our setting of η -expanded MLL proof nets, there are only two cut-eliminations steps, the $ax - ax$ -step and the $\otimes - \wp$ -step.

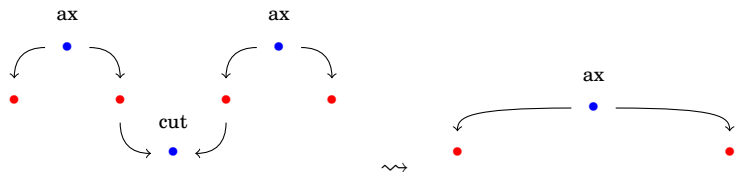


Figure: Cut-elimination: the $ax - ax$ -step

Cut-Elimination

We first recall the cut-elimination steps on proof nets in order to introduce notations that will be used in the next paragraph. In our setting of η -expanded MLL proof nets, there are only two cut-eliminations steps, the $ax - ax$ -step and the $\otimes - \wp$ -step.

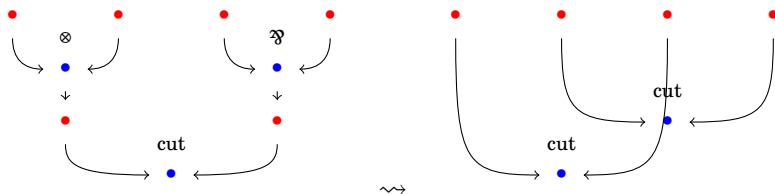
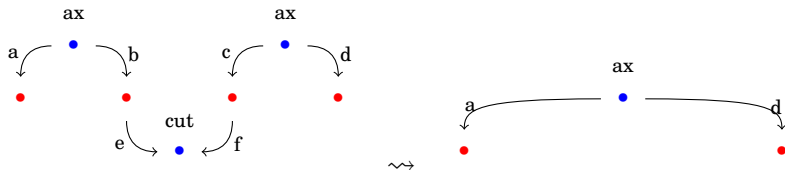


Figure: Cut-elimination: the $\otimes - \wp$ -step

Cut-Elimination II

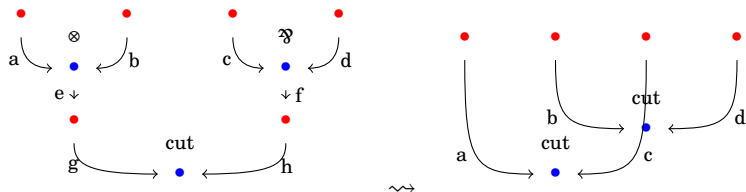
For a cut-elimination step $\rho : \mathcal{R} \rightarrow \mathcal{R}'$, we can define a map φ_ρ from $\mathcal{P}(\mathcal{R})$ to $\mathcal{P}(\mathcal{R}')$ as follows:



- **Ax-Ax steps.** The path $\varphi_\rho(p)$ is the path p where the sequence $ab^*e^*fcd^*$ (resp. $dc^*f^*eba^*$) is replaced by ad^* (resp. da^*).

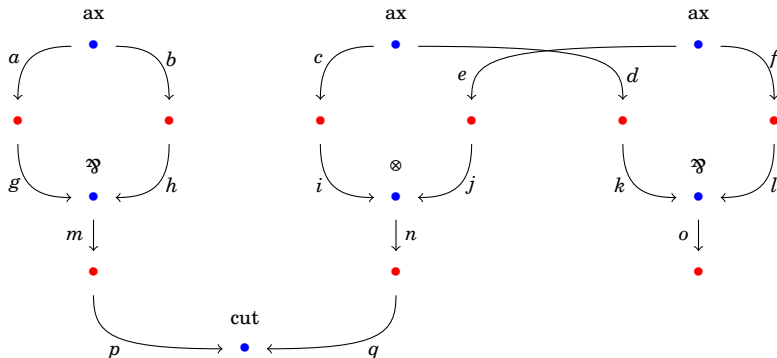
Cut-Elimination II

For a cut-elimination step $\rho : \mathcal{R} \rightarrow \mathcal{R}'$, we can define a map φ_ρ from $\mathcal{P}(\mathcal{R})$ to $\mathcal{P}(\mathcal{R}')$ as follows:



- $\otimes - \nabla$ steps.** The path $\varphi_\rho(p)$ is the path p where the sequences $a^*e^*g^*hfc$, $b^*e^*g^*hfd$, $c^*f^*h^*gea$, and $d^*f^*h^*geb$ are replaced by a^*c , b^*d , c^*a , and d^*b respectively and the sequences $a^*e^*g^*hfd$, $b^*e^*g^*hfc$, $c^*f^*h^*geb$, and $d^*f^*h^*gea$ are replaced by 0 (absorbing element).

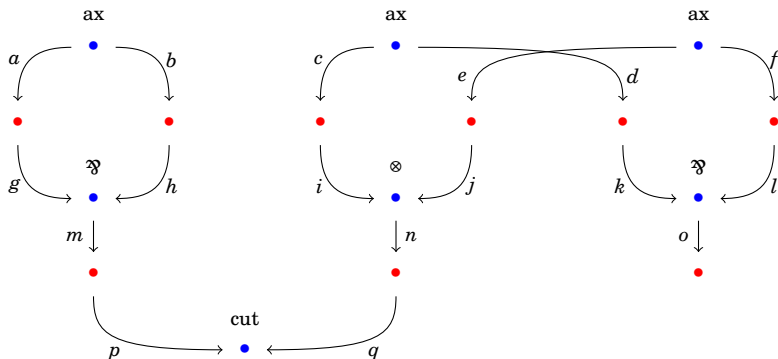
Cut-Elimination



- Path (straight, conclusion-to-conclusion):

$$okdc^*i^*n^*q^*pmhba^*g^*m^*p^*qnjef^*l^*o^*$$

Cut-Elimination



- Path (straight, conclusion-to-conclusion):

$$olfe^*j^*n^*q^*pmhba^*g^*m^*p^*qnicd^*k^*o^*$$

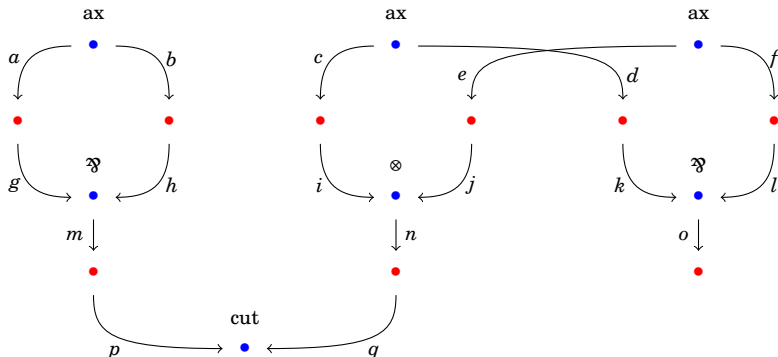
Persistent Paths

- ▶ This defines a map φ from $\mathcal{P}(\mathcal{R})$ to $\mathcal{P}(\mathcal{R}')$ ($\mathcal{R}'$ is the normal form of $\mathcal{R}$).

Definition

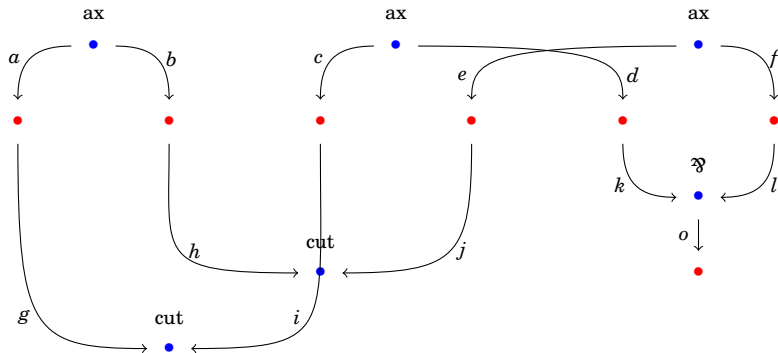
A conclusion-to-conclusion straight path p is called *persistent* if its image by φ is different from 0.

