MELL proof-nets in the category of graphs

Giulio Guerrieri[1] Giulia Manara[2] Luc Pellissier[3]
Lorenzo Tortora de Falco[4] Lionel Vaux Auclair[2]

[1]Department of Computer Science, University of Bath, UK

[2]Aix-Marseille Univ, CNRS, Centrale Marseille, I2M, Marseille, France

[3]Université Paris Est Créteil, LACL, F-94010 Créteil, France

[4]Università Roma Tre, Dipartimento di Matematica e Fisica, Rome, Italy

Trends in Linear Logic and Applications (TLLA 2021)
Virtual, 27 June 2021

Outline

1. MELL proof-nets as graphs

2. What for? MELL proof-nets in the category of graphs

Outline

1. MELL proof-nets as graphs
2. What for? MELL proof-nets in the category of graphs

Boxes in MELL

Multiplicative exponential linear logic (MELL):

$$A, B ::= X \mid X^{\perp} \mid A \otimes B \mid A \parr B \mid 1 \mid \perp \mid !A \mid ?A$$

MELL sequent calculus $\rightsquigarrow$ MELL proof-nets ($\approx$ labeled directed graphs).

The promotion rule is the only contextual rule in MELL sequent calculus.

$$\dfrac{\begin{array}{c}\pi \\ \vdots \\ \vdash ?\Gamma, A\end{array}}{\vdash ?\Gamma, !A}\,! \quad \Rightarrow$$

In MELL proof-nets, boxes are reminiscent of the sequentiality of promotion.

A box marks a subproof that can be erased or duplicated at will by cut-elimination.
$\rightsquigarrow$ Boxes encode crucial information in MELL proof-nets.

Boxes in MELL

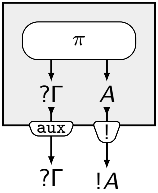
Multiplicative exponential linear logic (MELL):

$$A, B ::= X \mid X^{\perp} \mid A \otimes B \mid A \parr B \mid 1 \mid \perp \mid !A \mid ?A$$

MELL sequent calculus $\leadsto$ MELL proof-nets ($\approx$ labeled directed graphs).

The promotion rule is the only contextual rule in MELL sequent calculus.

$$\dfrac{\begin{array}{c}\pi \\ \vdots \\ \vdash ?\Gamma, A\end{array}}{\vdash ?\Gamma, !A}\,! \qquad \Rightarrow$$



In MELL proof-nets, boxes are reminiscent of the sequentiality of promotion.

A box marks a subproof that can be erased or duplicated at will by cut-elimination.
$\leadsto$ Boxes encode crucial information in MELL proof-nets.

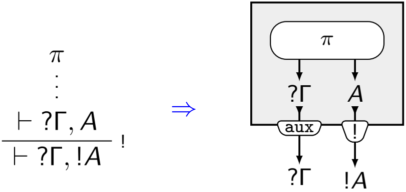
Boxes in MELL

Multiplicative exponential linear logic (MELL):

$$A, B ::= X \mid X^{\perp} \mid A \otimes B \mid A \parr B \mid 1 \mid \perp \mid !A \mid ?A$$

MELL sequent calculus $\rightsquigarrow$ MELL proof-nets ($\approx$ labeled directed graphs).

The promotion rule is the only contextual rule in MELL sequent calculus.

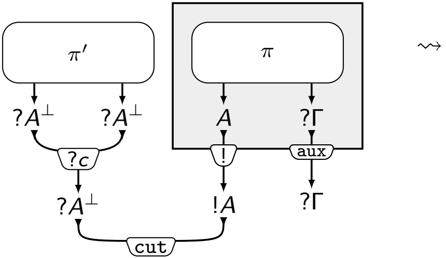


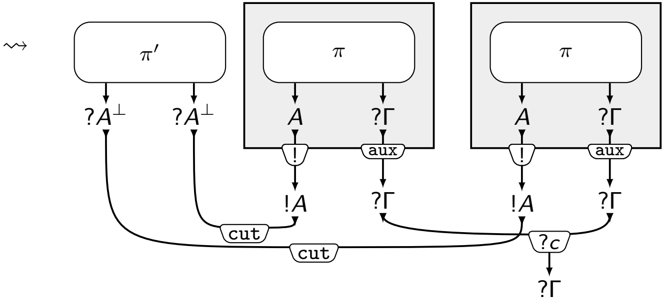
In MELL proof-nets, boxes are reminiscent of the sequentiality of promotion.

A box marks a subproof that can be erased or duplicated at will by cut-elimination.
$\rightsquigarrow$ Boxes encode crucial information in MELL proof-nets.

