Connectivity and correctness in linear logic proofs

Raffaele Di Donna [1, 2] Giulio Guerrieri [3] Lorenzo Tortora de Falco[2]

[1]Université Paris Cité

[2]Università Roma Tre

[3]University of Sussex

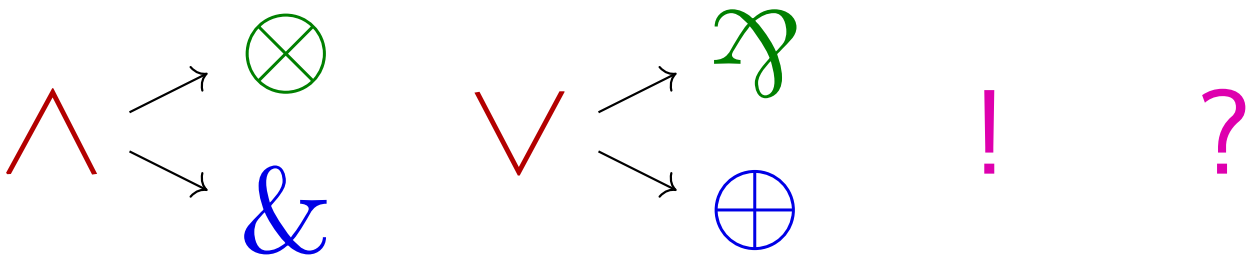
International Research Network: Logic and Interaction (IRN LI)
General meeting

ENS de Lyon
23 June 2026

Table of contents

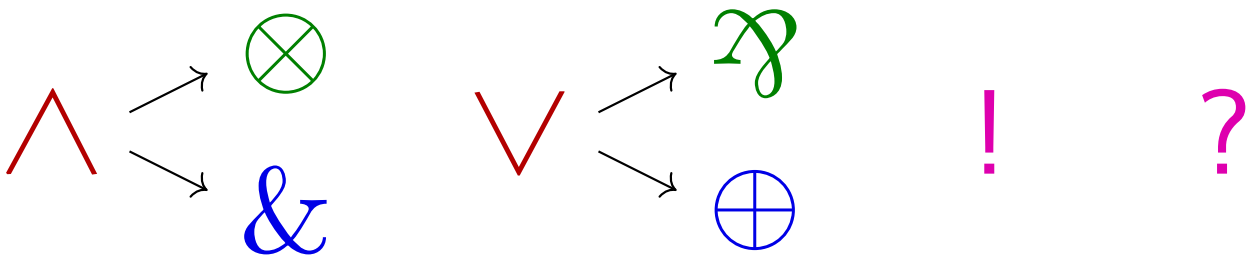
Introduction

L I N E A R L O G I C



(Girard, 1987)

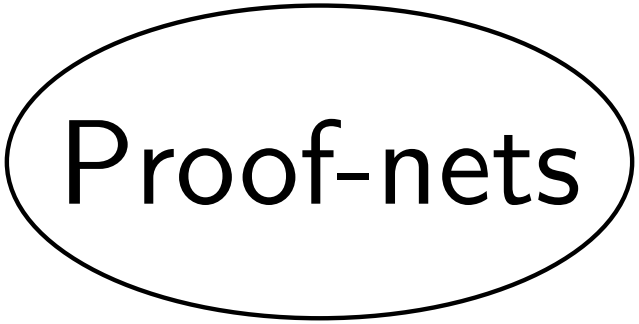
L I N E A R L O G I C

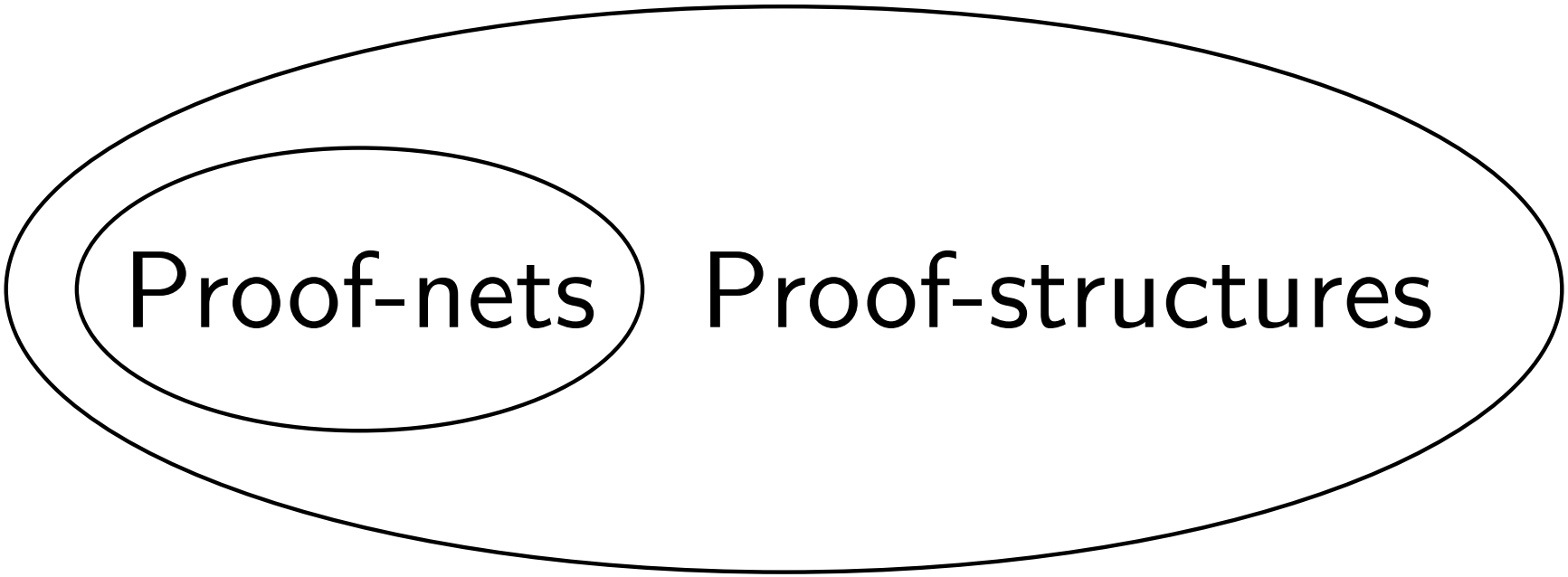


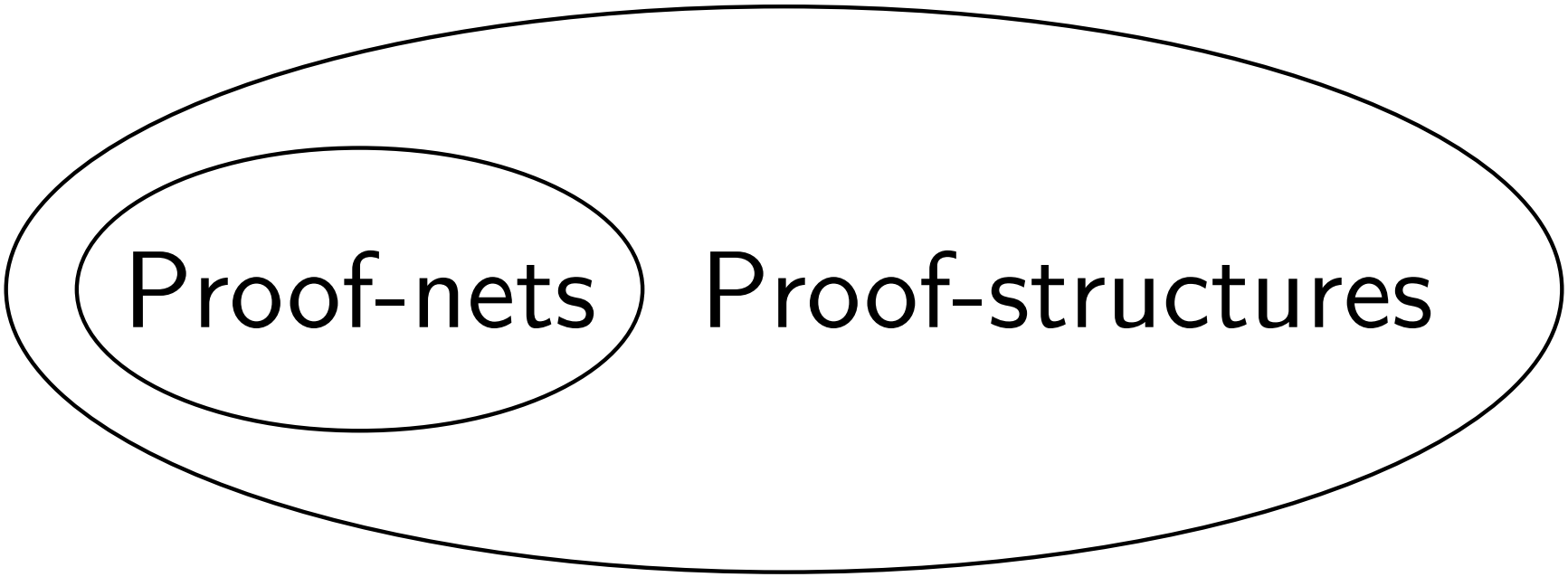
(Girard, 1987)

Representation of proofs

Terms / Trees $\rightsquigarrow$ **Graphs**


Proof-nets


Proof-nets
Proof-structures

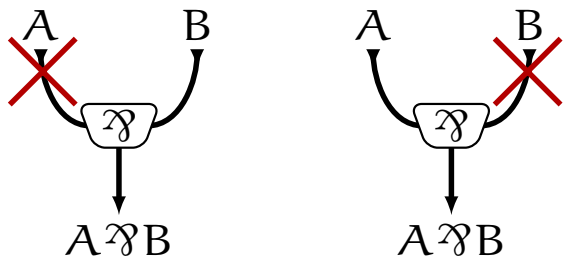


Correctness criteria

MLL^- $\leftrightsquigarrow$ ACC

$A ::= X \mid A \otimes A \mid A \parr A$

ACyclicity + Connectivity

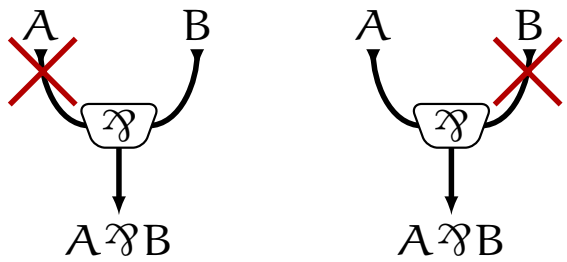


(Danos and Regnier, 1989)

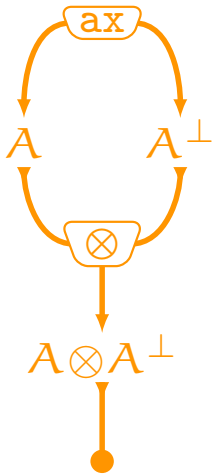
MLL^- $\leftrightsquigarrow$ ACC

$A ::= X \mid A \otimes A \mid A \parr A$

ACyclicity + Connectivity



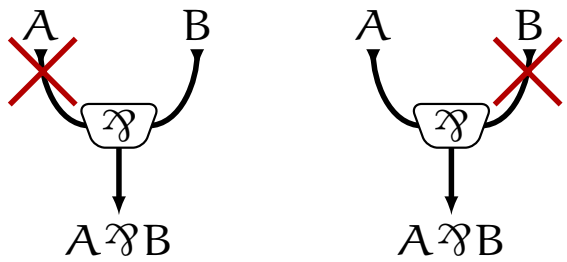
(Danos and Regnier, 1989)



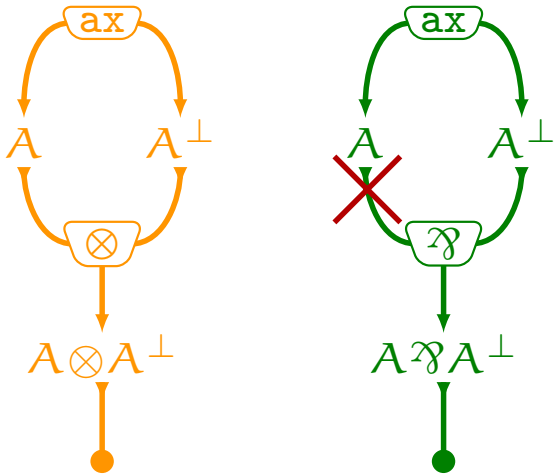
MLL^- $\leftrightsquigarrow$ ACC

$A ::= X \mid A \otimes A \mid A \parr A$

ACyclicity + Connectivity



(Danos and Regnier, 1989)