Persistent Paths



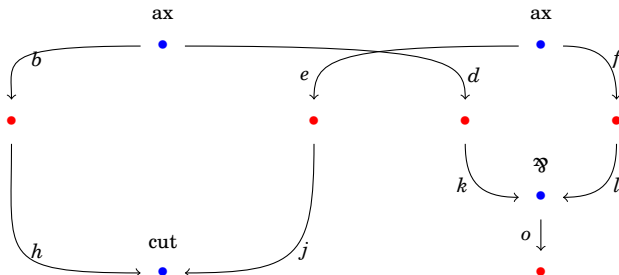
$$\begin{aligned}
 & olfe^* \underbrace{j^* n^* q^* pmhba^*}_{\text{}} \underbrace{g^* m^* p^* qnicd^* k^* o^*}_{\text{}} \\
 & \rightarrow olfe^* j^* hba^* g^* icd^* k^* o^*
 \end{aligned}$$

Persistent Paths



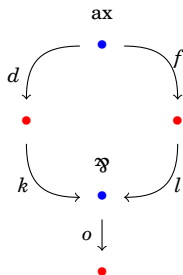
$$\begin{aligned}
 olfe^*j^*n^*q^*pmhba^*g^*m^*p^*qnicd^*k^*o^* &\rightarrow olfe^*j^*hba^*g^*icd^*k^*o^* \\
 olfe^*j^*\underbrace{hba^*g^*icd^*k^*o^*} &\rightarrow olfe^*j^*hbd^*k^*o^*
 \end{aligned}$$

Persistent Paths



$$\begin{aligned}
 olfe^*j^*n^*q^*pmhba^*g^*m^*p^*qnicd^*k^*o^* &\rightarrow olfe^*j^*hbd^*k^*o^* \\
 \underbrace{olfe^*j^*hbd^*k^*o^*} &\rightarrow olfd^*k^*o^*
 \end{aligned}$$

Persistent Paths



$$olfe^*j^*n^*q^*pmhba^*g^*m^*p^*qnicd^*k^*o^* \rightarrow olfd^*k^*o^*$$

The L^* involutive monoid

Definition

Define the free involutive monoid M^* generated by the elements r, l . This is the set of words over the alphabet $\{l, r, l^*, r^*\}$ endowed with the concatenation product and satisfying the equations $(w^*)^* = w$ and $(vw)^* = w^*v^*$.

Definition

We define L^* to be:

- ▶ an involutive monoid with two generators r, l (free as a monoid);
- ▶ with an absorbing element 0 , i.e. $w \cdot 0 = 0 \cdot w = 0$;
- ▶ satisfying the equations $l^*r = 0$ and $l^*l = r^*r = 1$.

Weight of Paths

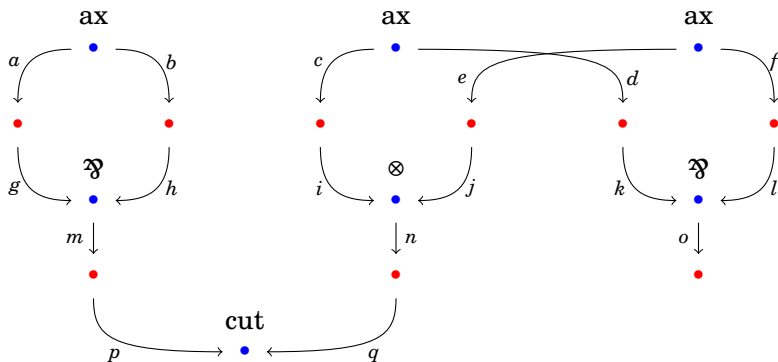
Definition

We define a weight function ω on the set of edges of a proof net $\mathcal{R}$:

- ▶ $\omega(e) = 1$ if e is a A -labelled edge of target $a \otimes$ (resp. $\wp$) vertex whose conclusion is labelled $A \otimes B$ (resp. $A \wp B$);
- ▶ $\omega(e) = r$ if e is a B -labelled edge of target $a \otimes$ (resp. $\wp$) vertex whose conclusion is labelled $A \otimes B$ (resp. $A \wp B$);
- ▶ $\omega(e) = 1$ otherwise.

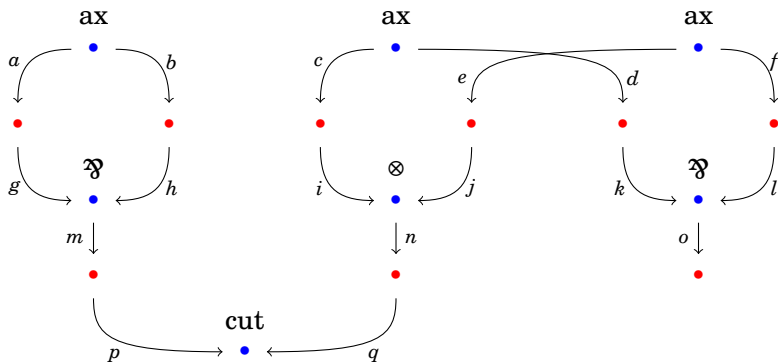
This weight function extends to reversed edges by defining $\omega(e^*) = \omega(e)^*$. It can then be extended to paths by letting $\omega(vw) = \omega(v)\omega(w)$.

Weight of paths



$$\omega(okdc^*i^*n^*q^*pmhba^*g^*m^*p^*qnjef^*l^*o^*) = 11^*r1^*rr^* = 0$$

Weight of paths



$$\omega(olfe^*j^*n^*q^*pmhba^*g^*m^*p^*qnicd^*k^*o^*) = rr^*rl^*ll^* = rl^*$$

Regular paths

Definition

A path p is said *regular* when $\omega(p) \neq 0$.

Theorem

A straight path p is persistent if and only if it is regular. Moreover, if p_1, p_2 are persistent paths with same source and target and such that neither is a subpath of the other, then $\omega(p_2^*p_1) = 0$.

Main Theorems

Definition (The Execution)

Let $\mathcal{R}$ be a proof net. The execution of $\mathcal{R}$ is defined as:

$$\text{Ex}(\mathcal{R}) = \sum_{p \in \mathcal{P}(\mathcal{R})} \omega(p)$$

Theorem

Let $\mathcal{R}$ be a proof net and $\mathcal{R}'$ its normal form. Then:

$$\text{Ex}(\mathcal{R}) = \text{Ex}(\mathcal{R}')$$

Denotational Semantics:

$$\begin{array}{ccc} \mathcal{R} & \xrightarrow{\text{Ex}(\cdot)} & \text{Ex}(\mathcal{R}) \\ \text{CE} \downarrow & & \parallel \\ \mathcal{R}' & \xrightarrow{\text{Ex}(\cdot)} & \text{Ex}(\mathcal{R}') \end{array}$$

GoI: Interpretation of Proofs



Jean-Yves Girard

Geometry of Interaction I: Interpretation of System F
in: [Proceedings of Logic Colloquium 88](#)



Jean-Yves Girard

Geometry of Interaction VI: Blueprint for a Transcendental
Syntax

Regular Paths, Axioms and Cuts

- ▶ A regular (therefore straight) path bounces up and down the net, changing direction only in axioms and cuts;
- ▶ A regular path can therefore be represented as an alternated sequence of axioms and cuts;
- ▶ The GoI interpretation will consist in representing the axioms on one hand and the cuts on another;
- ▶ The regular paths will then be recovered as an algebraic computation: the execution formula.

Addresses

Definition

An *address* in a proof net $\mathcal{R}$ is a sequence $(e_i)_{i=0}^n$ such that $e_0 = p_A$, where A is a conclusion of $\mathcal{R}$, and for all $1 \leq i \leq n$ the element e_i is either equal to r or to l .

Flows I

Definition

A *flow* is an ordered pair of addresses (a, a') written as $a \leftarrow a'$.

Remark

A flow $a \leftarrow a'$ can be identified with an edge of a graph with vertices a, a' .

Definition

Let $a \leftarrow a'$ and $b \leftarrow b'$ be two flows. We define their composition:

$$(a \leftarrow a').(b \leftarrow b') = \begin{cases} a \leftarrow b' & \text{if } a' = b \\ 0 & \text{otherwise} \end{cases}$$

Flows II

- ▶ An *extension* of an address $a = (e_i)_{i=0}^n$ is an address $a.w = (f_i)_{i=0}^{n+m}$ such that $(f_i)_{i=0}^n = a$ and $w = (f_i)_{i=n+1}^{n+m}$ is a word in the free monoid generated by $\{l, r\}$.
- ▶ A *bundle of addresses* is a sequence $(e_i)_{i=0}^n$ such that $a = (e_i)_{i=0}^{n-1}$ is an address and $e_n = \star$. A bundle of addresses represents the union of all extensions of a and will be written as $a.\star$.
- ▶ A *bundle of flows* is an ordered pair $a \leftarrow a'$ where a and a' are bundles of addresses. It represents the union of all the flows $b \leftarrow b'$ where $b = a.w$ and $b' = a'.w'$ are extensions of a and a' such that $w = w'$.