Exponential cut-elimination steps

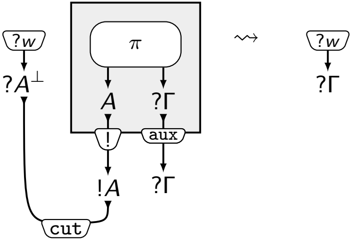
contraction/promotion: duplication of a resource





Exponential cut-elimination steps

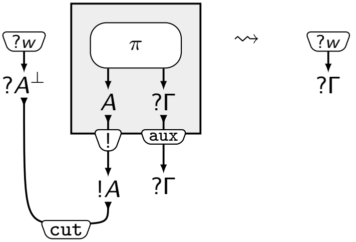
weakening/promotion: erasure of a resource



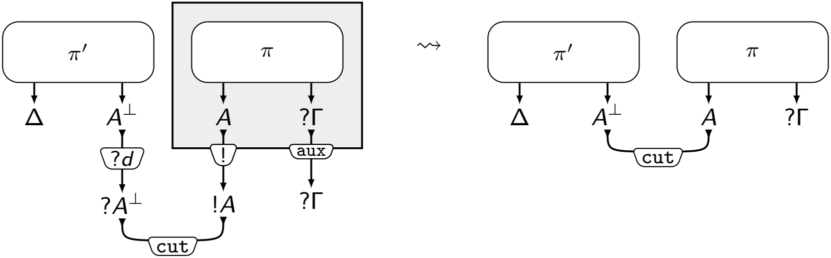
dereliction/promotion: access to a resource

Exponential cut-elimination steps

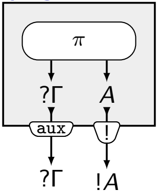
weakening/promotion: erasure of a resource



dereliction/promotion: access to a resource



MELL proof-nets are really graphs?



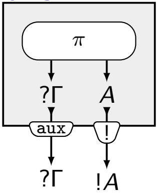
How do we identify boxes? We have to get out of standard graph representations.

In the literature, there are several solutions:

1. A MELL proof-net is a directed graph + something to identify boxes
   - something can be provided informally ⇝ not rigorous, hand-waving;
   - something can be provided formally ⇝ *ad hoc* and tricky, not handy.
2. A MELL proof-net is an inductive directed graph, where with any node !

is associated another inductive directed graph (the content of the box).
⇝ Girard's idea of proofs as graphs in watered-down.

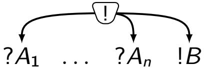
MELL proof-nets are really graphs?



How do we identify boxes? We have to get out of standard graph representations.

In the literature, there are several solutions:

1. A MELL proof-net is a directed graph + something to identify boxes
   - something can be provided informally ⇝ not rigorous, hand-waving;
   - something can be provided formally ⇝ *ad hoc* and tricky, not handy.
2. A MELL proof-net is an inductive directed graph, where with any node !



is associated another inductive directed graph (the content of the box).
⇝ Girard's idea of proofs as graphs in watered-down.

Our goal

Small goal:
Can we recover the border of a boxes using only usual notions from graph theory?

Big goal:
Having a canonical and not *ad-hoc* syntax for MELL proof-nets is important!

1. To answer the questions: What is a proof? When are two proofs identical?
2. To have proofs about properties of proof-nets that are not hand-waving.
3. To define sophisticated operation on proof-nets.
4. To hope to formalize proof-nets in proof-assistants.

Our contribution achieves the small goal, a first step towards the big goal.

Our goal

Small goal:
Can we recover the border of a boxes using only usual notions from graph theory?

Big goal:
Having a canonical and not *ad-hoc* syntax for MELL proof-nets is important!

1. To answer the questions: What is a proof? When are two proofs identical?
2. To have proofs about properties of proof-nets that are not hand-waving.
3. To define sophisticated operation on proof-nets.
4. To hope to formalize proof-nets in proof-assistants.

Our contribution achieves the small goal, a first step towards the big goal.

Our goal

Small goal:
Can we recover the border of a boxes using only usual notions from graph theory?

Big goal:
Having a canonical and not *ad-hoc* syntax for MELL proof-nets is important!

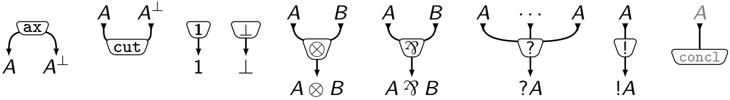
1. To answer the questions: What is a proof? When are two proofs identical?
2. To have proofs about properties of proof-nets that are not hand-waving.
3. To define sophisticated operation on proof-nets.
4. To hope to formalize proof-nets in proof-assistants.

Our contribution achieves the small goal, a first step towards the big goal.