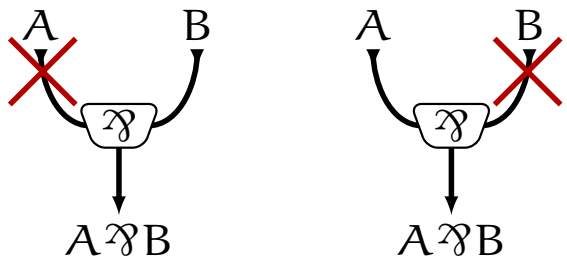


$$\dfrac{\dfrac{}{\vdash A, A^\perp}\, ax}{\vdash A \parr A^\perp}\, \parr$$

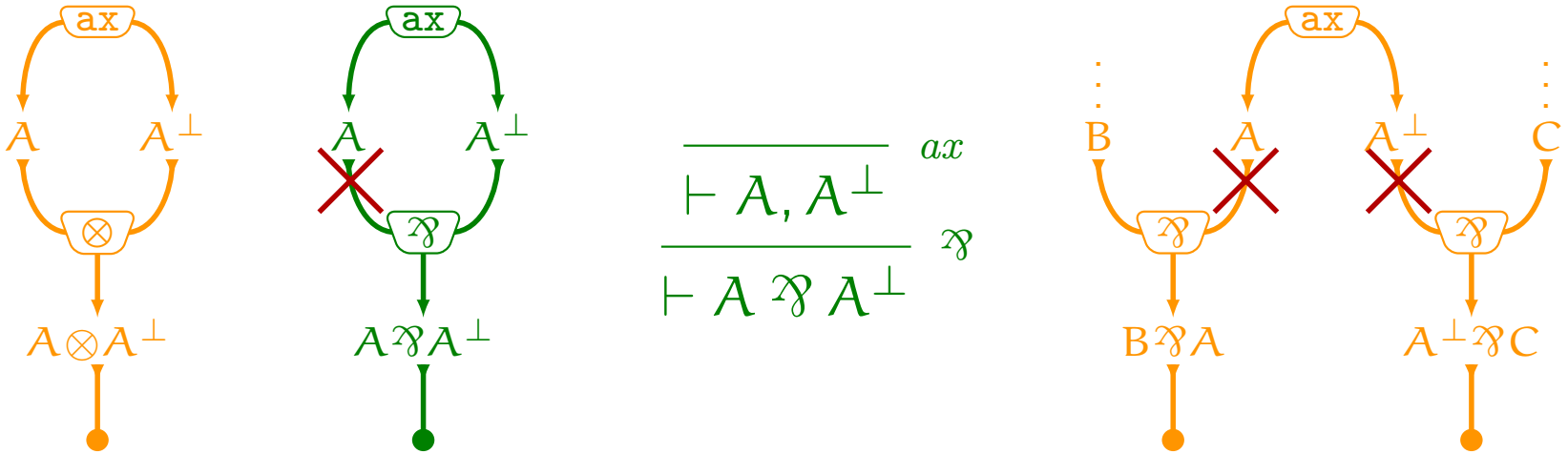
MLL^- $\leftrightsquigarrow$ ACC

$A ::= X \mid A \otimes A \mid A \parr A$

ACyclicity + Connectivity



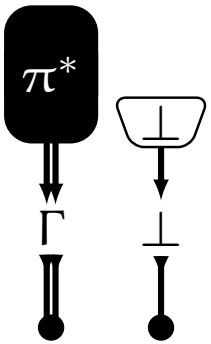
(Danos and Regnier, 1989)



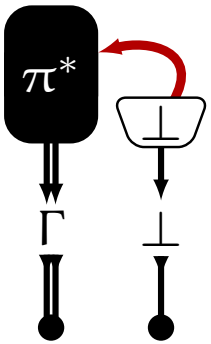
MLL

$$A ::= X \mid \mathbf{1} \mid \bot \mid A \otimes A \mid A \parr A$$

$$(\cdot)^* : \quad \dfrac{\begin{array}{c}\vdots\ \pi \\ \vdash \Gamma\end{array}}{\vdash \Gamma, \bot}\ \bot \quad \longmapsto$$



$$(\cdot)^* : \quad \dfrac{\begin{array}{c}\vdots\ \pi \\ \vdash \Gamma\end{array}}{\vdash \Gamma, \bot}\ \bot \quad \longmapsto$$



MLL $\leftrightsquigarrow$ ACC + jumps

$A ::= X \mid \mathbf{1} \mid \bot \mid A \otimes A \mid A \parr A$ ACyclicity + Connectivity

(Girard, common knowledge...)

$$\text{MLL} + mix \qquad \leftrightsquigarrow \qquad \text{AC}$$

$$A ::= X \mid \mathbf{1} \mid \bot \mid A \otimes A \mid A \parr A \qquad\qquad \text{ACyclicity}$$

(Fleury and Retoré, 1994)

MLL $\leftrightsquigarrow$ AC...

$A ::= X \mid \mathbf{1} \mid \bot \mid A \otimes A \mid A \parr A$

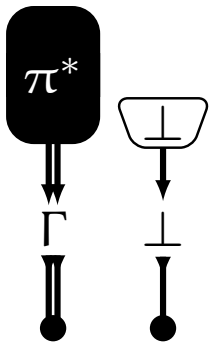
ACyclicity + ?????

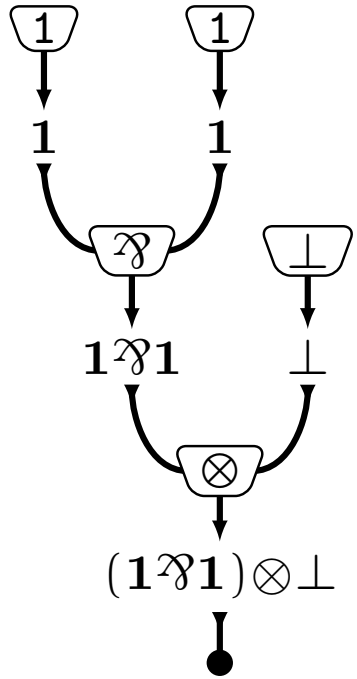
$\mathrm{ACC}_{\sharp w}$

ACyclicity and $\sharp cc = \sharp w + 1$

Where $\sharp cc$ is the number of connected components,
$\sharp w$ is the number of $\bot$ nodes.

$$(\cdot)^* : \quad \dfrac{\begin{array}{c}\vdots\ \pi \\ \vdash \Gamma\end{array}}{\vdash \Gamma, \bot}\ \bot \quad \longmapsto$$





An incorrect proof-structure of MLL satisfying $\mathrm{ACC}_{\sharp\mathrm{w}}$

(Regnier, 1992)

In COMLL (constant-only MLL)
provability is NP-complete

(Lincoln and Winkler, 1994)

$$\Downarrow$$

No feasible correctness criterion for MLL

Reverse the point of view!

What is the logical meaning of connectivity?

Reverse the point of view!

What is the logical meaning of connectivity?

1. How is connectivity related to the correctness criteria that are specific to some fragments of linear logic?

Reverse the point of view!

What is the logical meaning of connectivity?

1. How is connectivity related to the correctness criteria that are specific to some fragments of linear logic?
2. Are there interesting fragments for which $\mathrm{ACC}_{\sharp w}$ is a correctness criterion?

Proof-nets

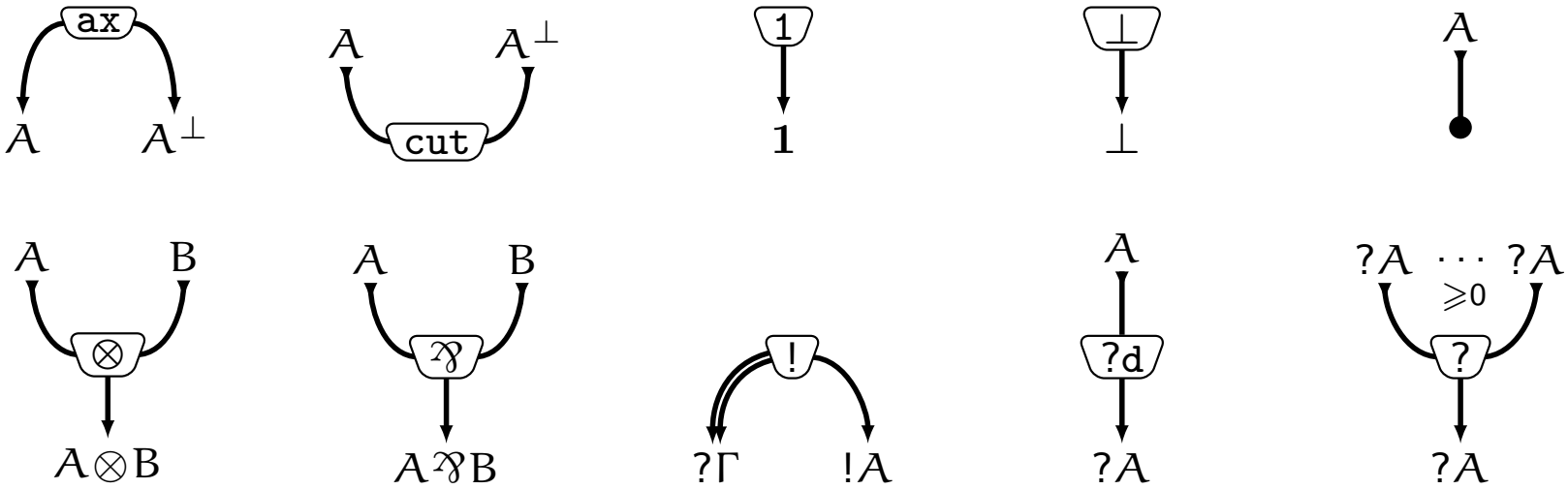
Multiplicatives and exponentials

Formulae of MELL:

$$A ::= X \mid \mathbf{1} \mid \bot \mid A \otimes A \mid A \parr A \mid !A \mid ?A$$

Proof-structures of MELL

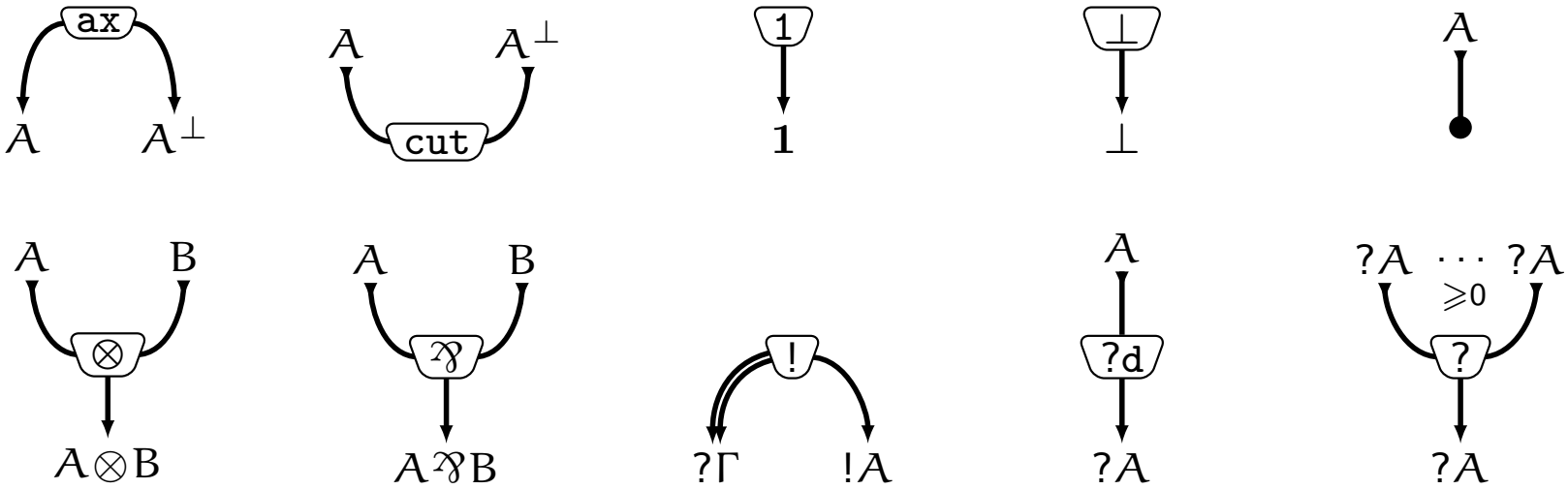
Graph made up of the following building blocks:



And equipped with a function mapping every ! node to a proof-structure of MELL with the same conclusions (box).

Proof-structures of MELL

Graph made up of the following building blocks:



And equipped with a function mapping every ! node to a proof-structure of MELL with the same conclusions (box).

Cut elimination $\approx$ graph rewriting.

From sequent calculus to proof-structures

π sequent calculus proof $\quad \mapsto \quad$ π^* proof-structure

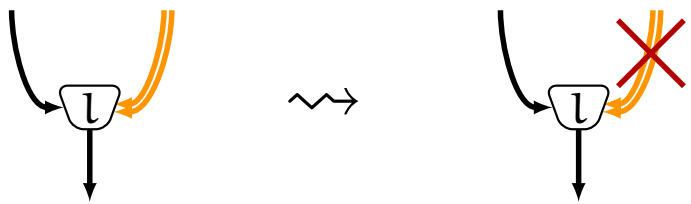
From sequent calculus to proof-structures

π sequent calculus proof $\quad \mapsto \quad$ π^* proof-structure

A proof-structure R is correct if there exists
a sequent calculus proof π such that $\mathrm{R} = \pi^*$.

Switchings

For every $\parr$ or ? node with at least two premises:



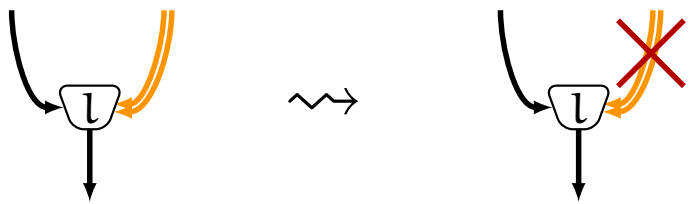
(Danos and Regnier, 1989)

AC = every switching graph is acyclic.

(and every box satisfies AC*)*

Switchings

For every $\parr$ or ? node with at least two premises:



(Danos and Regnier, 1989)

AC = every switching graph is acyclic.

(and every box satisfies AC*)*

(Danos and Regnier, 1989)

ACC = AC and every switching graph is connected.

(and every box satisfies ACC*)*

Extending ACC in the presence of the units

(Regnier, 1992)

$\mathrm{ACC}_{\sharp w} = \mathrm{AC}$ and $\sharp cc = \sharp w + 1$ in every switching graph.