Flows III

This definition can be extended to compose a flow with a sum of flows (and thus to a composition between two sums of flows):

$$(\sum_{i=0}^n a_i \leftarrow b_i).(c \leftarrow c') = \sum_{i=0}^n (a_i \leftarrow b_i).(c \leftarrow c')$$

It is also easily extended to a composition of a flow with a bundle of flows:

$$(a.\star \leftarrow a'.\star).(c \leftarrow c') = \sum_w (a.w \leftarrow a'.w).(c \leftarrow c')$$

Definition

Let $a = (e_i)_{i=0}^n$ and b be addresses, and let $w = (e_i)_{i=1}^n$. We define the substitution:

$$a\{p_A \mapsto b\} = \begin{cases} b.w & \text{if } e_0 = p_A \\ a & \text{otherwise} \end{cases}$$

Interpretation of proofs I

We represent proofs by triples

$$(\{A_1, \dots, A_n\}, \{(C_1, C_1^\perp), \dots, (C_k, C_k^\perp)\}, P)$$

where:

- ▶ $A_1, \dots, A_n$ are the conclusions of the proof;
- ▶ $\{(C_1, C_1^\perp), \dots, (C_k, C_k^\perp)\}$ are the cut formulas of the proof;
- ▶ P is a sum of flows on extensions of the addresses p_{A_i} ($1 \leq i \leq n$) and p_{C_i} ($1 \leq i \leq k$).

Interpretation of proofs II

Axioms. A proof π consisting in a single axiom rule introducing the atom X and its orthogonal $X^\perp$ is represented by the triple

$$\|\pi\| = (\{X, X^\perp\}, \emptyset, p_X \leftarrow p_{X^\perp} + p_{X^\perp} \leftarrow p_X)$$

Interpretation of proofs III

Tensor. Let π_1 and π_2 be proofs of conclusions $A_1, \dots, A_n, A$ and $B_1, \dots, B_m, B$ respectively. Let π be the proof obtained from π_1 and π_2 by a tensor rule on A and B . Define

$$\|\pi_1\| = (\{A_1, \dots, A_n, A\}, C_1, a_1)$$

$$\|\pi_2\| = (\{B_1, \dots, B_m, B\}, C_2, a_2)$$

and

$$a_\pi = a_{\pi_1} \{p_A \mapsto p_{A \otimes B} \mathbf{l}\} + a_{\pi_2} \{p_B \mapsto p_{A \otimes B} \mathbf{r}\}$$

The interpretation $\|\pi\|$ is then defined as

$$(\{A_1, \dots, A_n, B_1, \dots, B_m, A \otimes B\}, C_1 \cup C_2, a_\pi)$$

Interpretation of proofs IV

Parr. Let π' be a proof of conclusions $A_1, \dots, A_n, A, B$ and π the proof obtained from π' by applying a $\wp$ rule on A and B . We denote by $(\{A_1, \dots, A_n, A, B\}, C, a_{\pi'})$ the interpretation $\|\pi'\|$ of π' . The interpretation of π is then defined as:

$$\|\pi\| = (\{A_1, \dots, A_n, A \wp B\}, C, a_{\pi'}\{p_A \mapsto p_{A \wp B}^l; p_B \mapsto p_{A \wp B}^r\})$$

Interpretation of proofs V

Cut. Let π_1 and π_2 be proofs of conclusions $A_1, \dots, A_n, A$ and $B_1, \dots, B_m, A^\perp$ respectively. Let π be the proof obtained from π_1 and π_2 by a cut rule on A and $A^\perp$. Define:

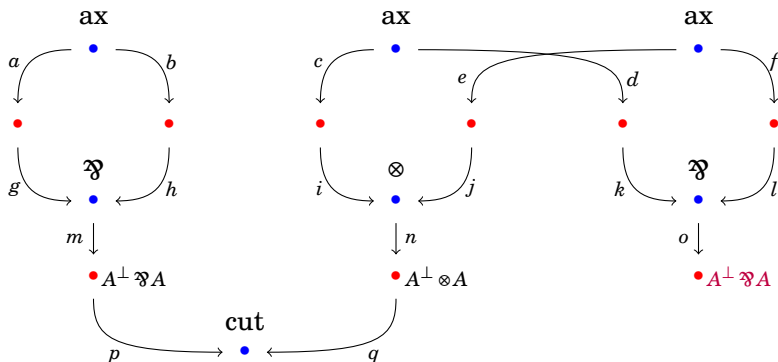
$$\|\pi_1\| = (\{A_1, \dots, A_n, A\}, C_1, P_1)$$

$$\|\pi_2\| = (\{B_1, \dots, B_m, A^\perp\}, C_2, P_2)$$

Then:

$$\|\pi\| = (\{A_1, \dots, A_n, B_1, \dots, B_m\}, C_1 \cup C_2 \cup \{(A, A^\perp)\}, P_1 + P_2)$$

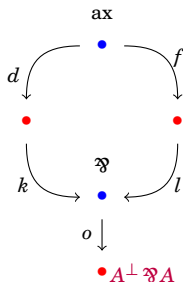
Interpretation of Proofs



$$(\{A^\perp \wp A\}, \{(A^\perp \wp A, A \otimes A^\perp)\}, P)$$

$$P = p_{A^\perp \wp A} \cdot r \Leftarrow p_{A^\perp \wp A} \cdot l + p_{A \otimes A^\perp} \cdot l \Leftarrow p_{A^\perp \wp A} \cdot l + p_{A \otimes A^\perp} \cdot r \Leftarrow p_{A^\perp \wp A} \cdot r$$

Interpretation of Proofs



$$(\{A^\perp \wp A\}, \emptyset, P')$$

$$P' = p_{A^\perp \wp A}.r \Leftrightarrow p_{A^\perp \wp A}.1$$

Execution I

- ▶ Let (A, C, P) be the interpretation of a proof, and $C' \subset C$ a set of cut formulas. Let us denote by R the set of cut formulas in $C - C'$. We define the sum of bundles of flows $1 - C'$ as:

$$1 - C' = \sum_{A \cup R} p_{A \cdot \star} \leftarrow p_{A \cdot \star}$$

This expression plays the role of a partial identity.

- ▶ Moreover, the set $C' \subset C$ defines a sum of bundles of flows we will abusively denote by C' :

$$S_{C'} = \sum_{(A, A^\perp) \in C'} (p_{A \cdot \star} \leftarrow p_{A^\perp \cdot \star} + p_{A^\perp \cdot \star} \leftarrow p_{A \cdot \star})$$

Execution II

- ▶ Let $\|\pi\| = (A, C, P)$ be the interpretation of a proof, and $C' \subset C$ a set of cut formulas. We define the *execution of (A, C, P) along C'* by:

$$\text{Ex}(\|\pi\|, C') = (A, C - C', (1 - C').P(\sum_{i \geq 0} (S_{C'} P)^i).(1 - C'))$$

- ▶ If $C' = C$, we write $\text{Ex}(\|\pi\|)$ the expression $\text{Ex}(\|\pi\|, C)$.

Theorem

Let $\|\pi\| = (A, C, P)$ be the interpretation of a proof, and $C_1, C_2 \subset C$ two disjoint sets of cut formulas. Then:

$$\text{Ex}(\text{Ex}(\|\pi\|, C_1), C_2) = \text{Ex}(\|\pi\|, C_1 \cup C_2) = \text{Ex}(\text{Ex}(\|\pi\|, C_2), C_1)$$

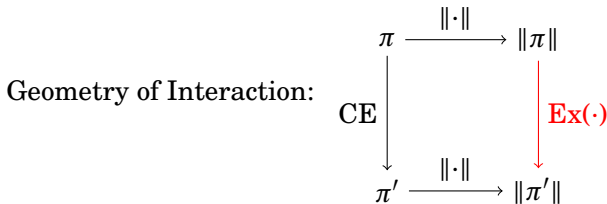
Main Theorems

Theorem (Nilpotency)

Let $\mathcal{R}$ be a proof net, $\|\mathcal{R}\| = (A, C, P)$ its interpretation and $C' \subset C$. Then $S_{C'}P$ is nilpotent, i.e. the expression $\text{Ex}(\|\mathcal{R}\|, C')$ is finite.

