Proof diagrams: another parallel syntax for proof theory

Matteo Acclavio

Laboratoire de Mathématiques Nicolas Oresme

22/06/2017 Bologna

Example (Pitagora's Theorem)

In a right triangle, the square of the hypotenuse (the side opposite the right angle) is equal to the sum of the squares of the other two sides.

$$c^2 = a^2 + b^2$$

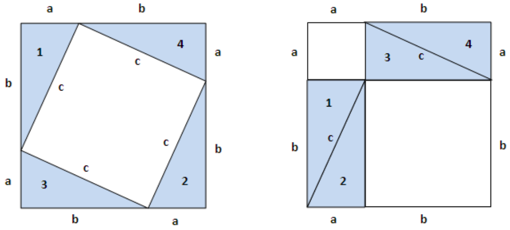
Proof:

Example (Pitagora's Theorem)

In a right triangle, the square of the hypotenuse (the side opposite the right angle) is equal to the sum of the squares of the other two sides.

$$c^2 = a^2 + b^2$$

Proof:

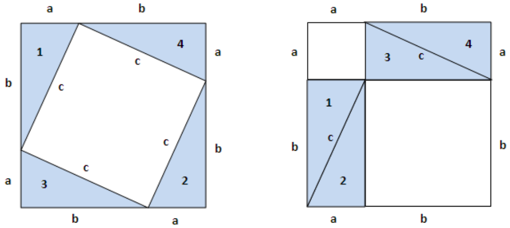


Example (Pitagora's Theorem)

In a right triangle, the square of the hypotenuse (the side opposite the right angle) is equal to the sum of the squares of the other two sides.

$$c^2 = a^2 + b^2$$

Proof:



$$(a+b)^2 = a^2 + b^2 + 4(\frac{1}{2}ab)$$

Example (Pitagora's Theorem)

In a right triangle, the square of the hypotenuse (the side opposite the right angle) is equal to the sum of the squares of the other two sides.

$$c^2 = a^2 + b^2$$

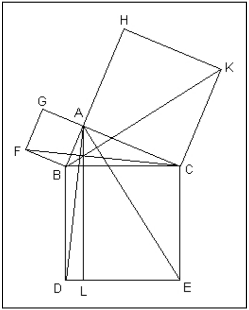
Proof [Euclide]:

- $A\hat{C}E = A\hat{C}E + B\hat{C}E = A\hat{C}B + C\hat{A}K = B\hat{C}K$;
- $AC = CK$ and $CB = CE$;

$\Rightarrow$ $\overset{\triangle}{BCK} = \overset{\triangle}{ACE}$;

- $A(\overset{\triangle}{BCK}) = \frac{1}{2}A(\overset{\square}{ACKH})$;
- $A(\overset{\triangle}{ACE}) = \frac{1}{2}A(\overset{\square}{CELM})$;

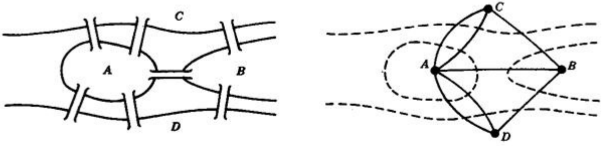
$\Rightarrow$ $A(\overset{\square}{ACKH}) = A(\overset{\square}{CELM})$



$\square$

Euler and the seven bridges of Königsberg (1736)

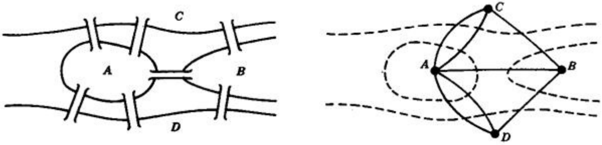
The foundation of graph theory.



(a) Königsberg in 1736 (b) Euler's graphical representation

Euler and the seven bridges of Königsberg (1736)

The foundation of graph theory.

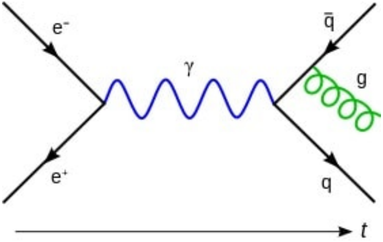


(a) Königsberg in 1736 (b) Euler's graphical representation

⚠ . . . but graphs are 3-dimensiontal objects.

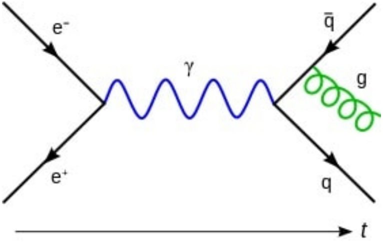
Feynman diagrams (1948)

Representation of interaction between subatomic particles.



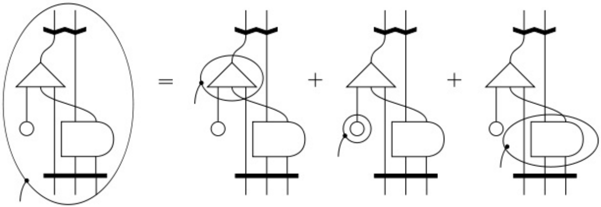
Feynman diagrams (1948)

Representation of interaction between subatomic particles.



Intertwining operators between positive-energy represetations of the Poincaré group
(morphisms of the symmetric monoidal category of positive-energy represetations of the Poincaré group).

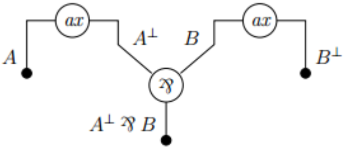
Penrose diagrams: spin networks (1971)



Interwiners operators between irreducible representations of a compact Lie group.

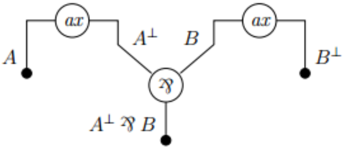
. . . back to logic

Girard's proof nets (1987)

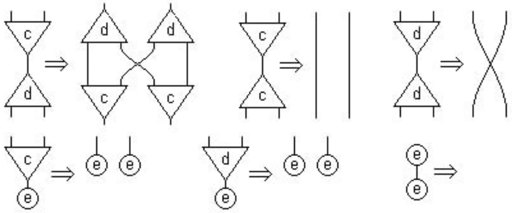


...back to logic

Girard's proof nets (1987)



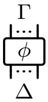
Lafont's interaction nets (1990)



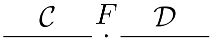
Joyal-Street's string diagrams

André Joyal and Ross Street, *The geometry of tensor calculus*, 1991.
Albert Burroni, *Higher dimensional word problem with application to equational logic*, 1993.