(and every box satisfies $\mathrm{ACC}_{\sharp w}$*)*

Where $\sharp cc$ is the number of connected components,
$\sharp w$ is the number of nodes $\bot$ or weakening.

Almost a correctness criterion

Theorem. Let R, R' be proof-structures with $R \to R'$.
If R satisfies $ACC_{\sharp w}$, then R' satisfies $ACC_{\sharp w}$.

(Common knowledge – Di Donna and Tortora de Falco, preprint)

Almost a correctness criterion

Theorem. Let R, R' be proof-structures with $R \to R'$.
If R satisfies $ACC_{\sharp w}$, then R' satisfies $ACC_{\sharp w}$.

(Common knowledge – Di Donna and Tortora de Falco, preprint)

Proposition. Let R be a proof-structure of MELL.
If R is correct, then R satisfies $ACC_{\sharp w}$.

(Obvious)

Almost a correctness criterion

Theorem. Let R, R' be proof-structures with $R \to R'$.
If R satisfies $ACC_{\sharp w}$, then R' satisfies $ACC_{\sharp w}$.

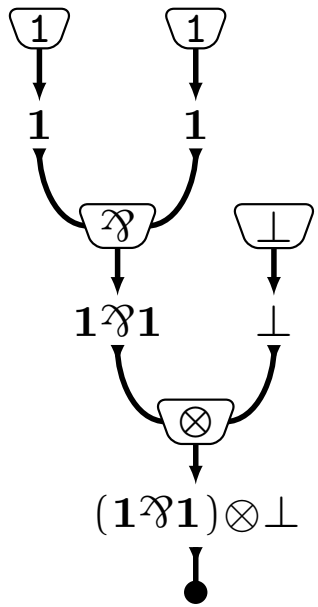
(Common knowledge – Di Donna and Tortora de Falco, preprint)

Proposition. Let R be a proof-structure of MELL.
If R is correct, then R satisfies $ACC_{\sharp w}$.

(Obvious)

But the converse is false in general:

(Regnier, 1992)



A generalized polarization

Reminder: the classical polarization

Fragment LL_{pol} of MELL:

(Girard, 1991 – Laurent, 1999)

$$\mathrm{P} ::= \mathrm{X} \mid \mathbf{1} \mid \mathrm{P} \otimes \mathrm{P} \mid !\mathrm{N} \qquad (\textit{positives})$$

$$\mathrm{N} ::= \mathrm{X}^{\perp} \mid \perp \mid \mathrm{N} \parr \mathrm{N} \mid ?\mathrm{P} \qquad (\textit{negatives})$$

- ☑ Has a simple correctness criterion;
- ☑ Contains the call-by-name λ-calculus;
- ☐ Contains the call-by-value λ-calculus.

Reminder: the intuitionistic polarization

Fragment IMELL of MELL:

(Danos, 1990 – Regnier, 1992 – Lamarche, 2008)

$$\mathrm{O} ::= X \mid \mathbf{1} \mid \mathrm{O} \otimes \mathrm{O} \mid \mathrm{O} \parr \mathrm{I} \mid \mathrm{I} \parr \mathrm{O} \mid !\mathrm{O} \qquad (\textit{outputs})$$

$$\mathrm{I} ::= X^{\perp} \mid \perp \mid \mathrm{I} \parr \mathrm{I} \mid \mathrm{I} \otimes \mathrm{O} \mid \mathrm{O} \otimes \mathrm{I} \mid ?\mathrm{I} \qquad (\textit{inputs})$$

- ☐ Has a simple correctness criterion;
- ☑ Contains the call-by-name λ-calculus;
- ☑ Contains the call-by-value λ-calculus.

A new unifying setting

Fragment PMELL of MELL:

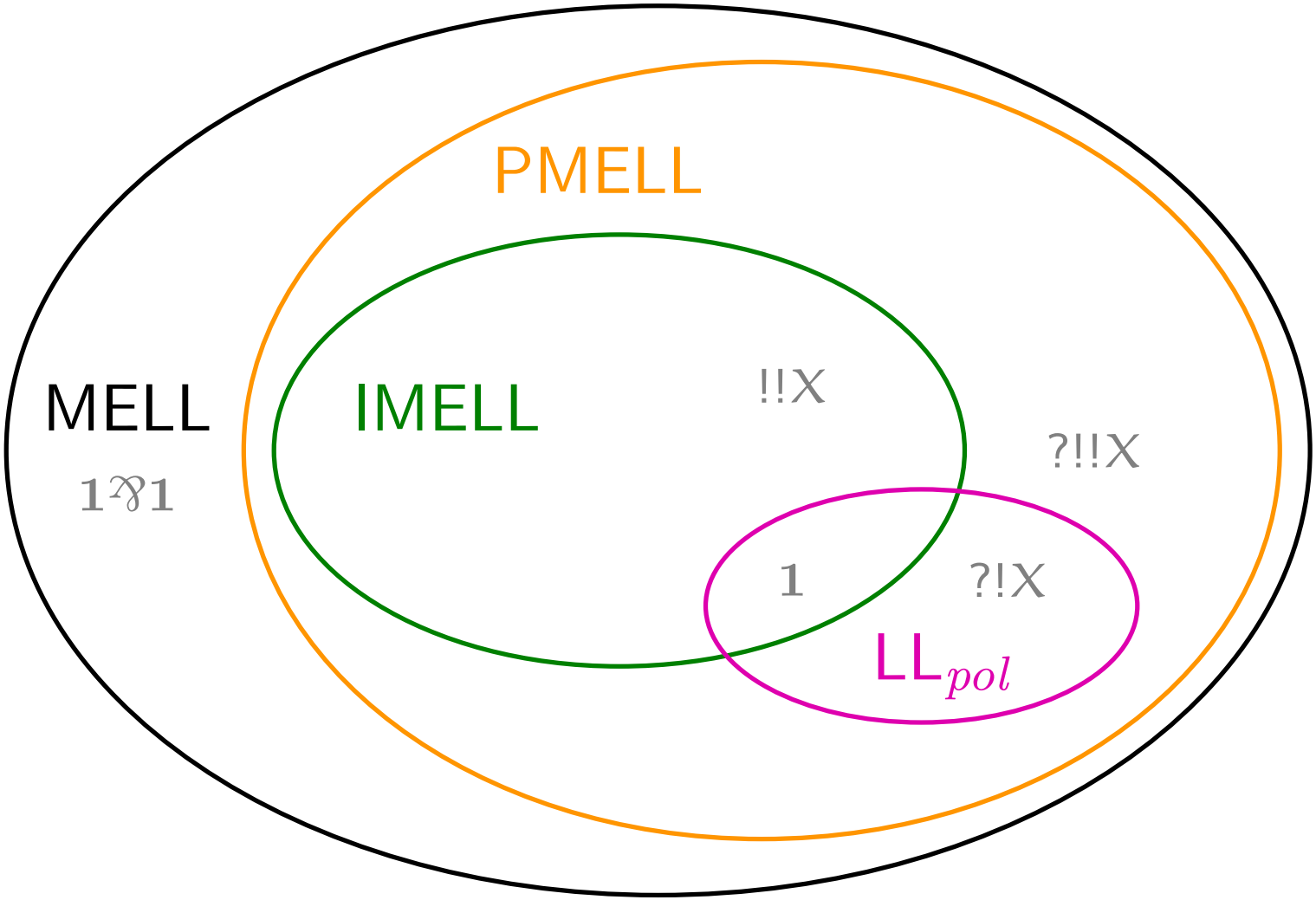
(Di Donna, Guerrieri and Tortora de Falco, technical report)

$$O ::= X \mid 1 \mid O \otimes O \mid O \parr I \mid I \parr O \mid !O \mid !I \quad (\textit{outputs})$$

$$I ::= X^{\perp} \mid \perp \mid I \parr I \mid I \otimes O \mid O \otimes I \mid ?I \mid ?O \quad (\textit{inputs})$$

Remark. Both LL_{pol} and IMELL are fragments of PMELL.

Venn diagram of the fragments



$ACC_{\sharp w}$ in PMELL and in IMELL

Let R be a proof-structure of PMELL. If R satisfies AC:

$\sharp cc = \sharp w + 1$ (i.e. $ACC_{\sharp w}$) if and only if $\sharp d_c + \sharp o = 1$

Where $\sharp d_c$ is the number of ?d nodes with output premise;
$\sharp o$ is the number of output conclusions.

(Di Donna, Guerrieri and Tortora de Falco, technical report)

$\mathrm{ACC}_{\sharp w}$ in PMELL and in IMELL

Let R be a proof-structure of PMELL. If R satisfies AC:

$\sharp cc = \sharp w + 1$ (i.e. $\mathrm{ACC}_{\sharp w}$) if and only if $\sharp d_c + \sharp o = 1$

Where $\sharp d_c$ is the number of ?d nodes with output premise;
$\sharp o$ is the number of output conclusions.

(Di Donna, Guerrieri and Tortora de Falco, technical report)

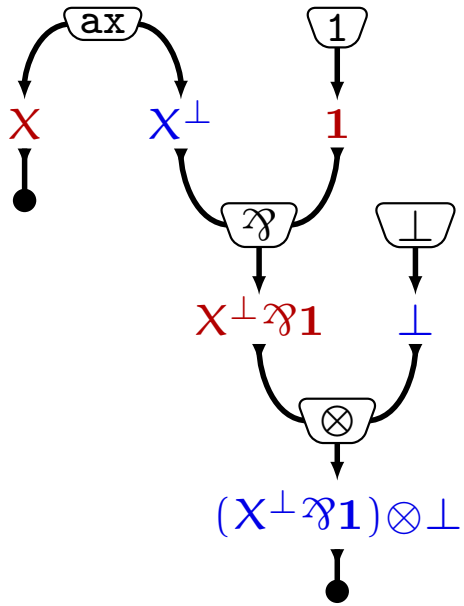
In IMELL: the premise of every ?d node is input
$\rightsquigarrow$ $\mathrm{ACC}_{\sharp w}$ equivalent to AC plus $\sharp o = 1$

$\mathrm{ACC}_{\sharp w}$ alone does not characterize correctness in IMELL

Every correct proof-structure of IMELL satisfies $\mathrm{ACC}_{\sharp w}$.

(This holds more generally for proof-structures of MELL.)

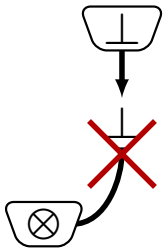
The converse is false: here is a proof-structure of IMELL (and thus of PMELL) satisfying $\mathrm{ACC}_{\sharp w}$ but not correct:



(Regnier's counterexample adapted to IMELL)

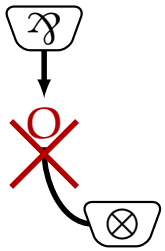
Two geometric restrictions

In untyped MELL:



(Di Donna and Tortora de Falco, preprint)

In PMELL:



(Di Donna, Guerrieri and Tortora de Falco, technical report)

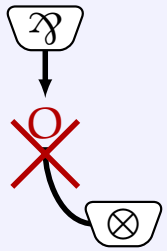
Theorem

Under any one of these restrictions, $\mathrm{ACC}_{\sharp w}$ becomes a correctness criterion.

Another characterization of $\text{ACC}_{\sharp w}$

Theorem

In PMELL with this restriction:



$\text{ACC}_{\sharp w}$ is equivalent to the directed acyclicity of *one* graph, plus the requirement $\sharp d_c + \sharp o = 1$

(Di Donna, Guerrieri and Tortora de Falco, technical report)

$\mathsf{ACC}_{\sharp\mathsf{w}}$ in LL_{pol}

Every proof-structure of LL_{pol} satisfies the geometric restriction. Thus:

Corollary

In LL_{pol}, the property $\mathsf{ACC}_{\sharp\mathsf{w}}$ is:

- A correctness criterion;
- Equivalent to the criterion in (Laurent, 1999).