MELL proof-nets as graphs

Definition: An MELL proof-net is a triple $R = (|R|, \mathcal{A}, \text{box})$ such that:

1. $|R|$ is a directed graph (without boxes) labeled by MELL formulas, such that



   and the premises of each $\parr$- or $\otimes$-node are ordered;

2. $\mathcal{A}$ is a rooted tree (its transitive-reflexive closure is denoted by $\mathcal{A}^{\circlearrowleft}$);
3. $box\colon |R| \to \mathcal{A}^{\circlearrowleft}$ is a graph homomorphism such that
   - it induces a bijection between premises of !-nodes in $|R|$ and edges in $\mathcal{A}$,
   - for any edge from v' to v in $|R|$, if $box(v') \neq box(v)$ then v is labeled by ! or ?.

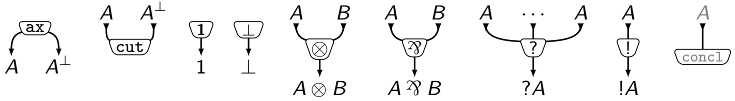
Remark: The last two conditions for *box* express the fact that

1. the rooted tree $\mathcal{A}$ represents the nesting of the boxes of R,
2. the border of a box is given by nodes labeled by ! or ?.

MELL proof-nets as graphs

Definition: An MELL proof-net is a triple $R = (|R|, \mathcal{A}, \text{box})$ such that:

1. $|R|$ is a directed graph (without boxes) labeled by MELL formulas, such that



and the premises of each ⅋- or ⊗-node are ordered;

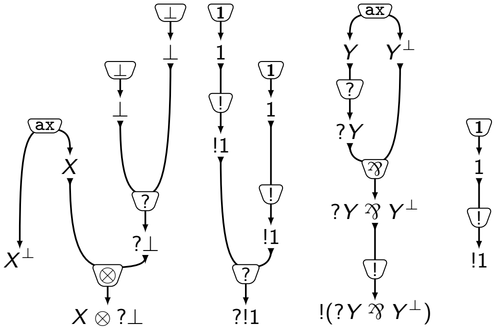
2. $\mathcal{A}$ is a rooted tree (its transitive-reflexive closure is denoted by $\mathcal{A}^{\circlearrowleft}$);
3. $box\colon |R| \to \mathcal{A}^{\circlearrowleft}$ is a graph homomorphism such that
   - it induces a bijection between premises of !-nodes in $|R|$ and edges in $\mathcal{A}$,
   - for any edge from v' to v in $|R|$, if $box(v') \neq box(v)$ then v is labeled by ! or ?.

Remark: The last two conditions for *box* express the fact that

1. the rooted tree $\mathcal{A}$ represents the nesting of the boxes of R,
2. the border of a box is given by nodes labeled by ! or ?.

Example of an MELL proof-net

An MELL proof-structure $R = (|R|, \mathcal{A}, box)$: ($box\colon |R| \to \mathcal{A}^{\circlearrowleft}$ is given by colors)



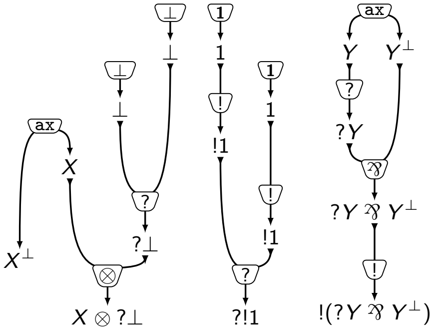
(a) The underlying graph $|R|$ (the information about boxes is extracted from box).

(b) The box-tree $\mathcal{A}$ (without dotted lines) and the reflexive-transitive closure $\mathcal{A}^{\circlearrowleft}$ of $\mathcal{A}$ (with also dotted lines).

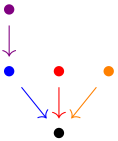
There is no explicit constructor for boxes in $|R|$!

Example of an MELL proof-net

An MELL proof-structure $R = (|R|, \mathcal{A}, box)$: $(box\colon |R| \to \mathcal{A}^{\circlearrowleft}$ is given by colors)



(a) The underlying graph $|R|$ (the information about boxes is extracted from box).

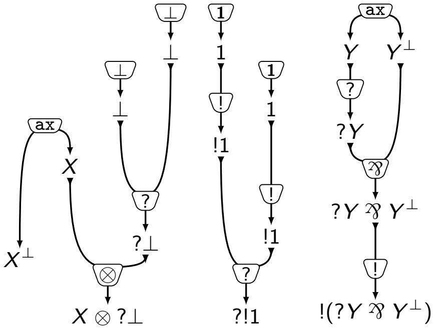


(b) The box-tree $\mathcal{A}$ (without dotted lines) and the reflexive-transitive closure $\mathcal{A}^{\circlearrowleft}$ of $\mathcal{A}$ (with also dotted lines).

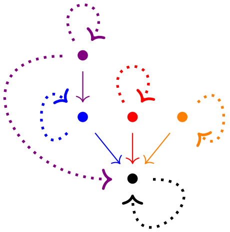
There is no explicit constructor for boxes in $|R|$!