Theorem (Execution and Cut-Elimination)

If π is a proof and ρ its normal form, then $\text{Ex}(\|\pi\|) = \|\rho\|$.



GoI: Reconstructing Logic



Jean-Yves Girard,

Multiplicatives

Rend. Semin. Mat. 17, Univ. Politec. Torino, 1987.



Jean-Yves Girard

Geometry of Interaction V: Logic in the Hyperfinite Factor

Theoretical Computer Science 412 (20), 2011.



Thomas Seiller.

Interaction Graphs: Multiplicatives.

Annals of Pure and Applied Logic 163, December 2012.

Arxiv preprint arXiv:1205.6558

LongTrips and Permutations

LongTrips criterion and permutations

We briefly recall the longtrips criterion.

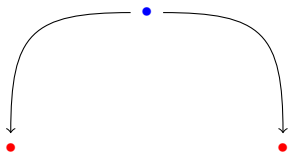
Definition

A switching is a map from the set of $\otimes$ and $\wp$ vertices to the set $\{1, 2\}$.

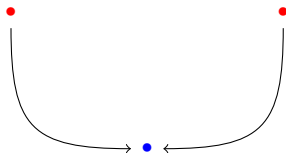
Define $\mathcal{R}_\bullet$ to be the set of couples (v, w) where v is a formula vertex (a red vertex) and w is an element in $\{\uparrow, \downarrow\}$.

Definition

A switching s defines a graph $\mathcal{R}_s$ whose vertices is $\mathcal{R}_\bullet$ (defined in the next slides).

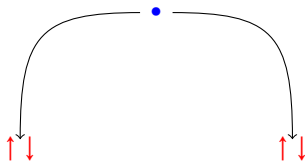


(a) Axiom Brick

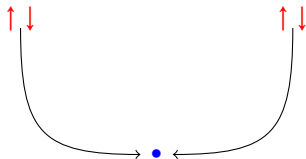


(b) Cut Brick

Figure: Axiom and Cut bricks

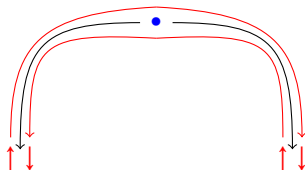


(a) Axiom Brick

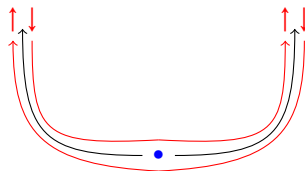


(b) Cut Brick

Figure: Axiom and Cut bricks



(a) Axiom Brick



(b) Cut Brick

Figure: Directions for Axiom and Cut bricks

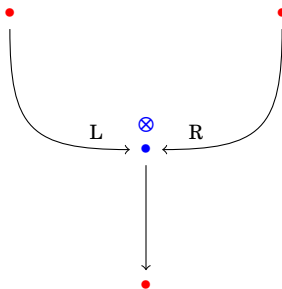


Figure: Tensor bricks

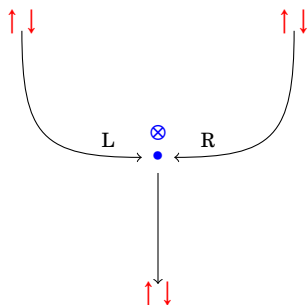
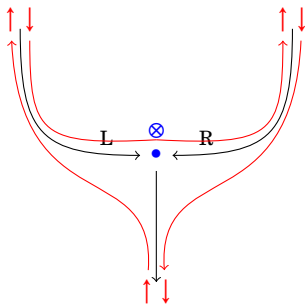
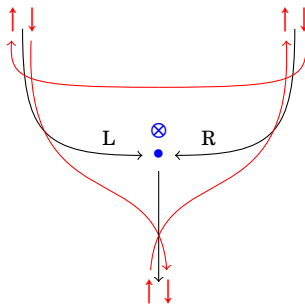


Figure: Tensor bricks

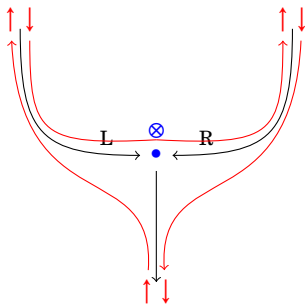


(a) First switching

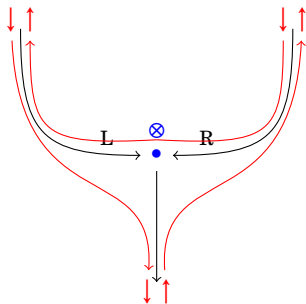


(b) Second switching

Figure: Directions for Tensor bricks



(a) First switching



(b) Second switching

Figure: Directions for Tensor bricks

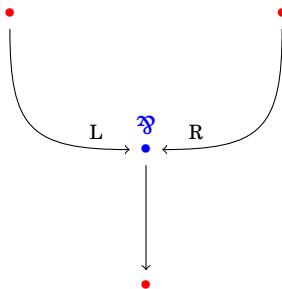


Figure: Parr bricks

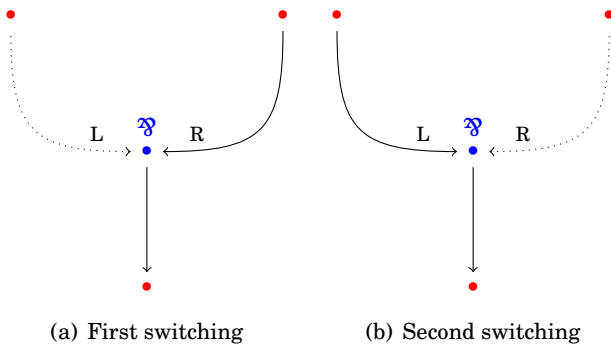


Figure: Parr bricks

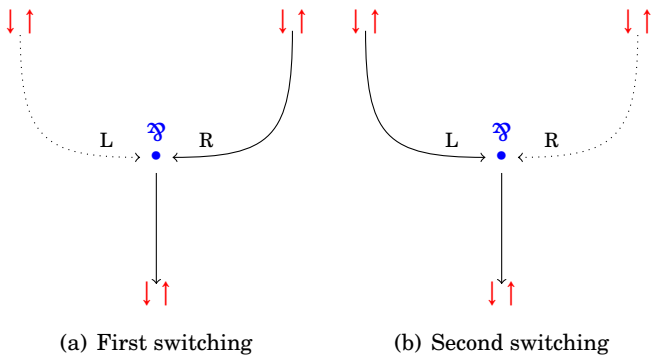


Figure: Directions for Parr bricks

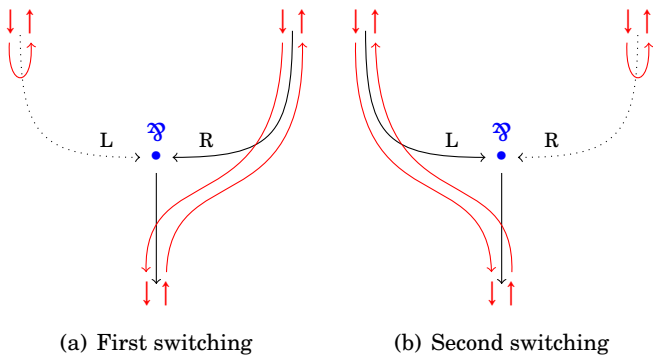


Figure: Directions for Parr bricks

LongTrips criterion

Theorem

A proof structure $\mathcal{R}$ is sequentializable if and only if for all switching s the graph $\mathcal{R}_s$ defines a long trip: if v is a vertex of $\mathcal{R}_s$, the paths from v to v (by following the edges) go through all the edges.

Let $\mathcal{R}$ be a proof structure. A switching s defines a permutation τ_s over the axioms of the atoms in $\mathcal{R}$ as follows:

- ▶ chose an atom a , and consider $a_{\downarrow} = (a, \downarrow)$;
- ▶ follow the edges of $\mathcal{R}_s$ until you arrive at a vertex $(b, \uparrow)$;
- ▶ define $\tau_s(a) = b$.