(Di Donna, Guerrieri and Tortora de Falco, technical report)

The (new) value fragment

Fragment VMELL of PMELL:

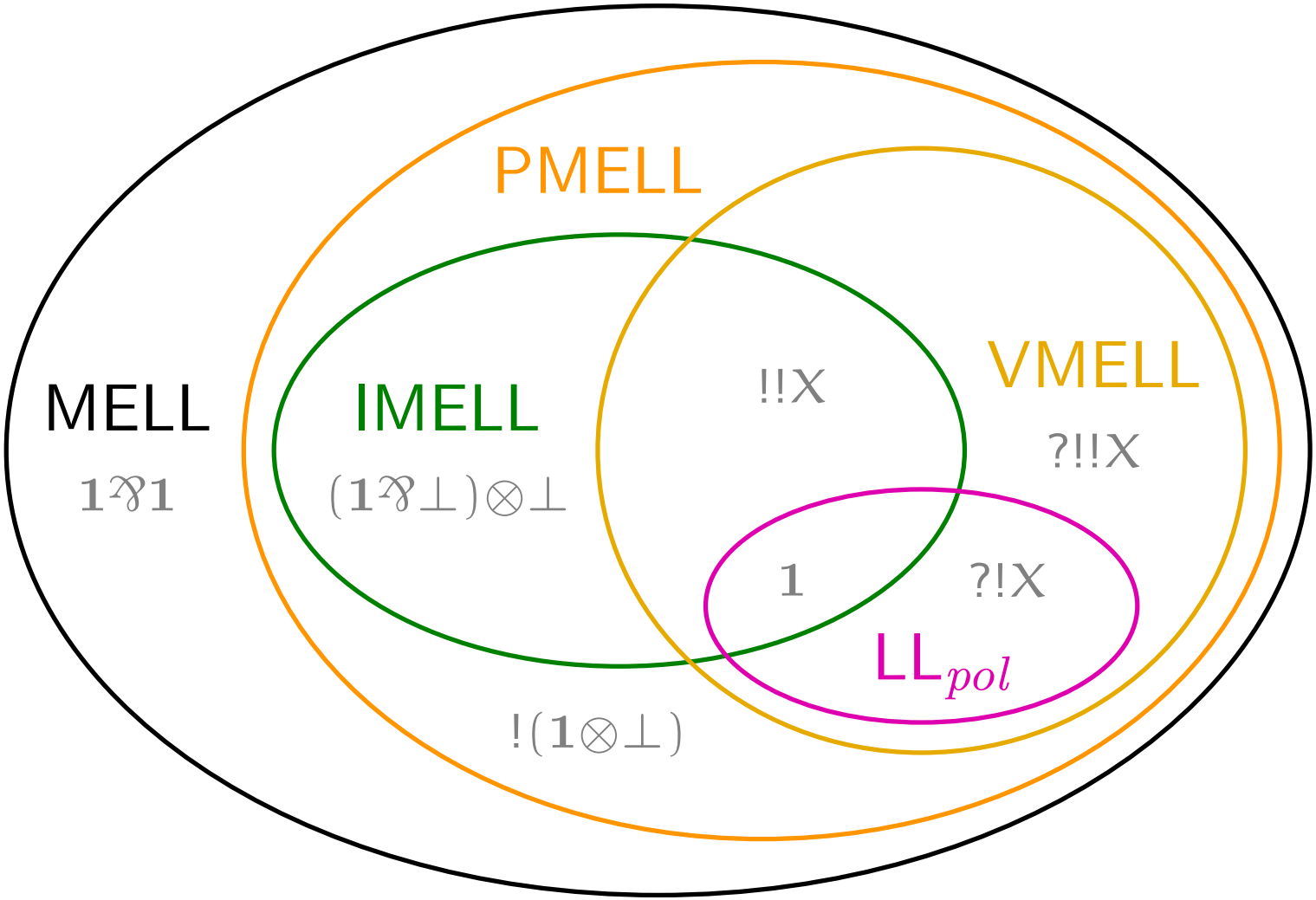
(Di Donna, Guerrieri and Tortora de Falco, technical report)

$$\begin{aligned}
P &::= X \mid 1 \mid P \otimes P \mid !O \mid !N && \text{(\textit{positives})}\\
O &::= P \mid O \parr N \mid N \parr O && \text{(\textit{outputs})}\\
N &::= X^{\perp} \mid \perp \mid N \parr N \mid ?I \mid ?P && \text{(\textit{negatives})}\\
I &::= N \mid I \otimes P \mid P \otimes I && \text{(\textit{inputs})}
\end{aligned}$$

Remark. VMELL contains LL_{pol} and the counterpart in MELL of the half-polarized typing system for CBPV.

(Ehrhard, 2016 – Ehrhard and Tasson, 2019)

Updated Venn diagram



Correctness criterion

Every multiplicative normal proof-structure of VMELL satisfies the geometric restriction. Thus:

Corollary

For the multiplicative normal proof-structures of VMELL, the property $\mathrm{ACC}_{\sharp w}$ is:

- A correctness criterion;
- Equivalent to the directed acyclicity of *one* graph, plus the requirement $\sharp d_c + \sharp o = 1$.

(Di Donna, Guerrieri and Tortora de Falco, technical report)

Comparison table

	Expressivity					$\mathsf{ACC}_{\sharp w}$ correctness criterion
	λ-calculus		Bang calculus	LL_{pol}	IMELL	
	CBN	CBV				
LL_{pol}	☑	☒	☒	☑	☒	☑
IMELL	☑	☑	☑	☒	☑	☒
PMELL	☑	☑	☑	☑	☑	☒
VMELL	☑	☑	☑	☑	☑ (emb.)	☑ (m. norm.)

Connectivity and the bang calculus

The bang calculus (Ehrhard and Guerrieri, 2016)

Syntax:

$$t ::= x \mid \lambda x\, t \mid \langle t \rangle t \mid \mathsf{der}\, t \mid t^{!}$$

The bang calculus (Ehrhard and Guerrieri, 2016)

Syntax:

$$t ::= x \mid \lambda x\, t \mid \langle t \rangle t \mid \mathsf{der}\, t \mid t^{!}$$

Dynamics:

$$\langle \lambda x t \rangle s^{!} \to_{\ell} t\{s/x\}$$

$$\mathsf{der}\, t^{!} \to_{\mathsf{d}} t$$

We define $\to_{\mathsf{b}} := \to_{\ell} \cup \to_{\mathsf{d}}$.

The bang fragment

Fragment !MELL of VMELL and IMELL:

(Di Donna, Guerrieri and Tortora de Falco, technical report)

$$O ::= X \mid ?I \parr O \mid !O \qquad (\textit{outputs})$$

$$I ::= X^{\perp} \mid !O \otimes I \mid ?I \qquad (\textit{inputs})$$

The bang fragment

Fragment !MELL of VMELL and IMELL:

(Di Donna, Guerrieri and Tortora de Falco, technical report)

$$O ::= X \mid ?I \parr O \mid !O \qquad (\textit{outputs})$$

$$I ::= X^{\perp} \mid !O \otimes I \mid ?I \qquad (\textit{inputs})$$

We define $!O \multimap O' := ?O^{\perp} \parr O'$.

Church-style typing

Abstractions are annotated: $\lambda x^{!O} t$

We define a *partial* typing function t_b mapping (Γ, t), with $\Gamma = x_1 : !O_1, \ldots, x_k : !O_k$, to an output formula:

- $t_b(\Gamma, x_i) \coloneqq O_i$;
- $t_b(\Gamma, \lambda x^{!O} t) \coloneqq !O \multimap t_b((\Gamma, x : !O), t)$;
- $t_b(\Gamma, \langle t \rangle s) \coloneqq O$ if $t_b(\Gamma, t) = t_b(\Gamma, s) \multimap O$;
- $t_b(\Gamma, \mathrm{der}\, t) \coloneqq O$ if $t_b(\Gamma, t) = !O$;
- $t_b(\Gamma, t^!) \coloneqq !t_b(\Gamma, t)$.

A term t is typed in the context Γ if $(\Gamma, t) \in \mathrm{dom}(t_b)$.

Girard's translations

Translations $(\cdot)^{\mathtt{N}}$ and $(\cdot)^{\mathtt{V}}$ of simple types into !MELL:

$$X^{\mathtt{N}} := X \qquad (T \Rightarrow S)^{\mathtt{N}} := !T^{\mathtt{N}} \multimap S^{\mathtt{N}}$$

$$X^{\mathtt{V}} := X \qquad (T \Rightarrow S)^{\mathtt{V}} := !T^{\mathtt{V}} \multimap !S^{\mathtt{V}}$$

(Girard, 1987)

Girard's translations

Translations $(\cdot)^{\mathtt{N}}$ and $(\cdot)^{\mathtt{V}}$ of simple types into !MELL:

$$X^{\mathtt{N}} \coloneqq X \qquad (T \Rightarrow S)^{\mathtt{N}} \coloneqq {!}T^{\mathtt{N}} \multimap S^{\mathtt{N}}$$
$$X^{\mathtt{V}} \coloneqq X \qquad (T \Rightarrow S)^{\mathtt{V}} \coloneqq {!}T^{\mathtt{V}} \multimap {!}S^{\mathtt{V}}$$

(Girard, 1987)

Translations $(\cdot)^{\mathtt{n}}$ and $(\cdot)^{\mathtt{v}}$ of λ-terms into the bang calculus:

$$x^{\mathtt{n}} \coloneqq x \qquad (\lambda x\, M)^{\mathtt{n}} \coloneqq \lambda x\, M^{\mathtt{n}} \qquad ((M)N)^{\mathtt{n}} \coloneqq \langle M^{\mathtt{n}} \rangle (N^{\mathtt{n}})^{!}$$
$$x^{\mathtt{v}} \coloneqq x^{!} \qquad (\lambda x\, M)^{\mathtt{v}} \coloneqq (\lambda x\, M^{\mathtt{v}})^{!} \qquad ((M)N)^{\mathtt{v}} \coloneqq \langle \mathsf{der}\, M^{\mathtt{v}} \rangle N^{\mathtt{v}}$$

(Guerrieri and Manzonetto, 2018)

Girard's translations

Translations $(\cdot)^{N}$ and $(\cdot)^{V}$ of simple types into !MELL:

$$X^{N} := X \qquad (T \Rightarrow S)^{N} := {!}T^{N} \multimap S^{N}$$
$$X^{V} := X \qquad (T \Rightarrow S)^{V} := {!}T^{V} \multimap {!}S^{V}$$

(Girard, 1987)

Translations $(\cdot)^{n}$ and $(\cdot)^{v}$ of λ-terms into the bang calculus:

$$x^{n} := x \qquad (\lambda x\, M)^{n} := \lambda x\, M^{n} \qquad ((M)N)^{n} := \langle M^{n} \rangle (N^{n})^{!}$$
$$x^{v} := x^{!} \qquad (\lambda x\, M)^{v} := (\lambda x\, M^{v})^{!} \qquad ((M)N)^{v} := \langle \mathsf{der}\, M^{v} \rangle N^{v}$$

(Guerrieri and Manzonetto, 2018)

If t_{λ} denotes the Church-style typing function for the λ-calculus:

$$\mathsf{t}_{b}({!}\Gamma^{N}, M^{n}) = (\mathsf{t}_{\lambda}(\Gamma, M))^{N} \qquad \mathsf{t}_{b}({!}\Gamma^{V}, M^{v}) = {!}(\mathsf{t}_{\lambda}(\Gamma, M))^{V}$$

(Di Donna, Guerrieri and Tortora de Falco, technical report)

Back to proof-nets

$$\text{Proof-net} = \text{proof-structure of !MELL} + \text{ACC}_{\sharp w}.$$

Back to proof-nets

Proof-net $=$ proof-structure of !MELL $+$ $\mathsf{ACC}_{\sharp\mathsf{w}}$.

Bang proof-net $=$ proof-net $+$ every input premise of a $\parr/?/\bullet$ node is a conclusion of a $!/?\mathrm{d}/?$ node $+$ input conclusions labeled by distinct variables.

Back to proof-nets

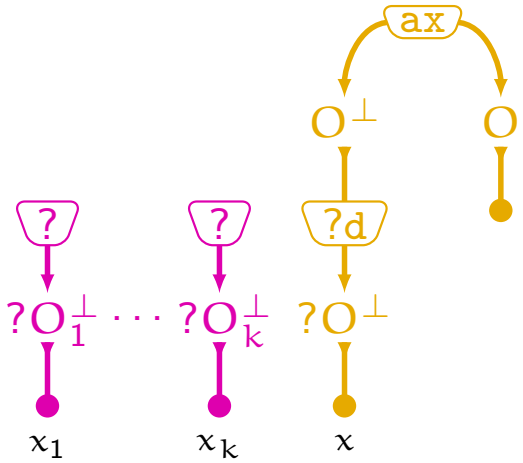
Proof-net $=$ proof-structure of !MELL $+$ $\mathrm{ACC}_{\sharp w}$.

Bang proof-net $=$ proof-net $+$ every input premise of a $\parr/?/\bullet$ node is a conclusion of a $!/?\mathrm{d}/?$ node $+$ input conclusions labeled by distinct variables.

We define by induction a function $(\cdot)^\circ$ mapping every $(\Gamma, t) \in \mathrm{dom}(t_b)$, with $\Gamma = x_1 : !O_1, \ldots, x_k : !O_k$, to a bang proof-net $(\Gamma, t)^\circ$.