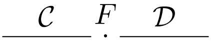
In general, a string diagram $\phi : \Gamma \Rightarrow \Delta$ is a morphism with inputs $\Gamma = \Gamma_1, \ldots, \Gamma_n$ and outputs $\Delta = \Delta_1, \ldots, \Delta_k$ and it is represented as follows:



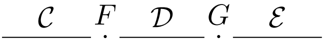
String diagrams (or simply *diagrams*) are a way of notating natural transformations and functors. The idea behind is to represent a functor $\mathcal{C} \xrightarrow{F} \mathcal{D}$ as



String diagrams (or simply *diagrams*) are a way of notating natural transformations and functors. The idea behind is to represent a functor $\mathcal{C} \xrightarrow{F} \mathcal{D}$ as

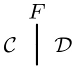


and a sequential composition $\mathcal{C} \xrightarrow{F} \mathcal{D} \xrightarrow{G} \mathcal{E}$ as

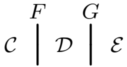


.

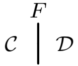
Expanding these representations into 2-dimensional ones, the notation for the previous functor and sequential composition are the following:



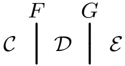
and



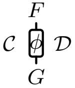
Expanding these representations into 2-dimensional ones, the notation for the previous functor and sequential composition are the following:



and



A natural transformation ϕ between two functon $F, G : \mathcal{C} \to \mathcal{D}$ will be represented as follow

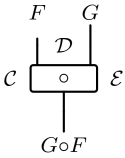


.

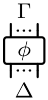
For example, the natural transformation of functor composition

$$\circ : \mathcal{C} \xrightarrow{F} \mathcal{D} \xrightarrow{G} \mathcal{E} \Rightarrow \mathcal{C} \xrightarrow{G\circ F} \mathcal{E}$$

is represented as follows:



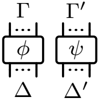
Back to our definition:
String diagrams (with no colors on backgrounds) are a 2-dimensional syntax for morphisms in monoidal categories
(here the comma denotes the product and $\square$ the neutral object).



Diagrams may be composed in two different ways:

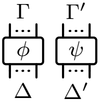
Diagrams may be composed in two different ways:

- *parallel* composition:

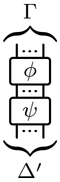


Diagrams may be composed in two different ways:

- *parallel* composition:



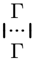
- (partial) *sequential* composition which corresponds to usual composition of maps:



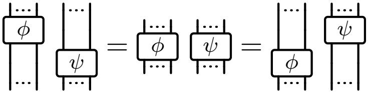
These compositions are associative with unit(s) respectivelly *empty diagram* $\mathbf{id}_0 : \square \Rightarrow \square$ and $\mathbf{id}_\Gamma : \Gamma \Rightarrow \Gamma$ for each $\Gamma \in \Sigma^*$. The *identity diagrams* id_Γ are pictured as follows:

$$\begin{array}{c}\Gamma\\ |\cdots|\\ \Gamma\end{array}$$

These compositions are associative with unit(s) respectivelly *empty diagram* $\mathbf{id}_0 : \square \Rightarrow \square$ and $\mathbf{id}_\Gamma : \Gamma \Rightarrow \Gamma$ for each $\Gamma \in \Sigma^*$. The *identity diagrams* id_Γ are pictured as follows:



Our two compositions satisfy the *interchange rule*:



Linear logic sintaxes for proofs

Actually, in linear logic we have two syntaxes to represent proofs:

- Sequent calculus formalism was introduced by Genzen in 1933 for classical logic: a proof is represented by means of a sequence (tree) of *inference rules* over sequents.
- Proof nets was introduced by Girard in 1987 for MLL sequent calculus: a proof is represented by means of a graph with vertexes *connectives* and edges *formulas*.
  The extension of this formalism to other fragments of LL requires additional syntactical expedients as *jumps* and *boxes*.

In this talk we will focus on *multiplicative linear logic with units*.

Linear logic sintaxes for proofs: sequent calculus

Structural Rules	Identity or Axiom $$\frac{}{\vdash A, A^\perp}\,Ax$$	Cut $$\frac{\vdash \Sigma, A \qquad \vdash \Gamma, A^\perp}{\vdash \Sigma, \Gamma}\,Cut$$
Multiplicative Rules	Par $$\frac{\vdash \Sigma, A, B}{\vdash \Sigma, A \parr B}\,\parr$$	Tensor $$\frac{\vdash \Sigma, A \qquad \vdash B, \Gamma}{\vdash \Sigma, (A \otimes B), \Gamma}\,\otimes$$
Constants	Bottom $$\frac{\vdash \Sigma}{\vdash \Sigma, \bot}\,\bot$$	1 $$\frac{}{\vdash 1}\,1$$

We also consider the usually omitted exchange rule:

$$\frac{\vdash A_1, \ldots, A_k}{\vdash A_{\sigma(1)}, \ldots, A_{\sigma(k)}}\,\sigma \in S_k$$

Theorem (Cut-elimination)

If $\vdash \Gamma$ is derivable in MLL_u, then it is derivable without Cut inference rule.

Proof.

The proof of theorem follows the termination of the following *cut-elimination procedure*. □

Definition (Cut-elimination procedure)

The cut-elimination procedure is the relation $\to_{Cut}$ generated by the following (oriented) relations called *cut-elimination steps*:

$$\dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, A\end{array} \qquad \dfrac{}{\vdash A^\perp, A}\,\mathsf{Ax}}{\vdash \Gamma, A}\,Cut \quad \to_{Cut} \quad \begin{array}{c}\vdots\\ \vdash \Gamma, A\end{array} \qquad\qquad \dfrac{\dfrac{}{\vdash A, A^\perp}\,\mathsf{Ax} \qquad \begin{array}{c}\vdots\\ \vdash \Gamma, A\end{array}}{\vdash \Gamma, A}\,Cut \quad \to_{Cut} \quad \begin{array}{c}\vdots\\ \vdash \Gamma, A\end{array}$$