Similarly, the axiomatic part $\mathcal{R}_{ax}$ of $\mathcal{R}$ defines a permutation σ (which is a disjoint union of transpositions).

Example

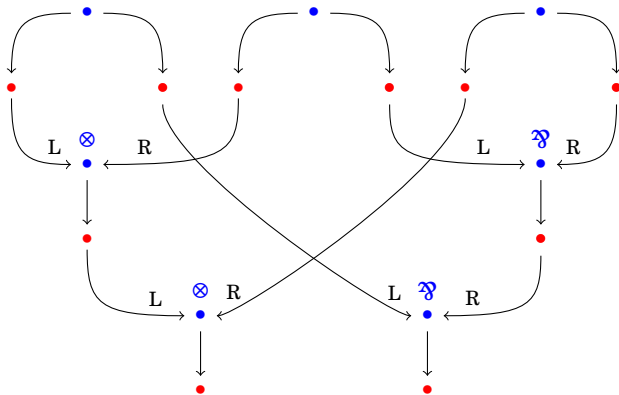


Figure: Associativity: Proof Net

Example

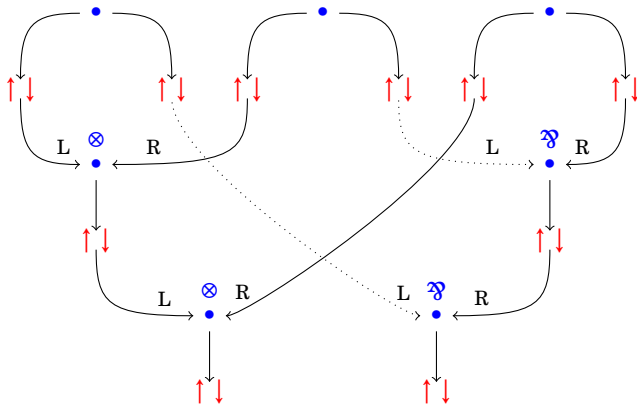


Figure: Associativity: Proof Net

Example

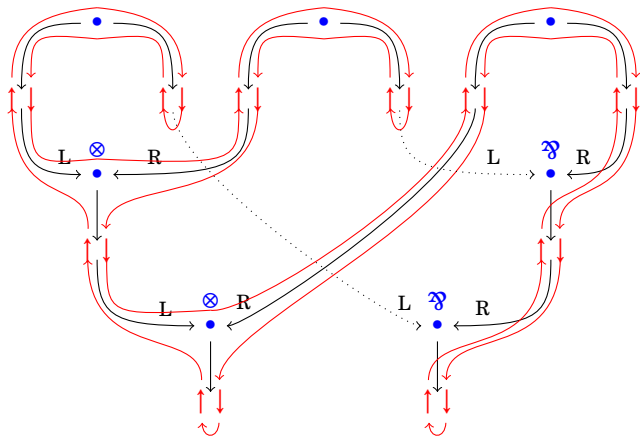


Figure: Associativity: A (Long) Trip

Example

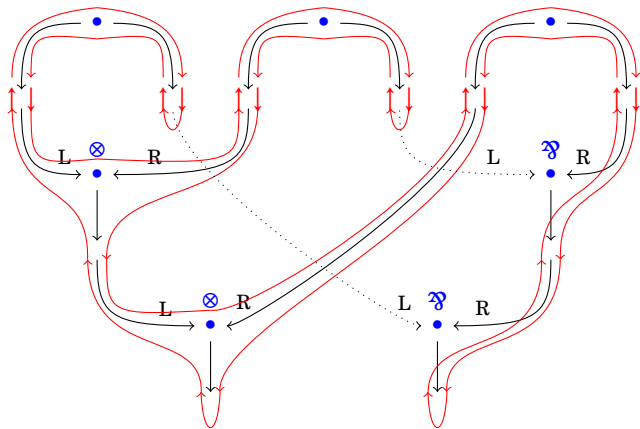


Figure: Associativity: A (Long) Trip

Example

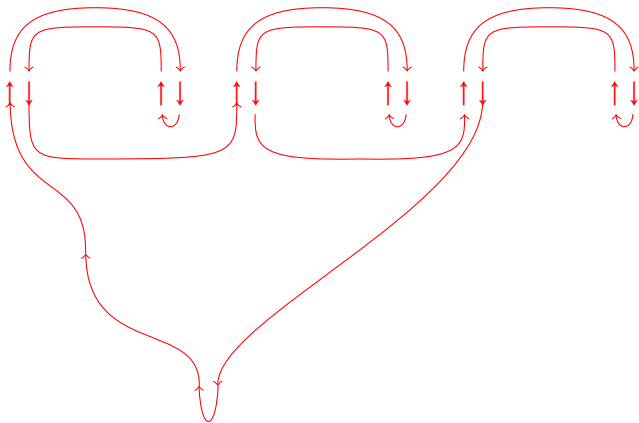


Figure: Associativity: Toward Permutations

Example

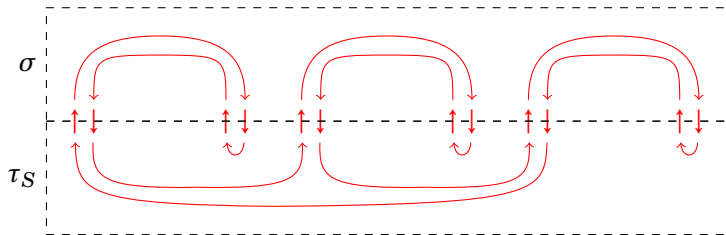


Figure: Associativity: Permutations

Example

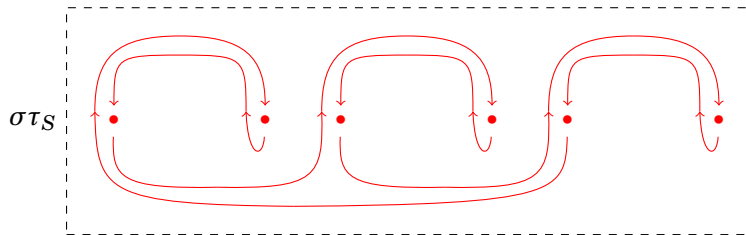


Figure: Associativity: Permutations

LongTrips Criterion: permutations

We get a reformulation of the correctness criterion.

Theorem

A proof structure $\mathcal{R}$ is sequentializable if and only if for all switching s , the permutation $\sigma\tau_s$ is cyclic.

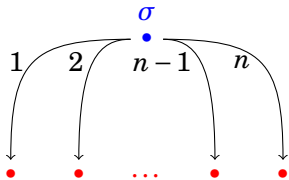
Switchings of $A = \text{Proofs of } A^\perp$

Extending the syntax

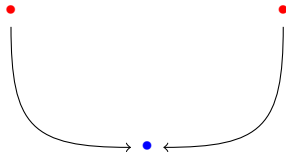
In order to understand better this point of view, we extend the syntax of proof structures. We replace the axiom brick by a *daimon brick*, which allows the introduction of an arbitrary (finite) number of atoms. The daimon brick is given with a cyclic permutation of the atoms it introduces.

Remark

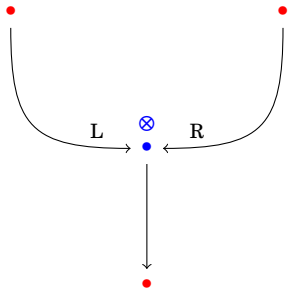
Since permutations can always be written as a product of cycles, the daimon brick allows us to write any permutation. The set of untyped proofs is therefore extended to all permutations (instead of only disjoint unions of transpositions).



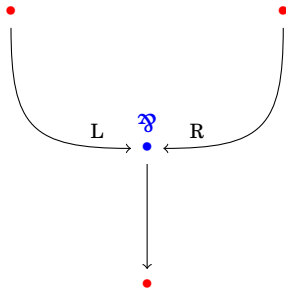
(a) Daimon Brick



(b) Cut Brick



(c) Tensor Brick



(d) Parr Brick

Figure: Basic bricks

Proofs as orthogonals

Given a formula A we define the set $\mathcal{S}(A)$ to be the set of permutations induced by the switchings of A .

Definition

Two permutations σ, τ are said to be *orthogonal* — denoted $\sigma \perp \tau$ — when $\sigma\tau$ is cyclic.

If A is a set of permutations, we write $A^\perp$ the set of permutations that are orthogonal to every element of A .