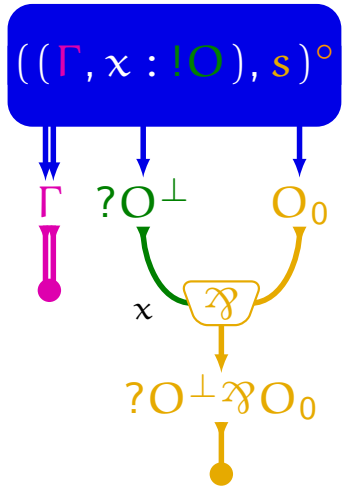
(Di Donna, Guerrieri and Tortora de Falco, technical report)

Translation into proof-nets: variable



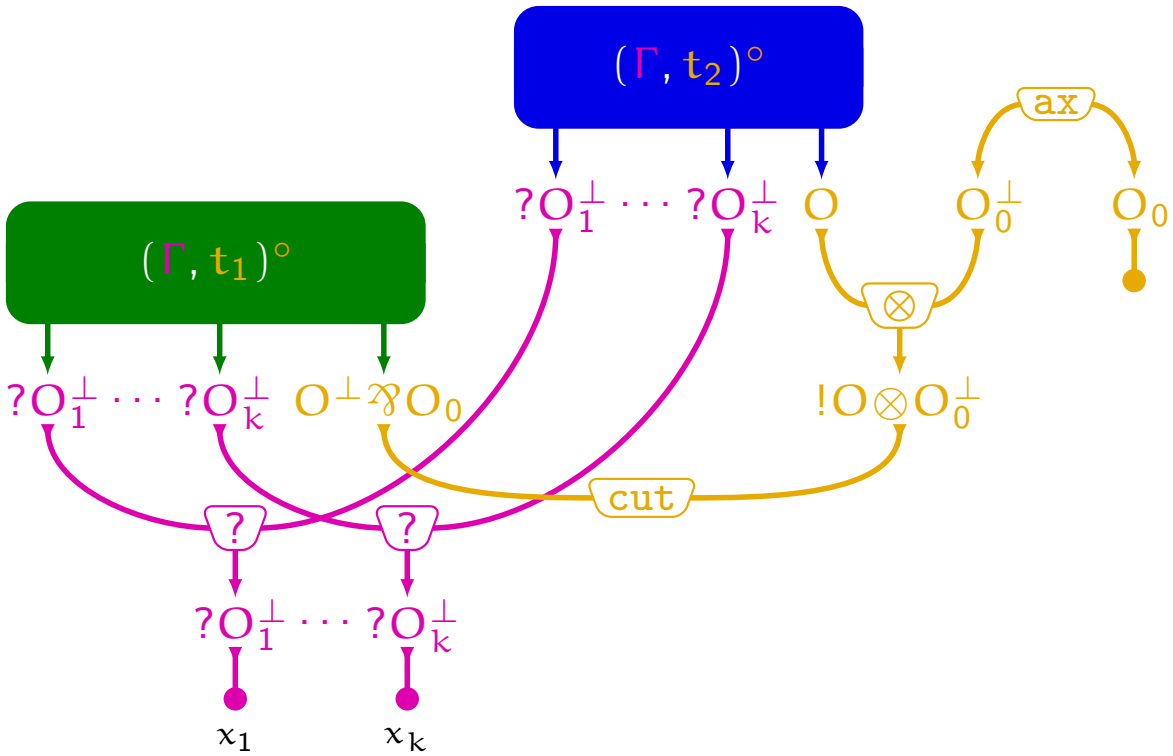
$$((\Gamma, x : !O), x)^\circ.$$

Translation into proof-nets: abstraction



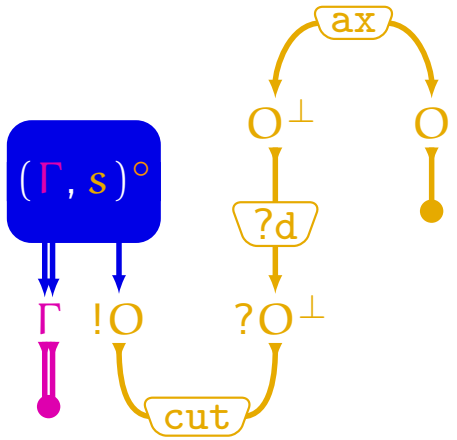
$$(\Gamma, \lambda x^{!O} s)^{\circ}.$$

Translation into proof-nets: application



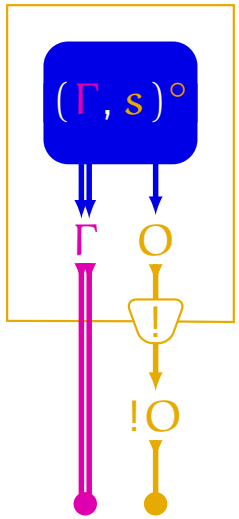
$(\Gamma, \langle t_1 \rangle t_2)^\circ$ is the axiom normal form of this proof-net.

Translation into proof-nets: dereliction



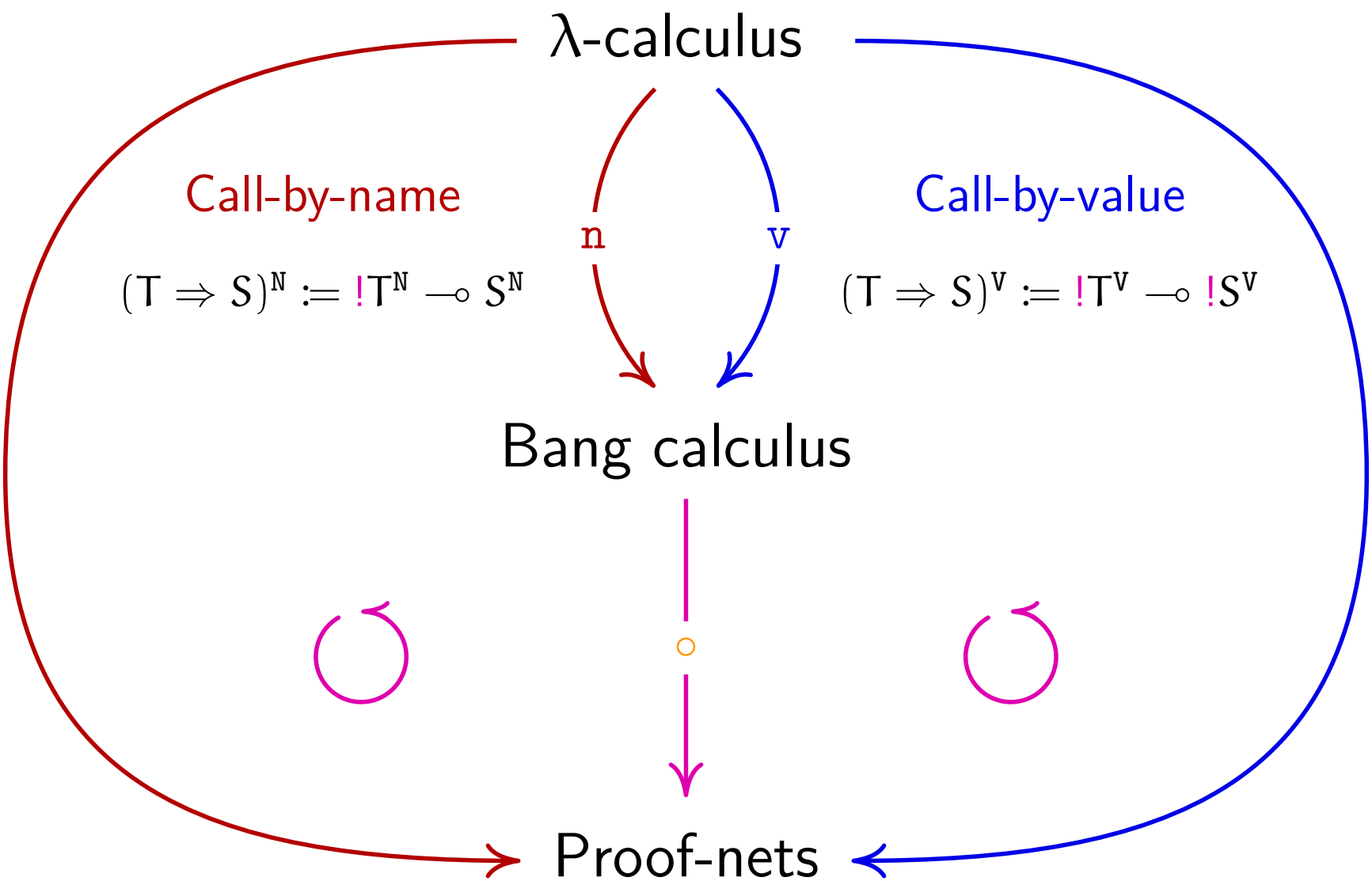
$(\Gamma, \mathrm{der}\, s)^\circ$ is the axiom normal form of this proof-net.

Translation into proof-nets: bang



$$(\Gamma, s^{!})^{\circ}.$$

Factorization



Simulation

Cut elimination in proof-nets simulates bang reduction:

Theorem

For every $(\Gamma, t), (\Gamma, s) \in \mathrm{dom}(t_b)$,
if $t \to_b s$, then $(\Gamma, t)^\circ \to^* (\Gamma, s)^\circ$.

(Di Donna, Guerrieri and Tortora de Falco, technical report)

Characterization of the bang calculus

Let $(\Gamma, t)^{\bullet}$ be the multiplicative normal form of $(\Gamma, t)^{\circ}$.

Characterization of the bang calculus

Let $(\Gamma, t)^\bullet$ be the multiplicative normal form of $(\Gamma, t)^\circ$.

Theorem

Let R be a multiplicative normal bang proof-net with input conclusions of types $?O_1^\perp, \ldots, ?O_k^\perp$ and labelled by $x_1, \ldots, x_k$ respectively. Then there exists a term t such that $R = (\Gamma, t)^\bullet$, where $\Gamma := x_1 : !O_1, \ldots, x_k : !O_k$.

(Di Donna, Guerrieri and Tortora de Falco, sketch of proof)

References

(Plotkin, 1975)

Gordon D. Plotkin.
"Call-by-name, call-by-value and the lambda-calculus".
Theoretical Computer Science, 1(2):125–159, 1975.

(Girard, 1987)

Jean-Yves Girard. "Linear Logic".
Theoretical Computer Science, 50:1-102, 1987.

(Danos and Regnier, 1989)

Vincent Danos and Laurent Regnier.
"The structure of multiplicatives".
Arch. Math. Log. 28.3:181-203, 1989.

References

(Danos, 1990)

Vincent Danos.
"La Logique Linéaire appliquée à l'étude de divers processus de normalisation (principalement du Lambda-calcul)".
PhD thesis, Paris 7, 1990.

(Girard, 1991)

Jean-Yves Girard. "A new constructive logic: classical logic".
Mathematical Structures in Computer Science, 1(3):255–296, 1991.

(Regnier, 1992)

Laurent Regnier. "Lambda-calcul et réseaux".
Thèse de doctorat, Université Paris VII, 1992.

References

(Fleury and Retoré, 1994)

Arnaud Fleury and Christian Retoré. "The Mix Rule".
Mathematical Structures in Computer Science, 4:273-285, 1994.

(Lincoln and Winkler, 1994)

Patrick Lincoln and Timothy C. Winkler.
"Constant-only multiplicative linear logic is NP-complete".
Theoretical Computer Science, 135(1):155–169, 1994.
doi:10.1016/0304-3975(94)00108-1.

(Laurent, 1999)

Olivier Laurent. "Polarized Proof-Nets: Proof-Nets for LC".
Jean-Yves Girard, editor, *Typed Lambda Calculi and Applications, 4th International Conference, TLCA'99, L'Aquila, Italy, April 7-9, 1999, Proceedings*, volume 1581 of *Lecture Notes in Computer Science*, pages 213–227. Springer, 1999. doi:10.1007/3-540-48959-2_16.

References

(Lamarche, 2008)

François Lamarche.
"Proof Nets for Intuitionistic Linear Logic: Essential Nets".
Rapport de recherche inria-00347336, INRIA, 2008.

(Ehrhard, 2016)

Thomas Ehrhard.
"Call-by-push-value from a linear logic point of view".
Peter Thiemann, editor, *Programming Languages and Systems - 25th European Symposium on Programming, ESOP 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings*, Lecture Notes in Computer Science, pages 202–228. Springer, 2016.

References

(Ehrhard and Guerrieri, 2016)

Thomas Ehrhard and Giulio Guerrieri. "The bang calculus: an untyped lambda-calculus generalizing call-by-name and call-by-value".

James Cheney and Germán Vidal, editors, *Proceedings of the 18th International Symposium on Principles and Practice of Declarative Programming, Edinburgh, United Kingdom, September 5-7, 2016*, pages 174–187. ACM, 2016.

(Guerrieri and Manzonetto, 2018)

Giulio Guerrieri and Giulio Manzonetto.
"The Bang Calculus and the Two Girard's Translations".

Proceedings Joint International Workshop on Linearity & Trends in Linear Logic and Applications, Linearity-TLLA@FLoC 2018, volume 292 of EPTCS, pages 15–30, 2018.

References

(Ehrhard and Tasson, 2019)

Thomas Ehrhard and Christine Tasson.
"Probabilistic call by push value".
Logical Methods in Computer Science, 15(1), 2019.

(Di Donna and Tortora de Falco, preprint)

Raffaele Di Donna and Lorenzo Tortora de Falco.
"On the role of connectivity in Linear Logic proofs".
https://arxiv.org/abs/2506.21678.

(Di Donna, Guerrieri and Tortora de Falco, technical report)

Raffaele Di Donna, Giulio Guerrieri and Lorenzo Tortora de Falco.
"Proof-nets and the bang calculus".
https://www.irif.fr/_media/users/didonna/bang_calculus_proof-nets.pdf.

Thank you for your attention!

Extra material

A simplified intuitionistic setting

Fragment ICOMLL of IMELL:

$$\mathrm{O} ::= \mathbf{1} \mid \mathrm{O} \otimes \mathrm{O} \mid \mathrm{O} \parr \mathrm{I} \mid \mathrm{I} \parr \mathrm{O}$$
$$\mathrm{I} ::= \bot \mid \mathrm{I} \parr \mathrm{I} \mid \mathrm{I} \otimes \mathrm{O} \mid \mathrm{O} \otimes \mathrm{I}$$

Correctness in ICOMLL

Theorem. Let R be a cut-free proof-structure of ICOMLL with no ax node. If R satisfies $\mathrm{ACC}_{\sharp \mathrm{w}}$, then R is correct.

(Di Donna and Tortora de Falco, preprint)

Correctness in ICOMLL

Theorem. Let R be a cut-free proof-structure of ICOMLL with no ax node. If R satisfies $\mathrm{ACC}_{\sharp\mathrm{w}}$, then R is correct.

(Di Donna and Tortora de Falco, preprint)

Corollary. A sequent of ICOMLL is provable if and only if it has exactly one output formula.

Correctness in ICOMLL

Theorem. Let R be a cut-free proof-structure of ICOMLL with no ax node. If R satisfies $\mathrm{ACC}_{\sharp\mathrm{w}}$, then R is correct.