Example of an MELL proof-net

An MELL proof-structure $R = (|R|, \mathcal{A}, box)$: ($box\colon |R| \to \mathcal{A}^{\circlearrowleft}$ is given by colors)



(a) The underlying graph $|R|$ (the information about boxes is extracted from *box*).

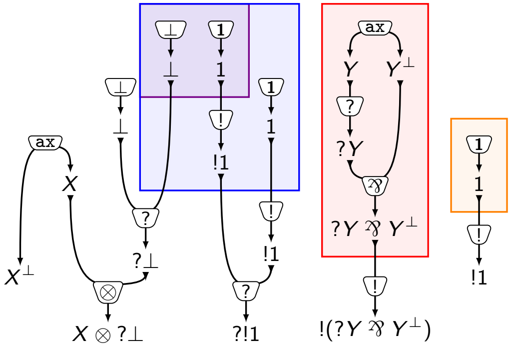


(b) The box-tree $\mathcal{A}$ (without dotted lines) and the reflexive-transitive closure $\mathcal{A}^{\circlearrowleft}$ of $\mathcal{A}$ (with also dotted lines).

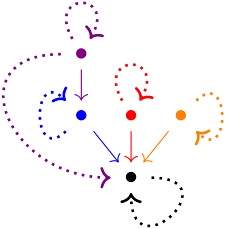
There is no explicit constructor for boxes in $|R|$!

Example of an MELL proof-net

An MELL proof-structure $R = (|R|, \mathcal{A}, box)$: ($box\colon |R| \to \mathcal{A}^{\circlearrowleft}$ is given by colors)



(a) The underlying graph $|R|$ (the information about boxes is extracted from *box*).

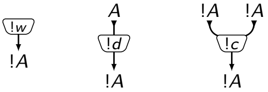


(b) The box-tree $\mathcal{A}$ (without dotted lines) and the reflexive-transitive closure $\mathcal{A}^{\circlearrowleft}$ of $\mathcal{A}$ (with also dotted lines).

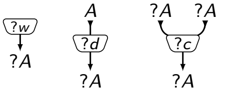
There is no explicit constructor for boxes in $|R|$!

A generalization: differential linear logic

Differential Linear Logic: In addition to promotion, three more rules introduce the modality ! (perfectly symmetric to structural rules) $\Rightarrow$ DiLL extends MELL.



(the duals of

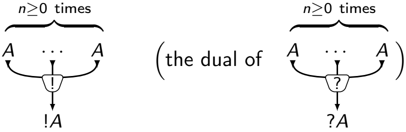


)

T. Ehrhard, L. Regnier. *Differential Interaction Nets*. TCS, 2006.

A generalization: differential linear logic

Differential Linear Logic: In addition to promotion, three more rules introduce the modality ! (perfectly symmetric to structural rules) $\Rightarrow$ DiLL extends MELL.

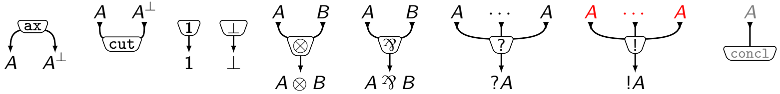


T. Ehrhard, L. Regnier. *Differential Interaction Nets*. TCS, 2006.

DiLL proof-nets as graphs

Definition: An DiLL proof-net is a triple $R = (|R|, \mathcal{A}, \text{box})$ such that:

1. $|R|$ is a directed graph (without boxes) labeled by MELL formulas, such that



   and the premises of each ⅋- or ⊗-node are ordered;

2. $\mathcal{A}$ is a rooted tree (its transitive-reflexive closure is denoted by $\mathcal{A}^{\circlearrowleft}$);

3. $box\colon |R| \to \mathcal{A}^{\circlearrowleft}$ is a graph homomorphism such that
   - it induces a partial bijection from premises of !-nodes in $|R|$ to edges in $\mathcal{A}$,
   - for any edge from v' to v in $|R|$, if $box(v') \neq box(v)$ then v is labeled by ! or ?.

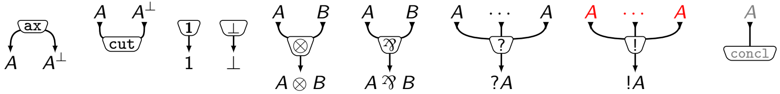
Remark: In a DiLL proof-net, a node labeled by !

1. may have several premises,
2. not all of its premises are necessarily associated with a box.