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, A\end{array} \qquad \begin{array}{c}\vdots\\ \vdash B, \Delta\end{array}}{\vdash \Gamma, \Delta, A \otimes B}\,\otimes \qquad \dfrac{\begin{array}{c}\vdots\\ \vdash B^\perp, A^\perp, \Sigma\end{array}}{\vdash B^\perp \parr A^\perp, \Sigma}\,\parr}{\vdash \Gamma, \Delta, \Sigma}\,\mathsf{Cut} \quad \to_{Cut} \quad \dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, A\end{array} \qquad \dfrac{\begin{array}{c}\vdots\\ \vdash B, \Delta\end{array} \qquad \begin{array}{c}\vdots\\ \vdash B^\perp, A^\perp, \Sigma\end{array}}{\vdash \Delta, A^\perp, \Sigma}\,Cut}{\vdash \Gamma, \Delta, \Sigma}\,Cut$$

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash B^\perp, A^\perp, \Sigma\end{array}}{\vdash B^\perp \parr A^\perp, \Sigma}\,\parr \qquad \dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, A\end{array} \qquad \begin{array}{c}\vdots\\ \vdash B, \Delta\end{array}}{\vdash \Gamma, \Delta, A \otimes B}\,\otimes}{\vdash \Gamma, \Delta, \Sigma}\,\mathsf{Cut} \quad \to_{Cut} \quad \dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash B^\perp, A^\perp, \Sigma\end{array} \qquad \begin{array}{c}\vdots\\ \vdash B, \Delta\end{array}}{\vdash \Delta, A^\perp, \Sigma}\,Cut \qquad \begin{array}{c}\vdots\\ \vdash \Gamma, A\end{array}}{\vdash \Gamma, \Delta, \Sigma}\,Cut$$

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma\end{array}}{\vdash \Gamma, \bot}\,\bot \qquad \dfrac{}{\vdash 1}\,1}{\vdash \Gamma}\,Cut \quad \to_{Cut} \quad \begin{array}{c}\vdots\\ \vdash \Gamma\end{array} \qquad\qquad \dfrac{\dfrac{}{\vdash 1}\,1 \qquad \dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma\end{array}}{\vdash \Gamma, \bot}\,\bot}{\vdash \Gamma}\,Cut \quad \to_{Cut} \quad \begin{array}{c}\vdots\\ \vdash \Gamma\end{array}$$

Some occurrences of Cut rule can not be directly removed by the cut-elimination procedure.

Definition (Commutative cut)

An occurrence of a Cut rule is a *commutative cut* if one of its active formula is not the principal.

In order to remove such occurrences, there are two options:

- Define some additional transformations over derivation;
- Define an equivalence over derivations.

Definition

We define the *standard equivalence over MLL_u derivations* (denoted by $\sim$) as the equivalence derivations generated by the following equivalences for all $A, B, C, D \in \mathfrak{F}_{M\ell\ell_u}$, $\Gamma, \Delta, \Sigma \in \mathfrak{F}^*_{M\ell\ell_u}$:

$$\dfrac{\dfrac{\vdots}{\vdash \Gamma, \Delta, \Sigma}}{\dfrac{\vdash \odot_1(\Gamma), \Delta, \Sigma}{\vdash \odot_1(\Gamma), \odot_2(\Delta), \Sigma}\,\odot_2}\,\odot_1 \quad \sim \quad \dfrac{\dfrac{\vdots}{\vdash \Gamma, \Delta, \Sigma}}{\dfrac{\vdash \Gamma, \odot_2(\Delta), \Sigma}{\vdash \odot_1(\Gamma), \odot_2(\Delta), \Sigma}\,\odot_1}\,\odot_2$$

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash \Delta, A} \qquad \dfrac{\vdots}{\vdash B, \Gamma, \Sigma}}{\vdash \Delta, \odot_1(A, B), \Gamma, \Sigma}\,\odot_1}{\vdash \Delta, \odot_1(A, B), \odot_2(\Gamma), \Sigma}\,\odot_2 \quad \sim \quad \dfrac{\dfrac{\vdots}{\vdash \Delta, A} \qquad \dfrac{\dfrac{\vdots}{\vdash B, \Gamma, \Sigma}}{\vdash \Delta, B, \odot_2(\Gamma), \Sigma}\,\odot_2}{\vdash \Delta, \odot_1(A, B), \odot_2(\Gamma), \Sigma}\,\odot_1$$

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash \Gamma, A, \Sigma} \qquad \dfrac{\vdots}{\vdash B, \Delta}}{\vdash \Gamma, \odot_1(A, B), \Delta, \Sigma}\,\odot_1}{\vdash \odot_2(\Gamma), \odot_1(A, B), \Delta, \Sigma}\,\odot_2 \quad \sim \quad \dfrac{\dfrac{\dfrac{\vdots}{\vdash \Gamma, A, \Sigma}}{\vdash \odot_2(\Gamma), A, \Sigma}\,\odot_2 \qquad \dfrac{\vdots}{\vdash B, \Delta}}{\vdash \odot_2(\Gamma), \odot_1(A, B), \Delta, \Sigma}\,\odot_1$$

$$\dfrac{\dfrac{\begin{matrix}\vdots\\ \vdash \Gamma, A\end{matrix} \qquad \begin{matrix}\vdots\\ \vdash \Sigma, B, C\end{matrix}}{\vdash \Gamma, \Sigma, \odot_1(A,B), C}\odot_1 \qquad \begin{matrix}\vdots\\ \vdash \Delta, D\end{matrix}}{\vdash \Gamma, \Sigma, \Delta, \odot_1(A,B), (C \odot_2 D)}\odot_2 \quad \sim \quad \dfrac{\begin{matrix}\vdots\\ \vdash \Gamma, A\end{matrix} \qquad \dfrac{\begin{matrix}\vdots\\ \vdash \Sigma, B, C\end{matrix} \qquad \begin{matrix}\vdots\\ \vdash \Delta, D\end{matrix}}{\vdash \Sigma, \Delta, B, \odot_2(C,D)}\odot_2}{\vdash \Gamma, \Sigma, \Delta, \odot_1(A,B), \odot_2(C,D)}\odot_1$$

$$\dfrac{\dfrac{\begin{matrix}\vdots\\ \vdash \Gamma, A, C\end{matrix} \qquad \begin{matrix}\vdots\\ \vdash \Sigma, B\end{matrix}}{\vdash \Gamma, \Sigma, \odot_1(A,B), C}\odot_1 \qquad \begin{matrix}\vdots\\ \vdash \Delta, D\end{matrix}}{\vdash \Gamma, \Sigma, \Delta, \odot_1(A,B), (C \odot_2 D)}\odot_2 \quad \sim \quad \dfrac{\dfrac{\begin{matrix}\vdots\\ \vdash \Gamma, A, C\end{matrix} \qquad \begin{matrix}\vdots\\ \vdash \Delta, D\end{matrix}}{\vdash \Gamma, \Delta, A, \odot_2(C,D)}\odot_2 \qquad \begin{matrix}\vdots\\ \vdash \Sigma, B\end{matrix}}{\vdash \Gamma, \Sigma, \Delta, \odot_1(A,B), (C \odot_2 D)}\odot_1$$