(Di Donna and Tortora de Falco, preprint)

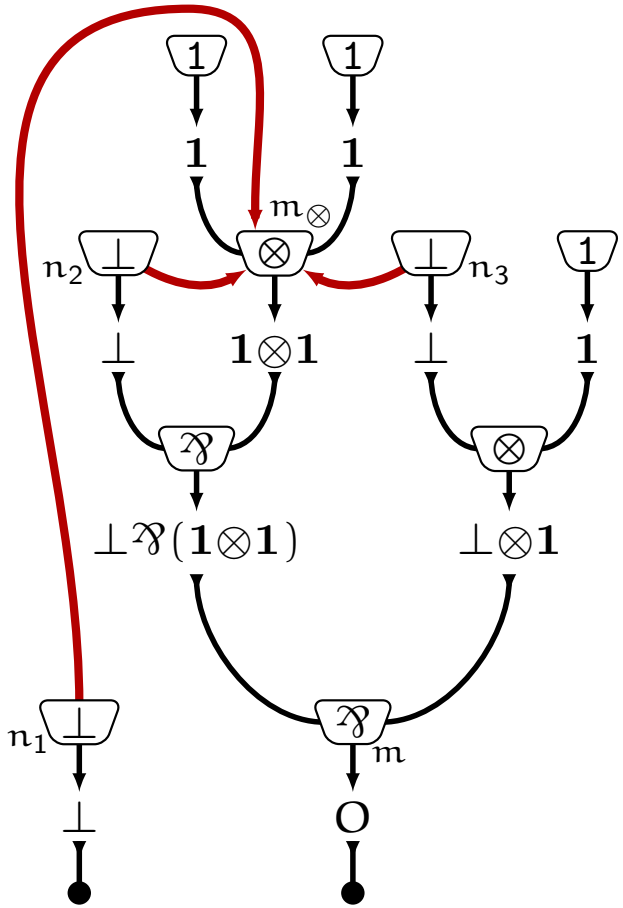
Corollary. A sequent of ICOMLL is provable if and only if it has exactly one output formula.

Thus, provability in ICOMLL is decidable in linear time.

In COMLL, provability is NP-complete.

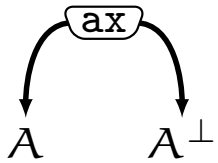
(Lincoln and Winkler, 1994)

Canonical jumps in ICOMLL



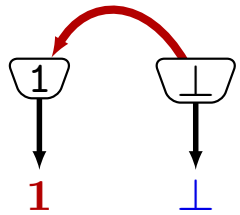
Why ICOMLL?

ax node



proof-structure of IMLL

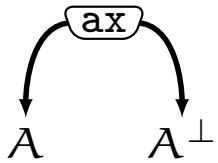
fixed jump



proof-structure of ICOMLL with a partial jump function

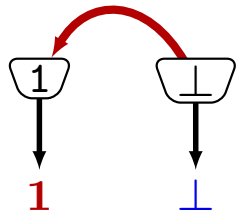
Why ICOMLL?

ax node



proof-structure of IMLL

fixed jump

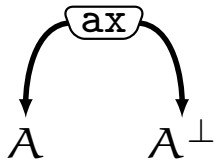


proof-structure of ICOMLL with a partial jump function

We found a solution to the base case (no fixed jumps).

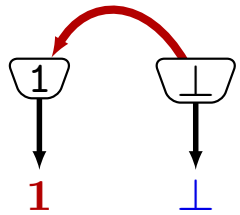
Why ICOMLL?

ax node



proof-structure of IMLL

fixed jump

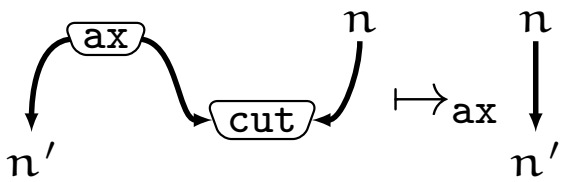


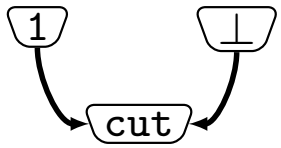
proof-structure of ICOMLL with a partial jump function

We found a solution to the base case (no fixed jumps).

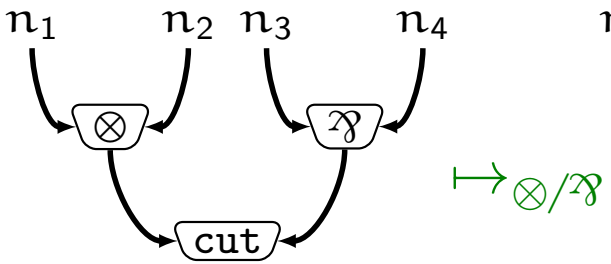
Beyond that: work in progress.

Cut elimination: axiom and multiplicatives

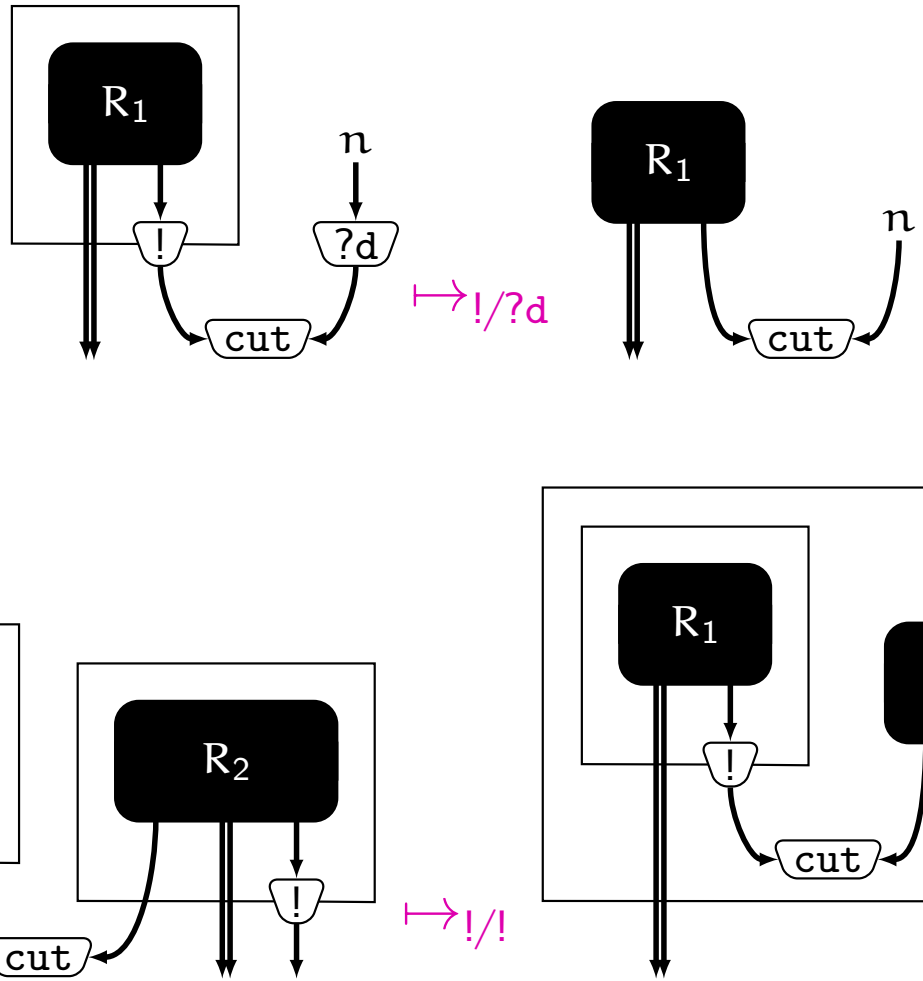




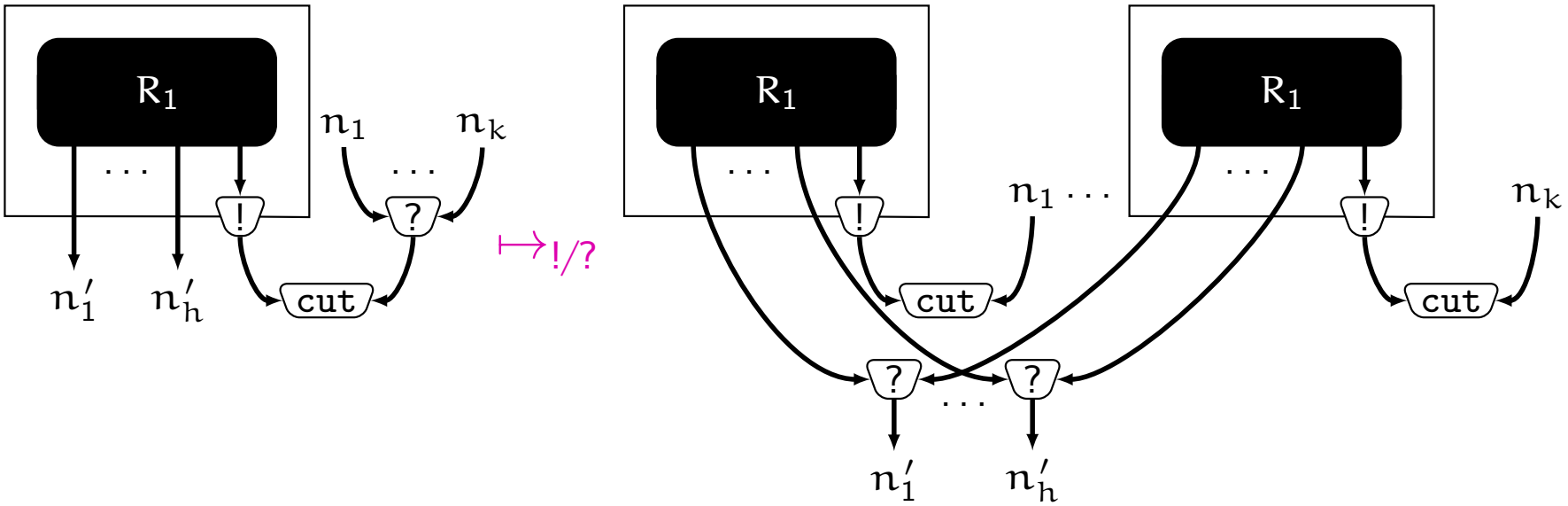
empty



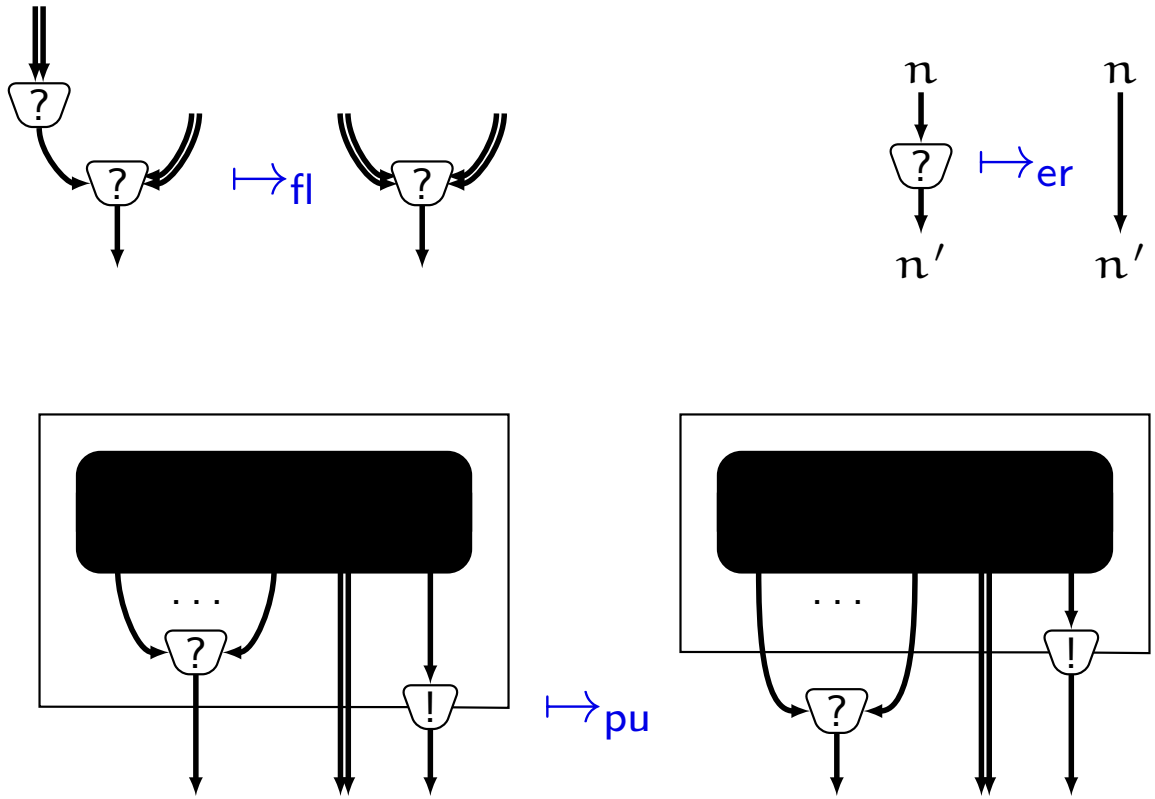
Cut elimination: dereliction and commutation