DiLL proof-nets as graphs

Definition: An DiLL proof-net is a triple $R = (|R|, \mathcal{A}, \text{box})$ such that:

1. $|R|$ is a directed graph (without boxes) labeled by MELL formulas, such that



and the premises of each ⅋- or ⊗-node are ordered;

2. $\mathcal{A}$ is a rooted tree (its transitive-reflexive closure is denoted by $\mathcal{A}^{\circlearrowleft}$);
3. $box\colon |R| \to \mathcal{A}^{\circlearrowleft}$ is a graph homomorphism such that
   - it induces a partial bijection from premises of !-nodes in $|R|$ to edges in $\mathcal{A}$,
   - for any edge from v' to v in $|R|$, if $box(v') \neq box(v)$ then v is labeled by ! or ?.

Remark: In a DiLL proof-net, a node labeled by !

1. may have several premises,
2. not all of its premises are necessarily associated with a box.

When are two DiLL proof-nets identical?

Remark: Our definition of proof-nets relies on standard notions from graph theory.

Pros of this purely graphical approach: definitions are

- not *ad hoc*, more “canonical”,
- more “manageable”, using standard mathematical notions.

Example: Identity of proof-nets = graph isomorphism that commutes with *box*.

Definition: Let $R = (|R|, \mathcal{A}, box)$ and $R' = (|R'|, \mathcal{A}', box')$ be DiLL proof-nets. An isomorphism from R to R' is a couple $(f_{|\cdot|}, f_{\text{tree}})$ where:

- $f_{|\cdot|} \colon |R| \to |R'|$ is an isomorphism of labeled directed graphs,
- $f_{\text{tree}} \colon \mathcal{A} \to \mathcal{A}'$ is an isomorphism of rooted trees, and the diagram commutes

$$\begin{array}{ccc} |R| & \xrightarrow{box} & \mathcal{A}^{\circlearrowleft} \\ \downarrow f_{|\cdot|} & & \downarrow f^{\circlearrowleft}_{\text{tree}} \\ |R'| & \xrightarrow{box'} & \mathcal{A}'^{\circlearrowleft} \end{array}$$

When are two DiLL proof-nets identical?

Remark: Our definition of proof-nets relies on standard notions from graph theory.

Pros of this purely graphical approach: definitions are

- not *ad hoc*, more "canonical",
- more "manageable", using standard mathematical notions.

Example: Identity of proof-nets = graph isomorphism that commutes with *box*.

Definition: Let $R = (|R|, \mathcal{A}, box)$ and $R' = (|R'|, \mathcal{A}', box')$ be DiLL proof-nets.
An isomorphism from R to R' is a couple $(f_{|\cdot|}, f_{\text{tree}})$ where:

- $f_{|\cdot|} \colon |R| \to |R'|$ is an isomorphism of labeled directed graphs,
- $f_{\text{tree}} \colon \mathcal{A} \to \mathcal{A}'$ is an isomorphism of rooted trees, and the diagram commutes

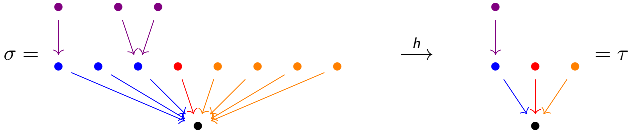
$$\begin{array}{ccc} |R| & \xrightarrow{box} & \mathcal{A}^{\circlearrowleft} \\ \downarrow{\scriptstyle f_{|\cdot|}} & & \downarrow{\scriptstyle f_{\mathbf{tree}}^{\circlearrowleft}} \\ |R'| & \xrightarrow{box'} & \mathcal{A}'^{\circlearrowleft} \end{array}$$

Outline

Exploiting the box-tree of DiLL proof-nets: thick subtrees

Definition: Let τ be a rooted tree. A thick subtree of τ is a pair (σ, h) of a rooted tree σ and a graph homomorphism $h\colon \sigma \to \tau$.

Example:



Idea: Let $R = (|R|, \mathcal{A}, box)$ be a DiLL proof-net. A thick subtree $(\mathcal{A}', h)$ of $\mathcal{A}$

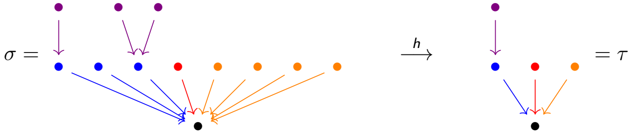
- says how many copies to take for each box, iteratively;
- "generates" a new DiLL proof-net $R' = (|R'|, \mathcal{A}', box')$.

Which one? What are $|R'|$ and box'?

Exploiting the box-tree of DiLL proof-nets: thick subtrees

Definition: Let τ be a rooted tree. A thick subtree of τ is a pair (σ, h) of a rooted tree σ and a graph homomorphism $h\colon \sigma \to \tau$.