$$\dfrac{\begin{matrix}\vdots\\ \vdash \Sigma, C\end{matrix} \qquad \dfrac{\begin{matrix}\vdots\\ \vdash \Gamma, A\end{matrix} \qquad \begin{matrix}\vdots\\ \vdash \Delta, D, B\end{matrix}}{\vdash \Gamma, \Delta, D, \odot_1(A,B)}\odot_1}{\vdash \Gamma, \Sigma, \Delta, \odot_1(A,B), \odot_2(C,D)}\odot_2 \quad \sim \quad \dfrac{\begin{matrix}\vdots\\ \vdash \Gamma, A\end{matrix} \qquad \dfrac{\begin{matrix}\vdots\\ \vdash \Sigma, C\end{matrix} \qquad \begin{matrix}\vdots\\ \vdash \Delta, D, B\end{matrix}}{\vdash \Sigma, \Delta, B, \odot_2(C,D)}\odot_2}{\vdash \Gamma, \Sigma, \Delta, \odot_1(A,B), \odot_2(C,D)}\odot_1$$

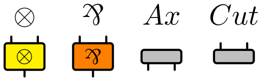
Linear logic sintaxes for proofs: proof nets

Definition (Interaction net)

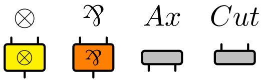
An *interaction net* is given by:

- a finite set of free ports X;
- a finite set of cells C;
- a label $l(c)$ for each $c \in C$ (which defines the number of its active and non-active ports);
- a finite set of wires W;
- a set $\partial(w)$ of 0 or 2 ports for each $w \in W$.

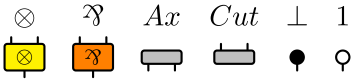
Set of cells types for MLL proof structures:



Set of cells types for MLL proof structures:

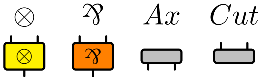


Set of cells types for MLL_u proof structures:

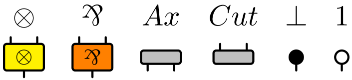


Plus an extra edge from each $\perp$-cell to a cell Ax or 1.

Set of cells types for MLL proof structures:

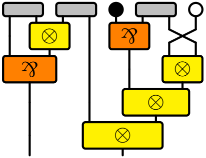


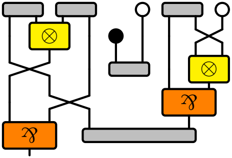
Set of cells types for MLL_u proof structures:



Plus an extra edge from each $\perp$-cell to a cell Ax or 1.

For example:





Remark

Proof nets represent equivalence classes of proofs.

Remark

Proof nets represent equivalence classes of proofs.

In MLL two derivations D and D' are equivalent if and only if their associate proof nets are the same (graph isomorphic).

In MLL_u two derivations D and D' are equivalent if and only if their associate proof nets are equivalent modulo jump re-assignations (graph isomorphism $+$ graph rewriting).

Definition (Proof net)

A proof net is a sequentializable proof structure, i.e. a proof structure which represents a derivation.

In order to recognize a sequentializable proof structure, we have some *proof net correctness criteria*:

- Girard (empires);
- Danos-Regnier (switching);
- Guerrini (parsing).

Each proof net correctness criteria verify the correct application of inference rules verifying arities by means of topological properties of the associated labeled graph.

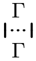
String diagram rewriting and Polygraphs

We present a diagram rewriting system $(\mathcal{S}_\Sigma, \mathcal{R}_\Sigma)$ by means of a 3-polygraph $\Sigma = (\Sigma_0, \Sigma_1, \Sigma_2, \Sigma_3)$:

Set	**String diagrams**	**Monoidal category**
Σ_0	Background labels	
Σ_1	String labels	Objects
Σ_2	The signature $\mathcal{S}_\Sigma$ (gate types)	Morphisms
Σ_3	The set of rewriting rules $\mathcal{R}_\Sigma$	Functors

We denote $\langle\Sigma\rangle$ monoidal category with objects the words over the alphabet Σ_1 and with morphisms the equivalence classes of diagrams generated by the signature Σ_2 modulo the equivalence relation generated by Σ_3.

We label string by formulas, that is $\Sigma_1 = \{Formulas\}$. Sequents are parallel compositions of strings, that are identities.



$$= \quad \vdash \Gamma$$

We label string by formulas, that is $\Sigma_1 = \{Formulas\}$. Sequents are parallel compositions of strings, that are identities.

$$\begin{array}{c}\Gamma\\ |\cdots|\\ \Gamma\end{array} \qquad = \qquad \vdash \Gamma$$

Gates are correspond to inference rules.

We label string by formulas, that is $\Sigma_1 = \{Formulas\}$. Sequents are parallel compositions of strings, that are identities.

$$\begin{array}{c}\Gamma\\ |\cdots|\\ \Gamma\end{array} \qquad = \qquad \vdash \Gamma$$

Gates are correspond to inference rules.

Rewriting rules corresponds to identities on derivations or cut-elimination.

We label string by formulas, that is $\Sigma_1 = \{Formulas\}$. Sequents are parallel compositions of strings, that are identities.

$$\begin{array}{c}\Gamma\\ |\cdots|\\ \Gamma\end{array} \qquad = \qquad \vdash \Gamma$$

Gates are correspond to inference rules.

Rewriting rules corresponds to identities on derivations or cut-elimination.