Cut elimination: why not

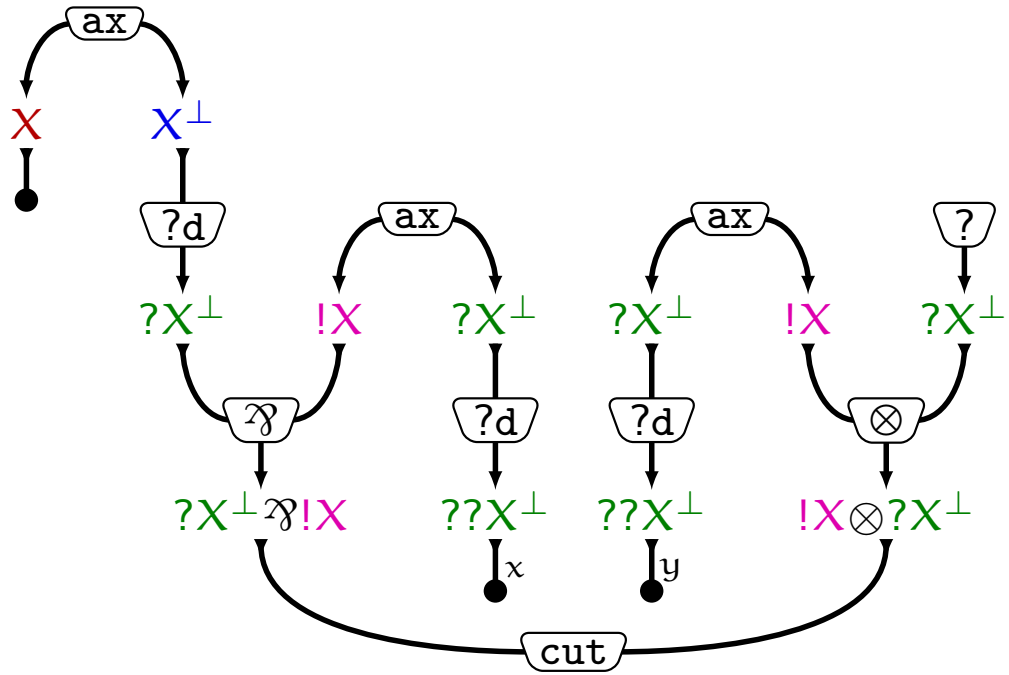


? reduction



Necessity of multiplicative normality

A bang proof-net that is not $(\Gamma, t)^\circ$ for any (Γ, t):

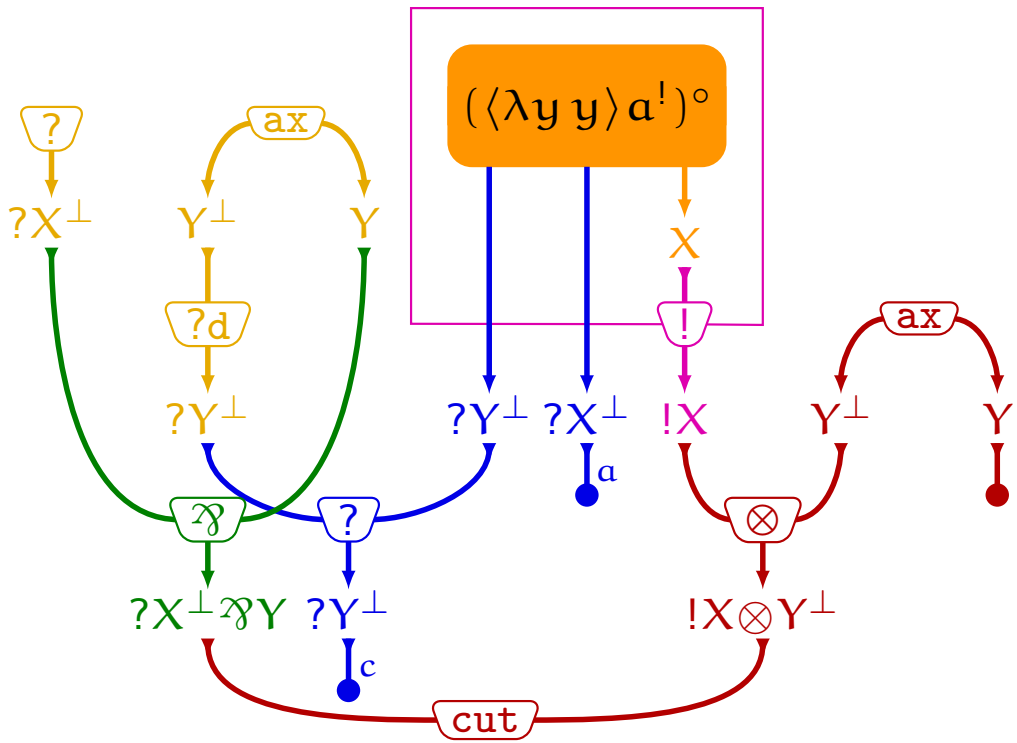


No contradiction with the previous theorem:
this proof-structure is not multiplicative normal.

Example of simulation

$$\dfrac{\dfrac{}{!\Gamma, x : !X \vdash c : Y}\,var}{!\Gamma \vdash \lambda x\, c : !X \multimap Y}\,\lambda \qquad \dfrac{\vdots}{!\Gamma \vdash \langle \lambda y\, y \rangle a^{!} : X}\,!}{!\Gamma \vdash (\langle \lambda y\, y \rangle a^{!})^{!} : !X}$$

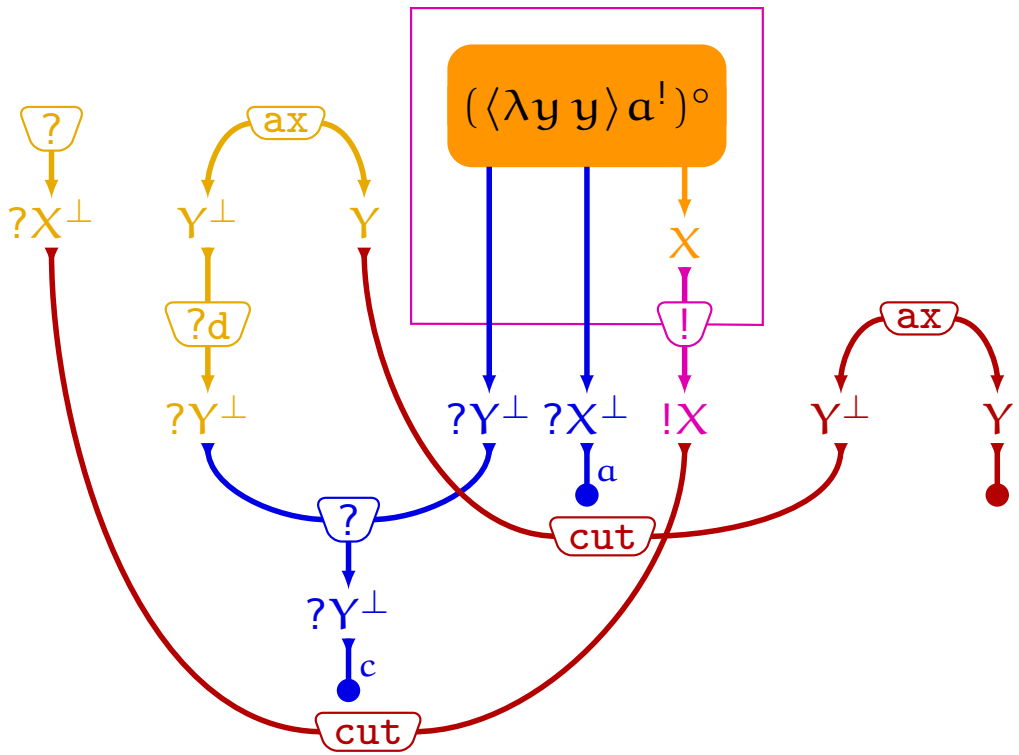
$$\dfrac{}{!\Gamma \vdash \langle \lambda x\, c \rangle (\langle \lambda y\, y \rangle a^{!})^{!} : Y}\,app \qquad \mapsto_{b} \qquad \dfrac{}{!\Gamma, x : !X \vdash c : Y}\,var$$



Example of simulation

$$\dfrac{\dfrac{}{!\Gamma, x : !X \vdash c : Y}\,var}{!\Gamma \vdash \lambda x\, c : !X \multimap Y}\,\lambda \qquad \dfrac{\vdots}{!\Gamma \vdash \langle \lambda y\, y \rangle a^{!} : X}\,!}{!\Gamma \vdash (\langle \lambda y\, y \rangle a^{!})^{!} : !X}$$

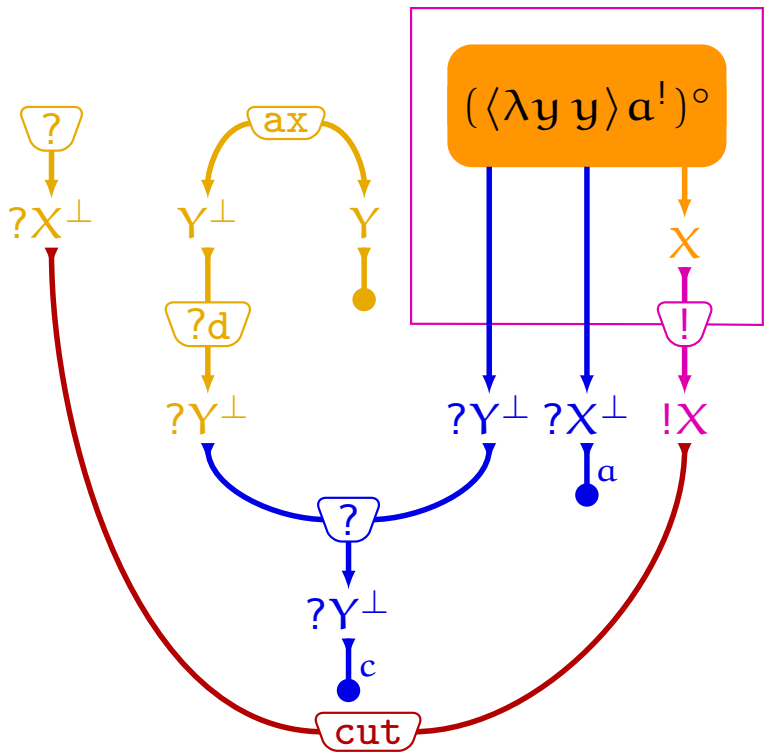
$$\dfrac{\dfrac{\dfrac{}{!\Gamma, x : !X \vdash c : Y}\,var}{!\Gamma \vdash \lambda x\, c : !X \multimap Y}\,\lambda \qquad \dfrac{\vdots}{!\Gamma \vdash \langle \lambda y\, y \rangle a^{!} : X}\,!}{!\Gamma \vdash (\langle \lambda y\, y \rangle a^{!})^{!} : !X}}{!\Gamma \vdash \langle \lambda x\, c \rangle (\langle \lambda y\, y \rangle a^{!})^{!} : Y}\,app \quad \mapsto_{b} \quad \dfrac{}{!\Gamma, x : !X \vdash c : Y}\,var$$



Example of simulation

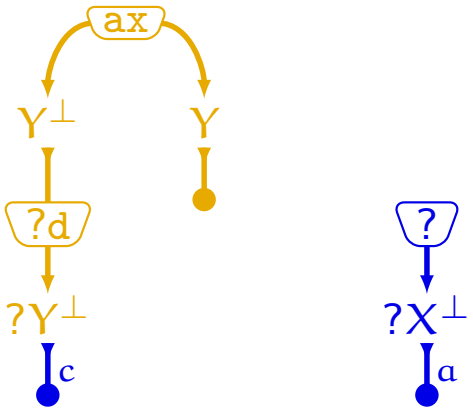
$$\dfrac{\dfrac{\dfrac{}{!\Gamma, x : !X \vdash c : Y}\,var}{!\Gamma \vdash \lambda x\, c : !X \multimap Y}\,\lambda \qquad \dfrac{\vdots}{!\Gamma \vdash \langle \lambda y\, y \rangle a^{!} : X}}{!\Gamma \vdash \langle \lambda x\, c \rangle (\langle \lambda y\, y \rangle a^{!})^{!} : Y}\,app \qquad \dfrac{!\Gamma \vdash \langle \lambda y\, y \rangle a^{!} : X}{!\Gamma \vdash (\langle \lambda y\, y \rangle a^{!})^{!} : !X}\,!$$

$$\mapsto_{b} \quad \dfrac{}{!\Gamma, x : !X \vdash c : Y}\,var$$



Example of simulation

$$\dfrac{\dfrac{\dfrac{}{!\Gamma, x : !X \vdash c : Y}\,var}{!\Gamma \vdash \lambda x\, c : !X \multimap Y}\,\lambda \qquad \dfrac{\vdots}{!\Gamma \vdash \langle \lambda y\, y \rangle a^{!} : X}}{!\Gamma \vdash \langle \lambda x\, c \rangle (\langle \lambda y\, y \rangle a^{!})^{!} : Y}\,app \qquad \dfrac{}{!\Gamma \vdash (\langle \lambda y\, y \rangle a^{!})^{!} : !X}\,! \qquad \mapsto_{b} \qquad \dfrac{}{!\Gamma, x : !X \vdash c : Y}\,var$$



The λ-calculus

$$M ::= v \mid (M)M \qquad (\textit{terms})$$

$$v ::= x \mid \lambda x\, M \qquad (\textit{values})$$

Call-by-name: $(\lambda x\, M)N \mapsto_\beta M\{N/x\}$

Call-by-value: $(\lambda x\, M)v \mapsto_{\beta_v} M\{v/x\}$

(Plotkin, 1975)

The λ-calculus

$$M ::= v \mid (M)M \qquad (\textit{terms})$$
$$v ::= x \mid \lambda x\, M \qquad (\textit{values})$$

Call-by-name: $(\lambda x\, M)N \mapsto_\beta M\{N/x\}$

Call-by-value: $(\lambda x\, M)v \mapsto_{\beta_v} M\{v/x\}$

(Plotkin, 1975)

For $r \in \{\beta, \beta_v\}$, we write $\to_r$
for the contextual closure of $\mapsto_r$.