Example:



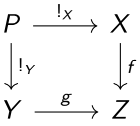
Idea: Let $R = (|R|, \mathcal{A}, box)$ be a DiLL proof-net. A thick subtree $(\mathcal{A}', h)$ of $\mathcal{A}$

- says how many copies to take for each box, iteratively;
- "generates" a new DiLL proof-net $R' = (|R'|, \mathcal{A}', box')$.

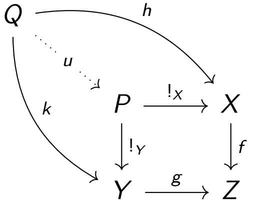
Which one? What are $|R'|$ and box'?

Pullbacks in category theory

Definition: Let $\mathcal{C}$ be a category, $X, Y, Z \in \mathcal{C}$ and $f\colon X \to Z$ and $g\colon Y \to Z$.
A pullback of f and g is a triple $(P, !_X, !_Y)$ such that (1) commutes and, for any $(Q, h : Q \to X, k : Q \to Y)$ making (1) commute, there is a unique arrow $u : Q \to P$ factorizing h and k, i.e. such that diagram (2) commutes.



(1)



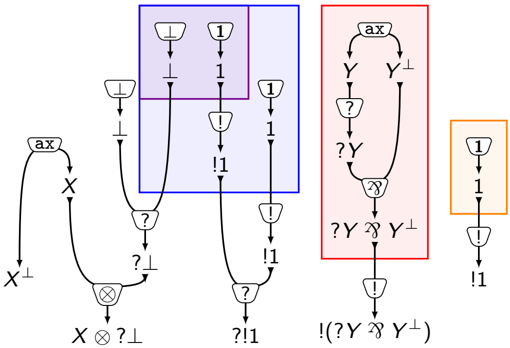
(2)

Theorem: The category of directed graphs has all pullbacks!

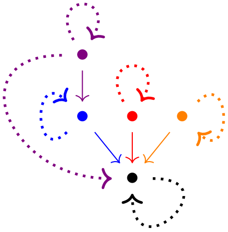
Idea: The pullback P of two graph morphisms $f\colon X \to Z$ and $g\colon Y \to Z$ is the "maximal" directed graph compatible with X and Y.

Pullback in DiLL proof-nets

An MELL proof-structure $R = (|R|, \mathcal{A}, box)$: *box* is given by colors.



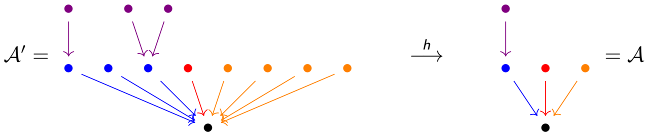
(a) The underlying graph $|R|$ (the information about boxes is extracted from *box*).



(b) The box-tree $\mathcal{A}$ (without dotted lines) and the reflexive-transitive closure $\mathcal{A}^{\circlearrowleft}$ of $\mathcal{A}$ (with also dotted lines).

Pullback in DiLL proof-nets

A thick subtree $(\mathcal{A}', h)$ of $\mathcal{A}$:



Remark: The graph homomorphism $h\colon \mathcal{A}' \to \mathcal{A}$ lifts to the trans.-refl. closure:

$$h^{\circlearrowleft}\colon {\mathcal{A}'}^{\circlearrowleft} \to \mathcal{A}^{\circlearrowleft}$$

Pullback in DiLL proof-nets

We have two graph homomorphisms $h^{\circlearrowleft} \colon {\mathcal{A}'}^{\circlearrowleft} \to \mathcal{A}^{\circlearrowleft}$ and $box \colon |R| \to \mathcal{A}^{\circlearrowleft}$.

Question: What is the pullback $|R'|$ of $h^{\circlearrowleft}$ and box?

$$\begin{array}{ccc} & & {\mathcal{A}'}^{\circlearrowleft} \\ & & \big\downarrow h^{\circlearrowleft} \\ |R| & \xrightarrow{box} & \mathcal{A}^{\circlearrowleft} \end{array}$$

$|R'| =$

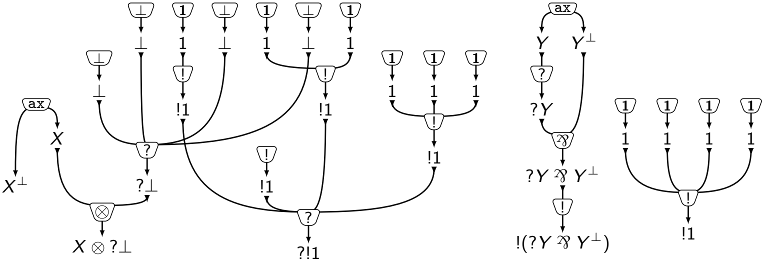
A new DiLL proof-net $R' = (|R'|, \mathcal{A}', box')$ generated by the thick subtree $(\mathcal{A}', h)$.