⚠ We need some morphisms in order to manage sequents since we use to consider sequents as multisets:

$$\vdash A, B \qquad = \qquad \vdash B, A$$

Control polygraphs for MLL_u

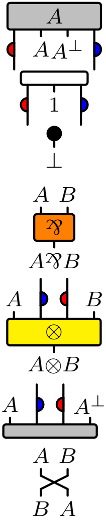
Definition

The *control polygraph of multiplicative linear logic with units* $\tilde{\mathfrak{U}}$ is given by the following sets of cells:

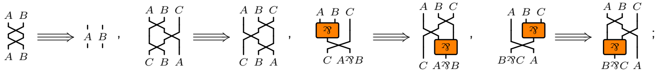
- $\tilde{\mathfrak{U}}_0 = \{ \square \}$;
- $\tilde{\mathfrak{U}}_1 = \mathfrak{F}_{M\ell\ell_u} \cup \{L = \quad, R = \quad\}$;

- $\tilde{\mathfrak{U}}_2 =$

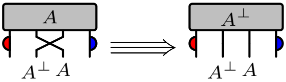
$$\left\{\begin{array}{lcccc}
Ax_A : & \square & \Rightarrow & L, A, A^\perp, R & = \\
1 : & \square & \Rightarrow & L, 1, R & = \\
\perp : & \square & \Rightarrow & \perp & = \\
⅋_{A,B} : & A, B & \Rightarrow & A ⅋ B & = \\
\otimes_{A,B} : & A, R, L, B & \Rightarrow & A \otimes B & = \\
Cut_A : & A, R, L, A^\perp & \Rightarrow & \square & = \\
\asymp_{A,B} : & A, B & \Rightarrow & B, A & =
\end{array}\right\}_{A,B \in \mathfrak{F}_{M\ell\ell_u}}$$



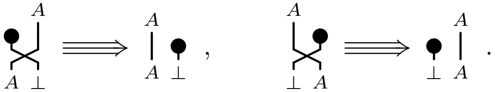
- $\tilde{\mathfrak{U}}_3 = \tilde{\mathfrak{M}}_{Twist} \cup \tilde{\mathfrak{U}}_{Twist}$ where:
    - $\tilde{\mathfrak{M}}_{Twist}$ is given by the following twisting relations:



together with one rule representing the involution $A^{\perp\perp} = A$:



- $\tilde{\mathfrak{U}}_{Twist}$ is given by the following twisting relations:



Theorem (Termination of $\tilde{\mathfrak{U}}$)

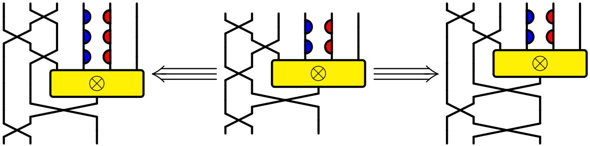
The polygraph $\tilde{\mathfrak{U}}$ is terminating.

Proposition ($\tilde{\mathfrak{U}}$ non-confluence)

The polygraph $\tilde{\mathfrak{U}}$ is not confluent.

Proof.

In $\tilde{\mathfrak{U}}$ the following critical pair is not confluent:



□

Theorem (Controlled proof diagram correspondence in $\tilde{\mathfrak{U}}$)

$$\vdash_{MLL_u} \Gamma \Leftrightarrow \exists \phi \in \tilde{\mathfrak{U}} \textit{ such that } \phi : \square \Rightarrow L, \Gamma, R.$$

Definition (Representation)

We say that a proof diagram $\phi \in \mathfrak{U}$ with $\phi : \square \Rightarrow L, \Gamma, R$ *represents* a derivation $d(\Gamma)$ if it can be sequentialized into the derivation $d(\Gamma)$.
We say that a derivation $d(\Gamma)$ is represented by ϕ or that ϕ is a *diagrammatic representation of* $d(\Gamma)$ if ϕ can be sequentialized into the derivation $d(\Gamma)$.

Equivalences over MLL_u derivations

- We denote $N_{d(\Gamma)}$ the proof net representing the derivation $d(\Gamma)$, we denote $\sim_N$ the equivalence relation over derivations induced by proof nets syntax. It is defined as follows:

$$d'(\Gamma) \sim_N d''(\Gamma) \text{ iff } N_{d'(\Gamma)} = N_{d''(\Gamma)}.$$

In other words, $d'(\Gamma) \sim_N d''(\Gamma)$ if and only if they can be represented by the same proof net (i.e. iff $N_{d'(\Gamma)}$ and $N_{d''(\Gamma)}$ are isomorph labeled graphs).

- We denote $\simeq_D$ the equivalence relation over derivations induced by proof diagram syntax. It is defined as follows:

$$d'(\Gamma) \simeq_D d''(\Gamma) \text{ iff } \exists \phi \in \tilde{\mathfrak{U}} \text{ such that } \phi_{d'(\Gamma)} = \phi = \phi_{d''(\Gamma)}.$$

- We denote $\sim_{\tilde{D}}$ the equivalence relation over derivations induced by $\langle \tilde{\mathfrak{U}} \rangle$. It is defined as follows:

$$d'(\Gamma) \sim_{\tilde{D}} d''(\Gamma) \text{ iff } \exists \phi_{d'(\Gamma)}, \phi_{d''(\Gamma)} \in \tilde{\mathfrak{U}} \text{ s.t. } [\phi_{d'(\Gamma)}]_{\tilde{\mathfrak{U}}} = [\phi_{d''(\Gamma)}]_{\tilde{\mathfrak{U}}}.$$

We have the following inclusions:

- $\sim_N \subsetneq \sim$ since $\bot$ permutations changing jump assignations are not captured by the proof net syntactical equivalence;
- $\sim_{\tilde{D}} \subsetneq \sim$ since binary rules permutation which changes branching order are not captured by the proof diagrams equivalence.

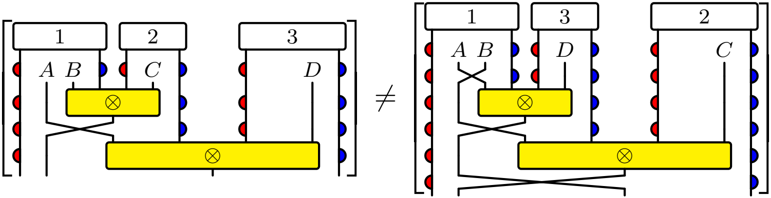
$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, \Delta, \Sigma\end{array}}{\vdash \odot_1(\Gamma), \Delta, \Sigma}\odot_1}{\vdash \odot_1(\Gamma), \odot_2(\Delta), \Sigma}\odot_2 \quad \sim \quad \dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, \Delta, \Sigma\end{array}}{\vdash \Gamma, \odot_2(\Delta), \Sigma}\odot_2}{\vdash \odot_1(\Gamma), \odot_2(\Delta), \Sigma}\odot_1$$