Discarding a resource

Call-by-name

We can discard the argument without evaluating it first:

$$(\lambda x\, c)(\lambda y\, y)\, a \mapsto_\beta c$$

Discarding a resource

Call-by-name

We can discard the argument without evaluating it first:

$$(\lambda x\, c)(\lambda y\, y)a \mapsto_\beta c$$

Call-by-value

Only values can be discarded.

We have to evaluate the argument first:

$$(\lambda x\, c)(\lambda y\, y)a \to_{\beta_v} (\lambda x\, c)a$$

Discarding a resource

Call-by-name

We can discard the argument without evaluating it first:

$$(\lambda x\, c)(\lambda y\, y)a \mapsto_\beta c$$

Call-by-value

Only values can be discarded.

We have to evaluate the argument first:

$$(\lambda x\, c)(\lambda y\, y)a \to_{\beta_v} (\lambda x\, c)a$$

$$\mapsto_{\beta_v} c$$

Duplicating a resource

Call-by-name

We can duplicate the argument without evaluating it first:

$$(\lambda x\,((f)x)x)(\lambda y\,y)a \mapsto_\beta ((f)(\lambda y\,y)a)(\lambda y\,y)a$$

Duplicating a resource

Call-by-name

We can duplicate the argument without evaluating it first:

$$(\lambda x\,((f)x)x)(\lambda y\,y)a \mapsto_\beta ((f)(\lambda y\,y)a)(\lambda y\,y)a$$
$$\to_{\beta_v} ((f)a)(\lambda y\,y)a$$

Duplicating a resource

Call-by-name

We can duplicate the argument without evaluating it first:

$$(\lambda x\,((f)x)x)(\lambda y\,y)a \mapsto_\beta ((f)(\lambda y\,y)a)(\lambda y\,y)a$$
$$\to_{\beta_v} ((f)a)(\lambda y\,y)a$$
$$\to_{\beta_v} ((f)a)a$$

Duplicating a resource

Call-by-name

We can duplicate the argument without evaluating it first:

$$\begin{aligned}(\lambda x\,((f)x)x)(\lambda y\,y)a &\mapsto_\beta ((f)(\lambda y\,y)a)(\lambda y\,y)a\\ &\to_{\beta_v} ((f)a)(\lambda y\,y)a\\ &\to_{\beta_v} ((f)a)a\end{aligned}$$

Call-by-value

Only values can be duplicated.

We have to evaluate the argument first:

$$(\lambda x\,((f)x)x)(\lambda y\,y)a \to_{\beta_v} (\lambda x\,((f)x)x)a$$

Duplicating a resource

Call-by-name

We can duplicate the argument without evaluating it first:

$$\begin{aligned}(\lambda x\,((f)x)x)(\lambda y\,y)a &\mapsto_\beta ((f)(\lambda y\,y)a)(\lambda y\,y)a\\ &\to_{\beta_v} ((f)a)(\lambda y\,y)a\\ &\to_{\beta_v} ((f)a)a\end{aligned}$$

Call-by-value

Only values can be duplicated.

We have to evaluate the argument first:

$$\begin{aligned}(\lambda x\,((f)x)x)(\lambda y\,y)a &\to_{\beta_v} (\lambda x\,((f)x)x)a\\ &\mapsto_{\beta_v} ((f)a)a\end{aligned}$$

Call-by-name λ-calculus $\to$ bang calculus: translation

$$((\lambda x\, c)(\lambda y\, y)a)^{n} = \langle (\lambda x\, c)^{n} \rangle (((\lambda y\, y)a)^{n})^{!}$$

Call-by-name λ-calculus → bang calculus: translation

$$\begin{aligned}((\lambda x\, c)(\lambda y\, y)a)^{n} &= \langle(\lambda x\, c)^{n}\rangle(((\lambda y\, y)a)^{n})^{!} \\ &= \langle\lambda x\, c^{n}\rangle(((\lambda y\, y)a)^{n})^{!}\end{aligned}$$

Call-by-name λ-calculus $\to$ bang calculus: translation

$$
\begin{aligned}
((\lambda x\, c)(\lambda y\, y)a)^{n} &= \langle (\lambda x\, c)^{n} \rangle (((\lambda y\, y)a)^{n})^{!} \\
&= \langle \lambda x\, c^{n} \rangle (((\lambda y\, y)a)^{n})^{!} \\
&= \langle \lambda x\, c \rangle (((\lambda y\, y)a)^{n})^{!}
\end{aligned}
$$

Call-by-name λ-calculus → bang calculus: translation

$$
\begin{aligned}
((\lambda x\, c)(\lambda y\, y)a)^{n} &= \langle (\lambda x\, c)^{n} \rangle (((\lambda y\, y)a)^{n})^{!} \\
&= \langle \lambda x\, c^{n} \rangle (((\lambda y\, y)a)^{n})^{!} \\
&= \langle \lambda x\, c \rangle (((\lambda y\, y)a)^{n})^{!} \\
&= \langle \lambda x\, c \rangle (\langle (\lambda y\, y)^{n} \rangle (a^{n})^{!})^{!}
\end{aligned}
$$

Call-by-name λ-calculus → bang calculus: translation

$$
\begin{aligned}
((\lambda x\, c)(\lambda y\, y)a)^{n} &= \langle (\lambda x\, c)^{n} \rangle (((\lambda y\, y)a)^{n})^{!} \\
&= \langle \lambda x\, c^{n} \rangle (((\lambda y\, y)a)^{n})^{!} \\
&= \langle \lambda x\, c \rangle (((\lambda y\, y)a)^{n})^{!} \\
&= \langle \lambda x\, c \rangle (\langle (\lambda y\, y)^{n} \rangle (a^{n})^{!})^{!} \\
&= \langle \lambda x\, c \rangle (\langle \lambda y\, y^{n} \rangle (a^{n})^{!})^{!}
\end{aligned}
$$

Call-by-name λ-calculus → bang calculus: translation

$$
\begin{aligned}
((\lambda x\, c)(\lambda y\, y)a)^n &= \langle (\lambda x\, c)^n \rangle (((\lambda y\, y)a)^n)^! \\
&= \langle \lambda x\, c^n \rangle (((\lambda y\, y)a)^n)^! \\
&= \langle \lambda x\, c \rangle (((\lambda y\, y)a)^n)^! \\
&= \langle \lambda x\, c \rangle (\langle (\lambda y\, y)^n \rangle (a^n)^!)^! \\
&= \langle \lambda x\, c \rangle (\langle \lambda y\, y^n \rangle (a^n)^!)^! \\
&= \langle \lambda x\, c \rangle (\langle \lambda y\, y \rangle (a^n)^!)^!
\end{aligned}
$$

Call-by-name λ-calculus $\to$ bang calculus: translation

$$
\begin{aligned}
((\lambda x\, c)(\lambda y\, y)a)^n &= \langle (\lambda x\, c)^n \rangle (((\lambda y\, y)a)^n)^! \\
&= \langle \lambda x\, c^n \rangle (((\lambda y\, y)a)^n)^! \\
&= \langle \lambda x\, c \rangle (((\lambda y\, y)a)^n)^! \\
&= \langle \lambda x\, c \rangle (\langle (\lambda y\, y)^n \rangle (a^n)^!)^! \\
&= \langle \lambda x\, c \rangle (\langle \lambda y\, y^n \rangle (a^n)^!)^! \\
&= \langle \lambda x\, c \rangle (\langle \lambda y\, y \rangle (a^n)^!)^! \\
&= \langle \lambda x\, c \rangle (\langle \lambda y\, y \rangle a^!)^!
\end{aligned}
$$

Call-by-name λ-calculus $\to$ bang calculus: simulation

The call-by-name reduction step:

$$(\lambda x\, c)(\lambda y\, y)\, a \mapsto_\beta c$$

is simulated by the bang reduction step:

$$\langle \lambda x\, c \rangle (\langle \lambda y\, y \rangle a^!)^! \mapsto_b c$$

Call-by-value λ-calculus → bang calculus: translation

$$((\lambda x\, c)(\lambda y\, y)a)^{v} = \langle \mathsf{der}(\lambda x\, c)^{v} \rangle ((\lambda y\, y)a)^{v}$$

Call-by-value λ-calculus → bang calculus: translation

$$\begin{aligned}((\lambda x\, c)(\lambda y\, y)a)^{v} &= \langle \mathsf{der}(\lambda x\, c)^{v} \rangle ((\lambda y\, y)a)^{v} \\ &= \langle \mathsf{der}(\lambda x\, c^{v})^{!} \rangle ((\lambda y\, y)a)^{v}\end{aligned}$$

Call-by-value λ-calculus → bang calculus: translation

$$
\begin{aligned}
((\lambda x\, c)(\lambda y\, y)a)^{v} &= \langle \mathsf{der}(\lambda x\, c)^{v} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{v})^{!} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle ((\lambda y\, y)a)^{v}
\end{aligned}
$$

Call-by-value λ-calculus → bang calculus: translation

$$
\begin{aligned}
((\lambda x\, c)(\lambda y\, y)a)^{v} &= \langle \mathsf{der}(\lambda x\, c)^{v} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{v})^{!} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle \langle \mathsf{der}(\lambda y\, y)^{v} \rangle a^{v}
\end{aligned}
$$

Call-by-value λ-calculus → bang calculus: translation

$$
\begin{aligned}
((\lambda x\, c)(\lambda y\, y)a)^{v} &= \langle \mathsf{der}(\lambda x\, c)^{v} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{v})^{!} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle \langle \mathsf{der}(\lambda y\, y)^{v} \rangle a^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle \langle \mathsf{der}(\lambda y\, y^{v})^{!} \rangle a^{v}
\end{aligned}
$$

Call-by-value λ-calculus → bang calculus: translation

$$
\begin{aligned}
((\lambda x\, c)(\lambda y\, y)a)^{v} &= \langle \mathsf{der}(\lambda x\, c)^{v} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{v})^{!} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle \langle \mathsf{der}(\lambda y\, y)^{v} \rangle a^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle \langle \mathsf{der}(\lambda y\, y^{v})^{!} \rangle a^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle \langle \mathsf{der}(\lambda y\, y^{!})^{!} \rangle a^{v}
\end{aligned}
$$

Call-by-value λ-calculus → bang calculus: translation

$$
\begin{aligned}
((\lambda x\, c)(\lambda y\, y)a)^{v} &= \langle \mathsf{der}(\lambda x\, c)^{v} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{v})^{!} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle ((\lambda y\, y)a)^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle \langle \mathsf{der}(\lambda y\, y)^{v} \rangle a^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle \langle \mathsf{der}(\lambda y\, y^{v})^{!} \rangle a^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle \langle \mathsf{der}(\lambda y\, y^{!})^{!} \rangle a^{v} \\
&= \langle \mathsf{der}(\lambda x\, c^{!})^{!} \rangle \langle \mathsf{der}(\lambda y\, y^{!})^{!} \rangle a^{!}
\end{aligned}
$$

Call-by-value λ-calculus → bang calculus: simulation

Bang reduction simulates call-by-value reduction:

$$(\lambda x\, c)(\lambda y\, y)\, a \to_{\beta_v} (\lambda x\, c)\, a$$

$$\langle \mathsf{der}(\lambda x\, c^!)^! \rangle \langle \mathsf{der}(\lambda y\, y^!)^! \rangle a^! \to_b \langle \mathsf{der}(\lambda x\, c^!)^! \rangle \langle \lambda y\, y^! \rangle a^!$$

Call-by-value λ-calculus → bang calculus: simulation

Bang reduction simulates call-by-value reduction:

$$(\lambda x\, c)(\lambda y\, y)\, a \to_{\beta_v} (\lambda x\, c)\, a$$

$$\langle \mathsf{der}(\lambda x\, c^{!})^{!}\rangle \langle \mathsf{der}(\lambda y\, y^{!})^{!}\rangle a^{!} \to_{b} \langle \mathsf{der}(\lambda x\, c^{!})^{!}\rangle \langle \lambda y\, y^{!}\rangle a^{!} \to_{b} \langle \mathsf{der}(\lambda x\, c^{!})^{!}\rangle a^{!}$$

Call-by-value λ-calculus → bang calculus: simulation

Bang reduction simulates call-by-value reduction:

$$\begin{aligned}(\lambda x\, c)(\lambda y\, y)a &\to_{\beta_v} (\lambda x\, c)a \\ &\mapsto_{\beta_v} c\end{aligned}$$

$$\begin{aligned}\langle \mathsf{der}(\lambda x\, c^!)^!\rangle\langle \mathsf{der}(\lambda y\, y^!)^!\rangle a^! &\to_b \langle \mathsf{der}(\lambda x\, c^!)^!\rangle\langle \lambda y\, y^!\rangle a^! \\ &\to_b \langle \mathsf{der}(\lambda x\, c^!)^!\rangle a^! \\ &\to_b \langle \lambda x\, c^!\rangle a^!\end{aligned}$$

Call-by-value λ-calculus → bang calculus: simulation

Bang reduction simulates call-by-value reduction:

$$\begin{aligned}(\lambda x\, c)(\lambda y\, y)a &\to_{\beta_v} (\lambda x\, c)a \\ &\mapsto_{\beta_v} c\end{aligned}$$

$$\begin{aligned}\langle \mathsf{der}(\lambda x\, c^!)^!\rangle\langle \mathsf{der}(\lambda y\, y^!)^!\rangle a^! &\to_b \langle \mathsf{der}(\lambda x\, c^!)^!\rangle\langle \lambda y\, y^!\rangle a^! \\ &\to_b \langle \mathsf{der}(\lambda x\, c^!)^!\rangle a^! \\ &\to_b \langle \lambda x\, c^!\rangle a^! \\ &\mapsto_b c^!\end{aligned}$$