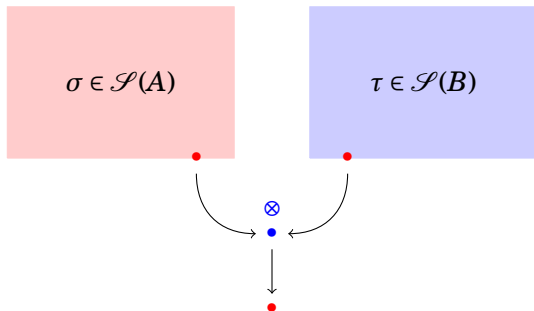
Definition

The set of *paraproofs* of A is the set $\mathcal{P}(A) = (\mathcal{S}(A))^\perp$.

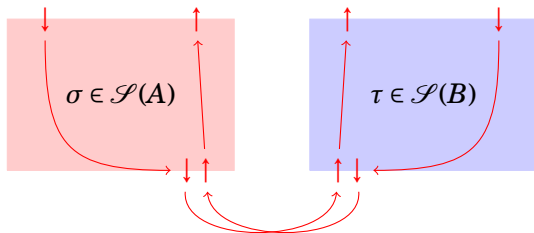
Remark

Both $\mathcal{P}(A)$ and $\mathcal{P}(A^\perp)$ are non-empty.

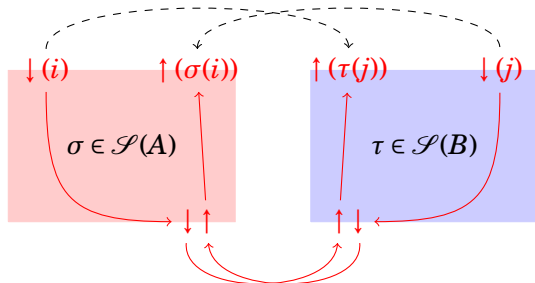
Switchings of $A \otimes B$



Switchings of $A \otimes B$



Switchings of $A \otimes B$



We denote by $\sigma \mathfrak{S}_i^j \tau$ the resulting permutation. It can be defined as $(\sigma(i) \tau(j)) \circ (\sigma \cup \tau)$.

Switchings of $A \otimes B$

Since a switching of $A^\perp \wp B^\perp$ is the union of a switching of $A^\perp$ and a switching of $B^\perp$, we obtain that the permutations $\sigma \wp_i^j \tau$ are orthogonal to all the elements in $\mathcal{S}(A^\perp \wp B^\perp)$.

Proposition

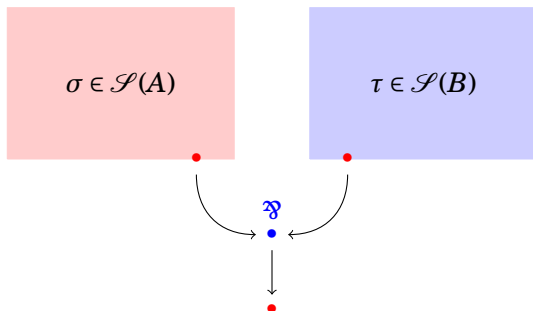
We have the inclusion $\mathcal{S}(A \otimes B) \subset \mathcal{P}(A^\perp \wp B^\perp)$.

Since for all atom a there exists a switching that links a with the conclusion of the proof structure, given a and b two atoms of A and B , there exists an element $\sigma \wp_i^j \tau$ such that $\sigma \wp_i^j \tau(a) = b$. Thus:

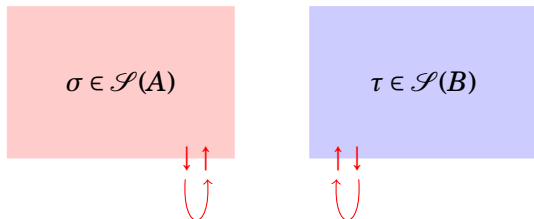
Proposition

$$\mathcal{P}(A \otimes B) = \{\sigma \cup \tau \mid \sigma \in \mathcal{P}(A), \tau \in \mathcal{P}(B)\}$$

Switchings of $A \wp B$



Switchings of $A \wp B$



So switchings of $A \wp B$ are of the form $\sigma \cup \tau$ for $\sigma \in \mathcal{S}(A)$ and $\tau \in \mathcal{S}(B)$. As a consequence:

Proposition

$$\mathcal{S}(A \wp B) \subset \mathcal{P}(A^\perp \otimes B^\perp)$$

Switchings of $A \wp B$

Using the fact that $\mathcal{P}(A^\perp \otimes B^\perp) = \{\sigma \cup \tau \mid \sigma \in \mathcal{P}(A^\perp), \tau \in \mathcal{P}(B^\perp)\}$, we even have:

Proposition

$$\mathcal{S}(A \wp B) = \mathcal{P}(A^\perp \otimes B^\perp)$$

From the preceding results and the definition of $\mathcal{P}$, we obtain:

$$\begin{aligned}\mathcal{P}(A^\perp \wp B^\perp) &= (\mathcal{S}(A^\perp \wp B^\perp))^\perp \\ &= (\mathcal{P}(A^\perp \otimes B^\perp))^\perp \\ &= (\mathcal{S}(A \otimes B))^{\perp\perp}\end{aligned}$$

Multiplicatives

Multiplicatives

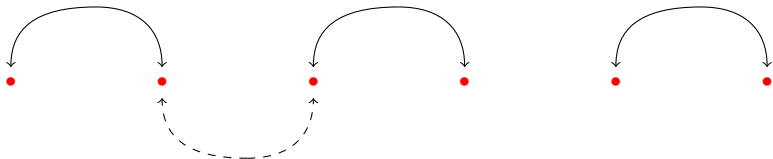
Based on all these remarks, we will now build an interpretation of multiplicative linear logic where proofs are represented by permutations.

The paraproofs (generalized untyped proofs) will be given by permutations of a finite set (addresses).

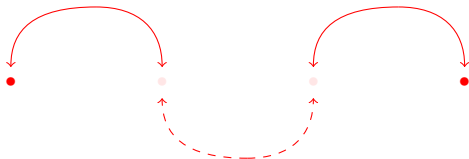
Definition

A *paraproof* is a couple (X, σ) of a finite set X and a permutation $\sigma \in \mathfrak{S}(X)$.

Multiplicatives: execution



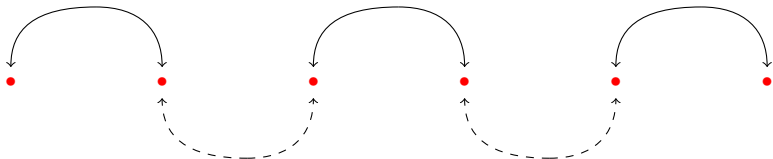
Multiplicatives: execution



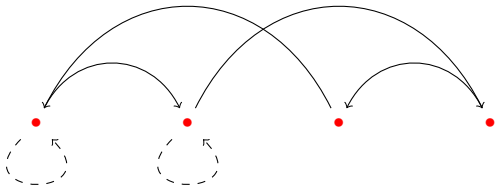
Multiplicatives: execution



Multiplicatives: execution



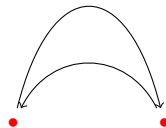
Multiplicatives: execution



Multiplicatives: execution



Multiplicatives: execution



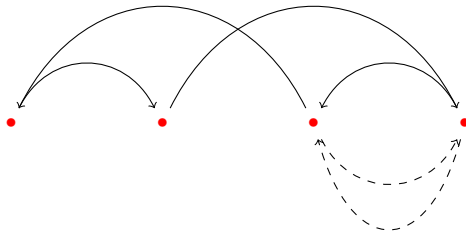
Multiplicatives: execution

Definition

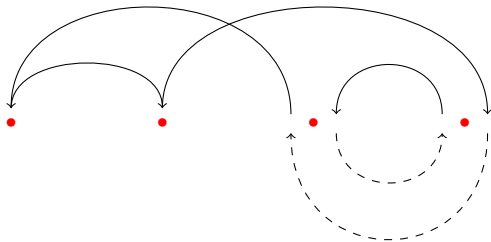
The *execution* of $(X \cup Y, \sigma)$ and (X, τ) is equal to $(Y, \text{Ex}(\sigma, \tau))$, where:

$$\begin{aligned}\text{Ex}(\sigma, \tau) &= \bigcup_{i=0}^{\infty} [\sigma(\tau\sigma)^i]_{|Y} \\ &= \sigma_{|Y} \cup (\sigma\tau\sigma)_{|Y} \cup (\sigma\tau\sigma\tau\sigma)_{|Y} \cup \dots\end{aligned}$$