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Delta, A\end{array} \quad \begin{array}{c}\vdots\\ \vdash B, \Gamma, \Sigma\end{array}}{\vdash \Delta, \odot_1(A,B), \Gamma, \Sigma}\odot_1}{\vdash \Delta, \odot_1(A,B), \odot_2(\Gamma), \Sigma}\odot_2 \sim \dfrac{\begin{array}{c}\vdots\\ \vdash \Delta, A\end{array} \quad \dfrac{\begin{array}{c}\vdots\\ \vdash B, \Gamma, \Sigma\end{array}}{\vdash \Delta, B, \odot_2(\Gamma), \Sigma}\odot_2}{\vdash \Delta, \odot_1(A,B), \odot_2(\Gamma), \Sigma}\odot_1 \quad \dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, A, \Sigma\end{array} \quad \begin{array}{c}\vdots\\ \vdash B, \Delta\end{array}}{\vdash \Gamma, \odot_1(A,B), \Delta, \Sigma}\odot_1}{\vdash \odot_2(\Gamma), \odot_1(A,B), \Delta, \Sigma}\odot_2 \sim \dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, A, \Sigma\end{array}}{\vdash \odot_2(\Gamma), A, \Sigma}\odot_2 \quad \begin{array}{c}\vdots\\ \vdash B, \Delta\end{array}}{\vdash \odot_2(\Gamma), \odot_1(A,B), \Delta, \Sigma}\odot_1$$

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, A\end{array} \quad \begin{array}{c}\vdots\\ \vdash \Sigma, B, C\end{array}}{\vdash \Gamma, \Sigma, \odot_1(A,B), C}\odot_1 \quad \begin{array}{c}\vdots\\ \vdash \Delta, D\end{array}}{\vdash \Gamma, \Sigma, \Delta, \odot_1(A,B), (C \odot_2 D)}\odot_2 \quad \sim \quad \dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, A\end{array} \quad \dfrac{\begin{array}{c}\vdots\\ \vdash \Sigma, B, C\end{array} \quad \begin{array}{c}\vdots\\ \vdash \Delta, D\end{array}}{\vdash \Sigma, \Delta, B, \odot_2(C,D)}\odot_2}{\vdash \Gamma, \Sigma, \Delta, \odot_1(A,B), \odot_2(C,D)}\odot_1$$

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, A, C\end{array} \quad \begin{array}{c}\vdots\\ \vdash \Sigma, B\end{array}}{\vdash \Gamma, \Sigma, \odot_1(A,B), C}\odot_1 \quad \begin{array}{c}\vdots\\ \vdash \Delta, D\end{array}}{\vdash \Gamma, \Sigma, \Delta, \odot_1(A,B), (C \odot_2 D)}\odot_2 \sim \dfrac{\dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, A, C\end{array} \quad \begin{array}{c}\vdots\\ \vdash \Delta, D\end{array}}{\vdash \Gamma, \Delta, A, \odot_2(C,D)}\odot_2 \quad \begin{array}{c}\vdots\\ \vdash \Sigma, B\end{array}}{\vdash \Gamma, \Sigma, \Delta, \odot_1(A,B), (C \odot_2 D)}\odot_1 \quad \dfrac{\begin{array}{c}\vdots\\ \vdash \Sigma, C\end{array} \quad \dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, A\end{array} \quad \begin{array}{c}\vdots\\ \vdash \Delta, D, B\end{array}}{\vdash \Gamma, \Delta, D, \odot_1(A,B)}\odot_1}{\vdash \Gamma, \Sigma, \Delta, \odot_1(A,B), \odot_2(C,D)}\odot_2 \sim \dfrac{\begin{array}{c}\vdots\\ \vdash \Gamma, A\end{array} \quad \dfrac{\begin{array}{c}\vdots\\ \vdash \Sigma, C\end{array} \quad \begin{array}{c}\vdots\\ \vdash \Delta, D, B\end{array}}{\vdash \Sigma, \Delta, B, \odot_2(C,D)}\odot_2}{\vdash \Gamma, \Sigma, \Delta, \odot_1(A,B), \odot_2(C,D)}\odot_1$$

The equivalence $\simeq_D$ does not capture the case of permutation in which binary rules have different branching order:



while the corresponding represented proof are $\sim$-equivalent:

$$\dfrac{\dfrac{\overset{1}{\vdots} \atop \vdash A, B \qquad \overset{2}{\vdots} \atop \vdash C}{\vdash A, B \otimes C}\otimes \qquad \overset{3}{\vdots} \atop \vdash D}{\vdash A \otimes D, B \otimes C}\otimes \quad \sim \quad \dfrac{\dfrac{\overset{1}{\vdots} \atop \vdash A, B \qquad \overset{3}{\vdots} \atop \vdash D}{\vdash A \otimes D, B}\otimes \qquad \overset{2}{\vdots} \atop \vdash C}{\vdash A \otimes D, B \otimes D}\otimes$$

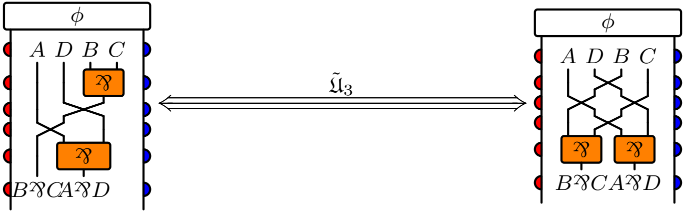
Proposition

The equivalence relation $\sim_{\tilde{D}}$ is equivalent to $\simeq_D$.

For example:

$$\dfrac{\dfrac{\dfrac{\vdots}{\vdash A, D, B, C}}{\vdash A, D, B \parr C}\,\parr}{\dfrac{\vdash B \parr C, A, D}{\vdash B \parr C, A \parr D}\,\parr}\,\sigma \quad = \quad \dfrac{\dfrac{\dfrac{\vdots}{\vdash A, D, B, C}}{\vdash B \parr C, A, D}\,\parr}{\vdash B \parr C, A \parr D}\,\parr \quad \sim \quad \dfrac{\dfrac{\dfrac{\vdots}{\vdash A, D, B, C}}{\vdash A \parr D, B, C}\,\parr}{\vdash B \parr C, A \parr D}\,\parr \quad = \quad \dfrac{\dfrac{\dfrac{\dfrac{\vdots}{\vdash A, D, B, C}}{\vdash B, C, A, D}\,\sigma}{\vdash B, C, A \parr D}\,\parr}{\vdash B \parr C, A \parr D}\,\parr$$

$\downarrow$ $\qquad\qquad\qquad\qquad\qquad\qquad$ $\downarrow$



- Extend the works on *"Rewriting modulo symmetric monoidal structures"* in order to internalize wire crossings;
- Use proof diagrams in order to study proof equivalence complexity of different sequent calculi (MELL, CyLL, MALL, modal logics, LK);
- Study 2-dimensional representations of non-commutative logics proof nets;
- ...