Remark: This is possible because all objects live in the category of directed graphs

Pullback in DiLL proof-nets

We have two graph homomorphisms $h^{\circlearrowleft} : \mathcal{A}'^{\circlearrowleft} \to \mathcal{A}^{\circlearrowleft}$ and $box : |R| \to \mathcal{A}^{\circlearrowleft}$.
Question: What is the pullback $|R'|$ of $h^{\circlearrowleft}$ and box?

$$\begin{array}{ccc} |R'| & \xrightarrow{box'} & \mathcal{A}'^{\circlearrowleft} \\ \Big\downarrow \pi & & \Big\downarrow h^{\circlearrowleft} \\ |R| & \xrightarrow{box} & \mathcal{A}^{\circlearrowleft} \end{array}$$

$|R'| =$ 

A new DiLL proof-net $R' = (|R'|, \mathcal{A}', box')$ generated by the thick subtree $(\mathcal{A}', h)$.

Remark: This is possible because all objects live in the category of directed graphs

What for?

We can define in an "easy" but rigorous way operations on proof-nets that are:

- intuitively clear,
- tricky to define precisely.

Examples: cut-elimination, Taylor expansion, identity of proof-nets, correctness . . .

For correctness criterion, identity of proof-nets and the Taylor expansion in MELL:

G. G., L. Pellissier, L. Tortora de Falco. *Proof-nets as graph, Taylor expansion as pullback*. WoLLIC, 2020.

What for?

We can define in an “easy” but rigorous way operations on proof-nets that are:

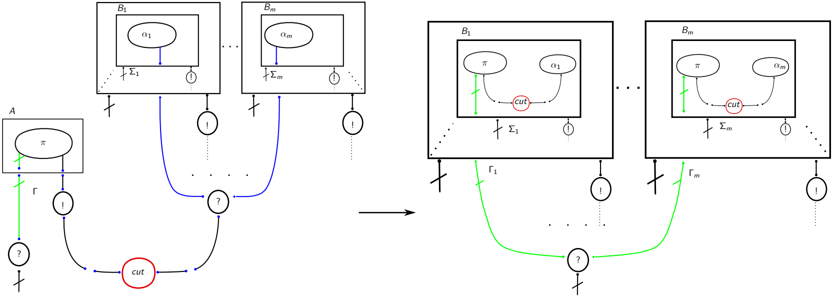
- intuitively clear,
- tricky to define precisely.

Examples: cut-elimination, Taylor expansion, identity of proof-nets, correctness . . .

For correctness criterion, identity of proof-nets and the Taylor expansion in MELL:

G. G., L. Pellissier, L. Tortora de Falco. *Proof-nets as graph, Taylor expansion as pullback*. WoLLIC, 2020.

Example: the exponential step for cut-elimination

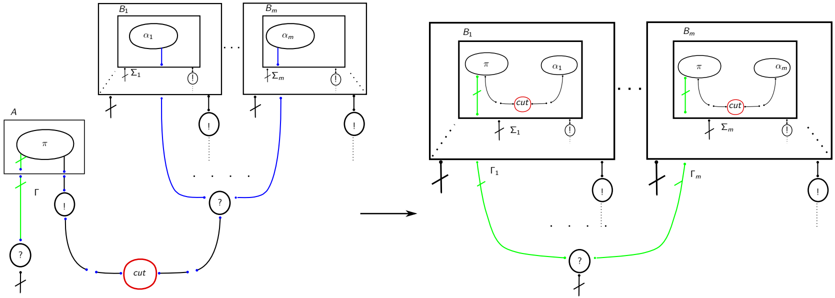


It is tricky (and usually boring) to define this step in a rigorous way!

Sketch of the definition in our framework $\rightsquigarrow$ get rid of most tedious aspects.

1. The exponential cut says which thick subtree to consider (how many copies?).
2. Pullback from that thick subtree, in which we duplicate the box.
3. Local rewriting on the underlying graph (erase !/? nodes, add new cut nodes).
4. Locally updating the box-tree and the graph homomorphism *box*.

Example: the exponential step for cut-elimination

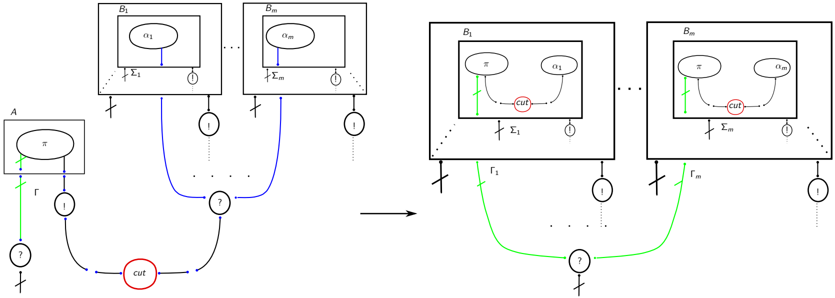


It is tricky (and usually boring) to define this step in a rigorous way!