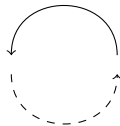
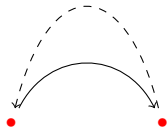
Multiplicatives: execution



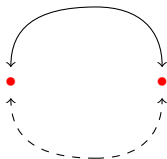
Multiplicatives: execution



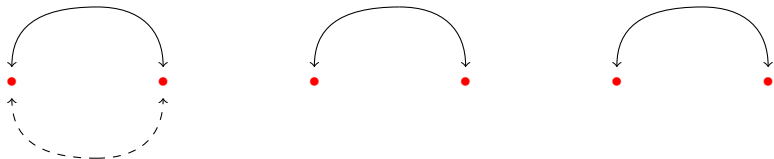
Multiplicatives: execution



Multiplicatives: execution



Multiplicatives: execution



There are two ways of dealing with this:

- ▶ Define execution only when no such *internal cycles* appear: this is what is done in *Multiplicatives*;
- ▶ Define execution in all cases and remember that such cycles appeared: this is what is done in GoI5 (see *Interaction Graphs: Multiplicatives* for an explanation of this);

Multiplicatives: execution

We chose to work in the second case (execution defined in all cases) because:

- ▶ the only thing needed to work in this case is to add a *cycle counter* to paraproofs;
- ▶ the resulting framework is nicer: it is possible to deal with units and the orthogonality can be defined through the execution;

Definition

A *design* is a couple $\mathfrak{a} = (a, p_a)$ where $a \in \mathbf{N}$ and $p_a = (X_a, \sigma_a)$ is a paraproof.

Multiplicatives: execution

Definition

Let (X, σ) and (Y, τ) be paraproofs. We define:

$$\llbracket (X, \sigma), (Y, \tau) \rrbracket_m = \text{Card}\{\text{cycles between } \sigma \text{ and } \tau\}$$

Definition

The execution $\text{Ex}(a, b)$ between two designs is equal to

$$(a + b + \llbracket p_a, p_b \rrbracket_m, \text{Ex}(p_a, p_b))$$

Multiplicatives: orthogonality

Definition

Two designs $\mathfrak{a}$ and $\mathfrak{b}$ are orthogonal — written $\mathfrak{a} \perp \mathfrak{b}$ — when $X_{\mathfrak{a}} = X_{\mathfrak{b}}$ and:

$$\mathfrak{a} + \mathfrak{b} + \llbracket (X_{\mathfrak{a}}, \sigma_{\mathfrak{a}}), (X_{\mathfrak{b}}, \sigma_{\mathfrak{b}}) \rrbracket_m = 1$$

Proposition

Two designs $\mathfrak{a}, \mathfrak{b}$ are orthogonal if and only if $\mathbf{Ex}(\mathfrak{a}, \mathfrak{b}) = (1, (\emptyset, \emptyset))$.

Multiplicatives: Types

If A is a set of designs, we write $A^\perp$ the set:

$$\{b \mid \forall a \in A, a \perp b\}$$

Definition

A *type* $\mathbf{A}$ is a non-empty set of designs which is equal to its bi-orthogonal, i.e. $\mathbf{A} = \mathbf{A}^{\perp\perp}$.

Multiplicatives: connectives

Definition

Let $\mathfrak{a}$ and $\mathfrak{b}$ be designs such that $X_{\mathfrak{a}} \cap X_{\mathfrak{b}} = \emptyset$. We define:

$$\mathfrak{a} \otimes \mathfrak{b} = (\mathfrak{a} + \mathfrak{b}, (X_{\mathfrak{a}} \cup X_{\mathfrak{b}}, \sigma_{\mathfrak{a}} \cup \sigma_{\mathfrak{b}}))$$

Notice that if $\mathbf{A}$ is a type, $\mathfrak{a} \in \mathbf{A}$ and $\mathfrak{b} \in \mathbf{A}$, then $X_{\mathfrak{a}} = X_{\mathfrak{b}}$. We therefore define $X_{\mathbf{A}} = X_{\mathfrak{a}}$ for $\mathfrak{a} \in \mathbf{A}$.

Definition

Let $\mathbf{A}$ and $\mathbf{B}$ be types such that $X_{\mathbf{A}} \cap X_{\mathbf{B}} = \emptyset$. We define:

$$\mathbf{A} \otimes \mathbf{B} = \{\mathfrak{a} \otimes \mathfrak{b} \mid \mathfrak{a} \in \mathbf{A}, \mathfrak{b} \in \mathbf{B}\}^{\sim\sim}$$

Proposition

The connective $\otimes$ is associative, commutative and has a neutral element, namely $\mathbf{1} = \{(0, (\emptyset, \emptyset))\}^{\sim\sim} = \{(0, (\emptyset, \emptyset))\}$.

Multiplicatives: Linear Implication

Proposition

Let $(X_a, \sigma_a), (X_b, \sigma_b), (X_c, \sigma_c)$ be three paraproofs with $X_a \cap X_b = \emptyset$.
Then:

$$\begin{aligned} & \llbracket (X_c, \sigma_c), (X_a \cup X_b, \sigma_a \cup \sigma_b) \rrbracket_m \\ &= \llbracket (X_c, \sigma_c), (X_a, \sigma_a) \rrbracket_m + \llbracket \text{Ex}((X_c, \sigma_c), (X_a, \sigma_a)), (X_b, \sigma_b) \rrbracket_m \end{aligned}$$

Proposition (Adjunction)

Let $\mathfrak{a}, \mathfrak{b}, \mathfrak{c}$ be designs with $X_a \cap X_b = \emptyset$. Then:

$$\llbracket \mathfrak{c}, \mathfrak{a} \otimes \mathfrak{b} \rrbracket_m = \llbracket \text{Ex}(\mathfrak{c}, \mathfrak{a}), \mathfrak{b} \rrbracket_m$$

Proposition

Let $\mathbf{A}, \mathbf{B}$ be types with $X_{\mathbf{A}} \cap X_{\mathbf{B}} = \emptyset$. Then:

$$(\mathbf{A} \otimes \mathbf{B}^\perp)^\perp = \{f \mid \forall \mathfrak{a} \in \mathbf{A}, \text{Ex}(f, \mathfrak{a}) \in \mathbf{B}\} = \mathbf{A} \multimap \mathbf{B}$$

Multiplicatives: truth

Definition

A design α is *successful* when $\alpha = 0$ and $(X_\alpha, \sigma_\alpha)$ is a disjoint union of transpositions. A type $\mathbf{A}$ is *true* when it contains a successful design.

Proposition

The types $\mathbf{A}, \mathbf{A}^\perp$ cannot be simultaneously true.

Proposition

If $\mathbf{A} \multimap \mathbf{B}$ and $\mathbf{B} \multimap \mathbf{C}$ are true, then $\mathbf{A} \multimap \mathbf{C}$ is true.

Multiplicatives: denotational semantics

In this setting, it is possible to interpret the proofs and formulas of Multiplicative Linear Logic with units (MLL) and this interpretation satisfies the following properties:

- ▶ A proof π of $\vdash A$ is interpreted as a winning design $\pi^\bullet$ in the interpretation $A^\bullet$ of A ;
- ▶ The interpretation is invariant under cut-elimination;

One can also define a category Graph_{MLL} whose objects are types over finite subsets of $\mathbf{N}$ and morphisms from $\mathbf{A}$ to $\mathbf{B}$ are elements of $\phi_0(\mathbf{A}) \multimap \phi_1(\mathbf{B})$. The resulting category, together with the bifunctor induced by $\otimes$ is a symmetric monoidal category with a dualizing object $\perp = \{(0, \emptyset)\}^\perp$, i.e. a $*$ -autonomous category.

Recent Work

Interaction Graphs



Thomas Seiller.

Interaction Graphs: Multiplicatives.

Annals of Pure and Applied Logic 163, December 2012.

Arxiv preprint arXiv:1205.6558



Thomas Seiller.

Interaction Graphs: Additives.

Submitted to *Annals of Pure and Applied Logic*

Arxiv preprint arXiv:1205.6557, 2012.