- The use of 2-dimensional syntax is suitable to express a notion of non-consequentiality without requiring a notion of contemporaneity;
- This notion plays an important role in any definition of proof equivalence compatible with cut elimination, especially in case of reversible inference rules;
- More in general, this notion is becoming prominent (and in some cases essential) in the formalization of many different science fields.

"We have taken the word, the sentence, logic and number as the foundation stones of our civilisation, forcing our brains to use limiting modes of expression which we assume are the only correct ones."

[Tony Buzan, *The mind map book*, 1993]

Thank you for the attention

"We have taken the word, the sentence, logic and number as the foundation stones of our civilisation, forcing our brains to use limiting modes of expression which we assume are the only correct ones."

[Tony Buzan, *The mind map book*, 1993]

Thank you for the attention

"We have taken the word, the sentence, logic and number as the foundation stones of our civilisation, forcing our brains to use limiting modes of expression which we assume are the only correct ones."

[Tony Buzan, *The mind map book*, 1993]

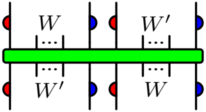
Thank you for the attention

Questions?

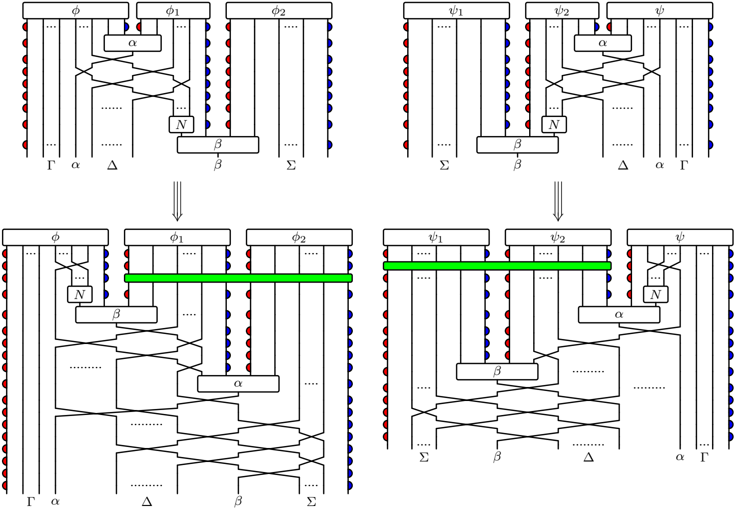
The polygraph of MLL_u proof diagrams

Definition (Polygraph of diagrammatic MLL_u proof diagrams)

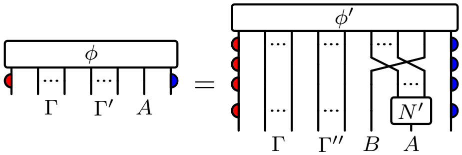
The *polygraph of diagrammatic MLL_u proof nets* is the polygraph obtained extended the polygraph $\mathfrak{U}$ with the following cells:

- $\mathfrak{U}_0 = \tilde{\mathfrak{U}}_0$;
- $\mathfrak{U}_1 = \tilde{\mathfrak{U}}_1$;
- $\mathfrak{U}_2 = \tilde{\mathfrak{U}}_2 \cup \mathfrak{B}ig = \Bigg\{ B_{W,W} =$  $\Bigg\}_{W,W' \in (\mathfrak{F}_{M\ell\ell_u} \cup \{L,R\})^*}$;

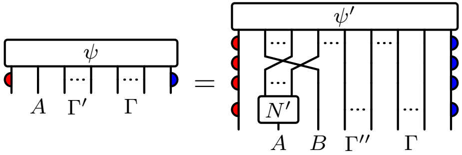
- $\mathfrak{U}_3 = \tilde{\mathfrak{U}}_3 \cup \mathfrak{U}_{Big}$ where $\mathfrak{U}_{Big}$ is made of the following 3-cells:
  $\mathfrak{B}$-introduction:



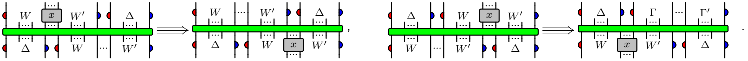
where ϕ and ψ are of the form



and

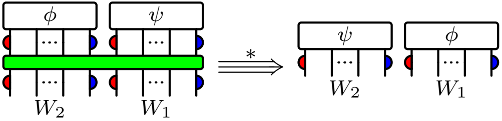


The untangle relations: for any $\boxed{x} \in \tilde{\mathfrak{U}}_2$

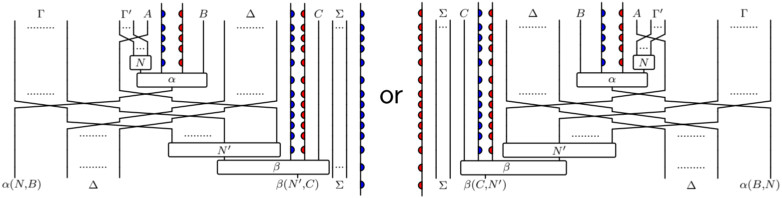


Proposition

If $\phi \in \mathfrak{U}$ is a proof diagram $\phi : \square \Rightarrow L, \Gamma, R$ containing a B-gate, then there is a rewriting sequence made by untangle relations of the form



B-introduction rules are applied on configurations of the form



where α, β are *splitting gates*, that are gates of type $\otimes$ or Cut.
For example, consider the two following configurations with A active formula of α:

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma_1 \quad \Gamma_B \quad \Gamma_2 \quad \boxed{\Gamma_A}}{\vdash \Gamma, \boxed{\Gamma'_A}, B} \qquad \vdash C, \Delta}{\vdash \Gamma, \Delta, \boxed{\Gamma'_A}, \beta(B,C)}\,\beta(B,C)}{\vdash \Gamma', \Delta', \boxed{A}, \beta(B,C)} \qquad \vdash D, \Sigma}{\vdash \Gamma', \Delta', \beta(B,C), \alpha(\boxed{A}, D), \Sigma}\,\alpha(A,D)$$

or

$$\dfrac{\vdash \Sigma, D \qquad \dfrac{\dfrac{\vdash \Delta, C \qquad \dfrac{\boxed{\Gamma_A} \quad \Gamma_1 \quad \Gamma_B \quad \Gamma_2}{\vdash \Gamma, \boxed{\Gamma'_A}, B}}{\vdash \Delta, \beta(B,C), \boxed{\Gamma'_A}, \Gamma}\,\beta(C,B)}{\vdash \beta(B,C), \boxed{A}, \Gamma', \Delta'}}{\vdash \Sigma, \beta(B,C), \alpha(D, \boxed{A}), \Gamma'\Delta'}\,\alpha(D,A)$$

Theorem (Termination of $\mathfrak{U}$)

The polygraph $\mathfrak{U}$ is terminating.