Sketch of the definition in our framework ⇝ get rid of most tedious aspects.

1. The exponential cut says which thick subtree to consider (how many copes?).
2. Pullback from that thick subtree, in which we duplicate the box.
3. Local rewriting on the underlying graph (erase !/? nodes, add new cut nodes).
4. Locally updating the box-tree and the graph homomorphism *box*.

Example: the exponential step for cut-elimination

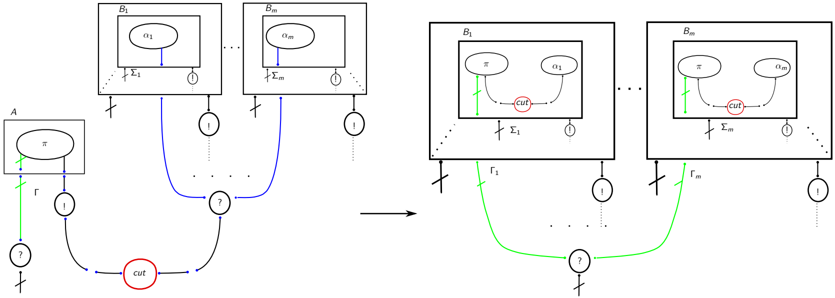


It is tricky (and usually boring) to define this step in a rigorous way!

Sketch of the definition in our framework ⇝ get rid of most tedious aspects.

1. The exponential cut says which thick subtree to consider (how many copes?).
2. Pullback from that thick subtree, in which we duplicate the box.
3. Local rewriting on the underlying graph (erase !/? nodes, add new cut nodes).
4. Locally updating the box-tree and the graph homomorphism *box*.

Example: the exponential step for cut-elimination

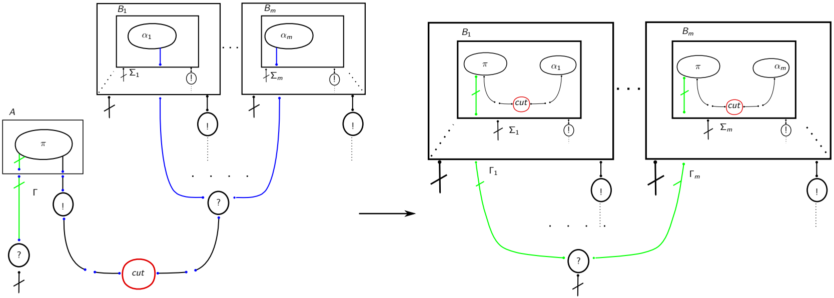


It is tricky (and usually boring) to define this step in a rigorous way!

Sketch of the definition in our framework $\rightsquigarrow$ get rid of most tedious aspects.

1. The exponential cut says which thick subtree to consider (how many copes?).
2. Pullback from that thick subtree, in which we duplicate the box.
3. Local rewriting on the underlying graph (erase !/? nodes, add new cut nodes).
4. Locally updating the box-tree and the graph homomorphism *box*.

Example: the exponential step for cut-elimination



It is tricky (and usually boring) to define this step in a rigorous way!

Sketch of the definition in our framework ⇝ get rid of most tedious aspects.

1. The exponential cut says which thick subtree to consider (how many copes?).
2. Pullback from that thick subtree, in which we duplicate the box.
3. Local rewriting on the underlying graph (erase !/? nodes, add new cut nodes).
4. Locally updating the box-tree and the graph homomorphism *box*.

What's next?

1. Parallel cut-elimination for MELL proof-nets:
   - confluence;
   - factorization/standardization;
   - strong normalization.
2. Commutation of Taylor expansion and cut-elimination.
3. . . .

Many of these results are already established with different methods.
Our goal: To provide a unified and robust approach.

THANKS!

What's next?

1. Parallel cut-elimination for MELL proof-nets:
   - confluence;
   - factorization/standardization;
   - strong normalization.
2. Commutation of Taylor expansion and cut-elimination.
3. ...

Many of these results are already established with different methods.
Our goal: To provide a unified and robust approach.

THANKS!

What's next?

1. Parallel cut-elimination for MELL proof-nets:
   - confluence;
   - factorization/standardization;
   - strong normalization.
2. Commutation of Taylor expansion and cut-elimination.
3. ...

Many of these results are already established with different methods.
Our goal: To provide a unified and robust approach.

THANKS!

What's next?

1. Parallel cut-elimination for MELL proof-nets:
   - confluence;
   - factorization/standardization;
   - strong normalization.
2. Commutation of Taylor expansion and cut-elimination.
3. ...

Many of these results are already established with different methods.
Our goal: To provide a unified and robust approach.

THANKS!