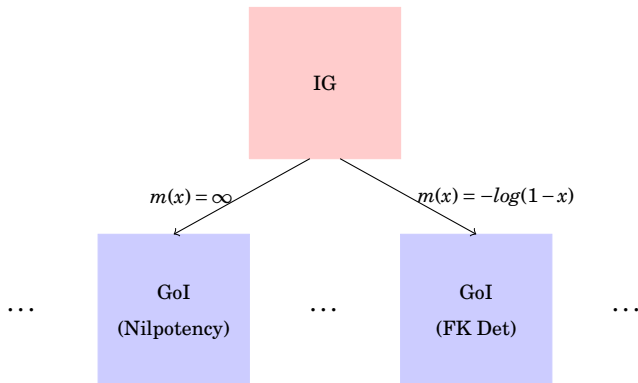
Thomas Seiller.

Logique dans le Facteur Hyperfini: Géométrie de
l'Interaction et Complexité.

PhD Thesis, Aix-Marseille University, November 2012.

Interaction Graphs

Instead of counting cycles, we measure them (edges are weighted). We obtain a family of constructions parametrized by a "measure" m .



- ▶ In a generalization of this setting one can define several exponential connectives !.
- ▶ In particular, we can define a ! connective yielding a restricted system: Elementary Linear Logic (ELL).

Elementary Linear Logic is a version of Linear Logic with a restriction on the rules governing the exponential connectives so that the proofs of the type $!nat \multimap nat$ are exactly the functions from $\mathbf{N}$ to $\mathbf{N}$ which are computable in elementary time.

Computational Complexity



Clément Aubert and Thomas Seiller.

Characterizing co-NL by a Group Action.

Submitted to Mathematical Structures in Computer Science

Arxiv preprint arXiv:1209.3422, 2012.



Clément Aubert and Thomas Seiller.

Logarithmic Space and Permutations.

Submitted to Information and Computation

Arxiv preprint arXiv:1301.3189, 2013.



Thomas Seiller.

Logique dans le Facteur Hyperfini: Géométrie de
l'Interaction et Complexité.

PhD Thesis, Aix-Marseille University, November 2012.

Representation of Integers

- ▶ Principle: an integer n is represented as a binary list, i.e. as a proof of

$$\underbrace{!(X \multimap X) \multimap}_{0} \underbrace{!(X \multimap X) \multimap}_{1} \underbrace{!(X \multimap X)}_{\star}$$

Notice that this is not the usual type of binary lists: this is the binary lists type in Elementary Linear Logic.

- ▶ The list can be read from the contraction rules.
- ▶ The GoI interpretation of these proofs are matrices (M_n) representing the sets of axiom links: we obtain a 6×6 matrix whose coefficients are $k \times k$ matrices ($k = \log_2(n)$ is the length of the list).

Results

- In the formal setting:

$$\underbrace{\mathfrak{M}_6(\mathbf{C}) \otimes \mathfrak{R} \otimes \mathfrak{M}_n(\mathbf{C})}_{\text{integers}}$$

we define "machines" as a set R of operators in $\mathfrak{M}_6 \otimes \mathfrak{M}_n$ and we can show that $[R] = \{[r], r \in R\}$ is the set of regular languages.

- In the more complex setting:

$$\mathfrak{M}_6(\mathbf{C}) \otimes \left(\left(\bigotimes_{n \in \mathbf{N}} \mathfrak{R} \right) \rtimes \mathfrak{G} \right) \otimes \mathfrak{M}_n(\mathbf{C})$$

we define two sets P_+ and $P_{+,1}$ of operators in $\mathfrak{M}_6 \otimes \text{vN}(\mathfrak{G}) \otimes \mathfrak{M}_n$ and show:

$$[P_+] = \mathbf{co-NL} \quad [P_{+,1}] = \mathbf{L}$$

Untyped Proof Theory: Logical Constants



Alberto Naibo, Mattia Petrolo and Thomas Seiller.

On the Dynamical Meaning of Axioms.

In preparation.



Alberto Naibo, Mattia Petrolo and Thomas Seiller.

Logical Constants from a Computational Point of View.

In preparation.

Logical Constants in GoI

- ▶ Every operation on untyped proofs allows one to define an operation on types. For instance, if $g(x,y)$ defines an untyped proof from two untyped proofs x and y , we define the operation on types

$$g(A,B) = \{g(x,y) : x \in A, y \in B\}^{\perp\perp}$$

An operation on types will be a logical constant when both it and its dual have a computational meaning, i.e. when they are defined as a "natural" construction on untyped proofs.

- ▶ For instance, in the permutations framework a natural construction on untyped proofs can be defined as a construction that preserves inclusions and/or correctness.

A non-logical operation

- ▶ Let's take an untyped proof $a = \langle X, \sigma \rangle$ and define the operation of *square* exponentiation: $a^2 = \langle X, \sigma^2 \rangle$
- ▶ Given a type $\mathbf{A}$, the square operation over untyped proofs induces the new type: $\Box \mathbf{A} = \{a^2 \mid a \in \mathbf{A}\}^{\perp\perp}$

We now take a look at some examples:

- ▶ The sets $\mathbf{F}_2 = \{\langle\{1, 2\}, id\rangle\}$ and $\mathbf{F}_3 = \{\langle\{1, 2, 3\}, id\rangle\}$ are types.
- ▶ We have $\mathbf{C}_3 = \{\langle\{1, 2, 3\}, (1, 2, 3)\rangle, \langle\{1, 2, 3\}, (1, 3, 2)\rangle\} = \mathbf{F}_3^{\perp}$ and $\mathbf{C}_2 = \{\langle\{1, 2\}, (1, 2)\rangle\} = \mathbf{F}_2^{\perp}$
- ▶ These equalities are satisfied: $\Box \mathbf{F}_2 = \mathbf{F}_2$, $\Box \mathbf{C}_2 = \mathbf{F}_2$, $\Box \mathbf{F}_3 = \mathbf{F}_3$, $\Box \mathbf{C}_3 = \mathbf{C}_3$, hence $(\Box \mathbf{C}_2)^{\perp} = \mathbf{C}_2$ and $(\Box \mathbf{C}_3)^{\perp} = \mathbf{F}_3$

A non-logical operation

- ▶ There is not a general and univocal operation over permutations that permits to define $(\Box \mathbf{A})^\perp$, for *every* type $\mathbf{A}$.
- ▶ This operation should be an operation f on permutations, such that if $\mathfrak{b} = \langle X, \tau \rangle \in \mathbf{A}^\perp$, then $\langle X, f(\tau) \rangle \in (\Box \mathbf{A})^\perp$.
- ▶ This means that f should respect the following condition: for every $\mathfrak{a} = \langle X, \sigma \rangle \in \mathbf{A}$ and $\mathfrak{b} = \langle X, \tau \rangle \in \mathbf{A}^\perp$, if $\tau\sigma$ is cyclic then $f(\tau)\sigma^2$ is cyclic.
- ▶ It is not possible to define a unique f for every type $\mathbf{A}$.

A logical operation

- ▶ Define $\mathcal{T}(\mathbf{A}) = \{\langle X, \tau^{-1} \rangle \mid \langle X, \tau \rangle \in \mathbf{A}\}^{\perp\perp}$.
- ▶ Then $(\mathcal{T}(\mathbf{A}))^{\perp} = \mathcal{T}(\mathbf{A}^{\perp})$.
- ▶ This construction on types satisfies a lot of good properties:
 - ▶ It preserves correctness: the inverse of a disjoint union of transpositions is a disjoint union of transpositions;
 - ▶ It is “natural”: if $\sigma \subset \tau$, then $\sigma^{-1} \subset \tau^{-1}$;
 - ▶ It is “internally complete”: $\mathcal{T}(\mathbf{A}) = \{\langle X, \tau^{-1} \rangle \mid \langle X, \tau \rangle \in \mathbf{A}\}$.

Conclusions

- ▶ The framework is rich enough: each operation on untyped proofs defines an operation on types and there already exists many properties of untyped proofs and types that could be used to distinguish between logical and non logical operations on types, for instance:
 - ▶ the preservation of correctness;
 - ▶ the "naturality", i.e. the preservation of inclusion in case of permutations;
 - ▶ internal completeness (i.e. the closure by bi-orthogonality is not necessary);
 - ▶ ...
- ▶ The notion of execution needs not be adapted when a new construction is introduced, as opposed to what happens when one works with lambda-calculus.