In particular, if $\phi \in \mathfrak{U}$ is irreducible, then $\phi \in \tilde{\mathfrak{U}}$.

Even if the polygraph $\mathfrak{U}$ is not confluent, the rewriting concerning only B-gates elimination is:

Proposition

The order of the Ax and 1 gates (derivation tree leafs) is the same for every $\mathfrak{U}_3$-equivalent irreducible diagram in $\mathfrak{U}$.

A denotational semantics for MLL_u

We define the following equivalence relation:

$$d'(\Gamma) \sim_D d''(\Gamma) \text{ iff } \exists \phi_{d'(\Gamma)}, \phi_{d''(\Gamma)} \in \tilde{\mathfrak{U}} \text{ s.t. } [\phi_{d'(\Gamma)}]_{\mathfrak{U}} = [\phi_{d''(\Gamma)}]_{\mathfrak{U}}.$$

Theorem

Two derivations are equivalent modulo $\sim$ if and only if they are represented by two equivalent proof diagrams with respect of $\langle \mathfrak{U} \rangle$. That is:

$$d(\Gamma) \sim d'(\Gamma) \Leftrightarrow d(\Gamma) \sim_D d'(\Gamma)$$

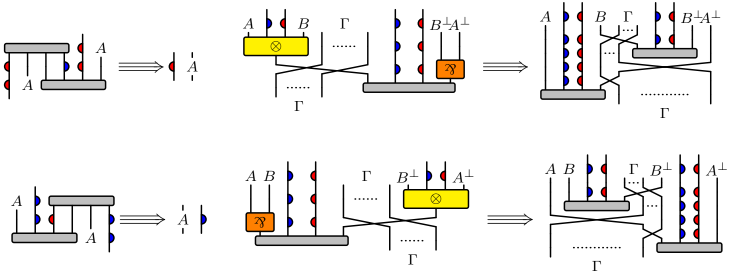
Thus the following function is well defined:

$$\begin{array}{rccc} [-]_{\mathfrak{U}} : & \{MLL_u \textit{ proofs (mod } \sim)\} & \to & \{ \textit{ morphisms in } \langle \mathfrak{U} \rangle\} \\ & d(\Gamma) & \to & [\phi_{d(\Gamma)}]_{\mathfrak{U}} \end{array}$$

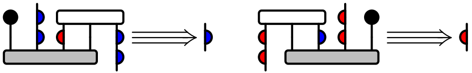
Definition (Polygraph of MLL_u semantics)

The *polygraph of multiplicative linear logic semantics* $\mathcal{S}_{MLL_u}$ is given by $\mathfrak{U}$ enriched with the set of 3-cells $\mathcal{S}^{Cut}_{MLL_u} = \mathfrak{M}_{Cut} \cup \mathfrak{U}_{Cut}$:

- $\mathfrak{M}_{Cut}$ is made of the following 3-cells:



- $\mathfrak{U}_{Cut}$ is made of the following 3-cells:



Theorem (Termination in $\mathcal{S}_{MLL_u}$)

The polygraph $\mathcal{S}_{MLL_u}$ is terminating.

Consequently, we have a *cut-elimination* Theorem for the set of sequentializable proof diagrams in $\mathcal{S}_{MLL_u}$.

Theorem (Cut-elimination)

An irreducible proof diagram $\phi \in \mathcal{S}_{MLL_u}$ which represent a derivation contains no Cut-gates.

Theorem (Multiplicative linear logic correspondence)

$$\vdash_{MLL_u} \Gamma \Leftrightarrow \exists \phi \in \mathcal{S}_{MLL_u} \textit{ such that } \phi : \square \Rightarrow L, \Gamma, R.$$

We define the following function associating to any MLL_u derivation $d(\Gamma)$ a morphism of the category $\langle \mathcal{S}_{MLL_u} \rangle$ as follows:

Definition (Denotational semantics of proof diagrams)

$$\begin{array}{llll} [-]_D : & \{MLL_u \text{ derivations}\} & \to & \{ \text{ morphisms in } \langle \mathcal{S}_{MLL_u} \rangle \} \\ & d(\Gamma) & \to & [d(\Gamma)]_D = [\phi_{d(\Gamma)}]_{\mathcal{S}_{MLL_u}} \end{array}$$

where $\phi_{d(\Gamma)}$ is an arbitrary representation of $d(\Gamma)$.

Theorem (Proof diagram semantics)

$[-]_D$ is a denotational semantics for MLL_u sequent calculus.

Proof.

We define the following equivalence relation $\approx_D$ over MLL_u derivations:

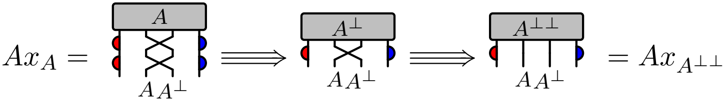
$$d'(\Gamma) \approx_D d''(\Gamma) \text{ iff } [d'(\Gamma)]_D = [d''(\Gamma)]_D$$

We have the following properties:

1. *if $d(\Gamma) \to_{Cut} \hat{d}(\Gamma)$, then $d(\Gamma) \approx_D \hat{d}(\Gamma)$;*
2. *$\approx_D$ is non-degenerated, i.e. one can find a formula with at least two non-equivalent proofs;*
3. *$\approx_D$ is a congruence;*

$\square$

We remark that $[-]_D$ is coherent with the involutivity of negation. In fact, the invariance of diagram inputs and outputs with respect to rewriting impose the equivalence $A^{\perp\perp} = A$:



Similarly, De Morgan's laws follow by the definition of Cut-gates. By means of example, consider the equivalence of $A \parr B = (B^\perp \otimes A^\perp)^\perp$: