Proof nets and mainstream graph theory

Nguyễn Lê Thành Dũng (a.k.a. Tito) — nltd@nguyentito.eu
partially based on joint work with Lutz Straßburger
ANR LambdaComb kickoff meeting, April 11th, 2022

Proof nets in Multiplicative Linear Logic

Proofs-as-programs: Intuitionistic Multiplicative Linear Logic $\simeq$ linear λ-calculus

Here, we work with *classical* MLL: $A \to B = A^\perp \vee B$ (or rather $A \multimap B = A^\perp \parr B$)

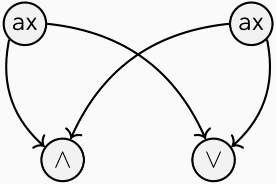
Proof nets in Multiplicative Linear Logic

Proofs-as-programs: Intuitionistic Multiplicative Linear Logic $\simeq$ linear λ-calculus
Here, we work with *classical* MLL: $A \to B = A^\perp \vee B$ (or rather $A \multimap B = A^\perp \parr B$)

A *proof net* is a sort of graph made of ax, $\vee$ and $\wedge$ nodes which represents a proof

- i.e. translated from a sequent calculus proof
- Equivalently, set of proof nets inductively generated

$$\dfrac{\dfrac{\dfrac{}{\vdash A, A^\perp}\,\mathsf{ax} \qquad \dfrac{}{\vdash B, B^\perp}\,\mathsf{ax}}{\vdash A \wedge B, A^\perp, B^\perp}\,\wedge}{\vdash A \wedge B, A^\perp \vee B^\perp}\,\vee$$



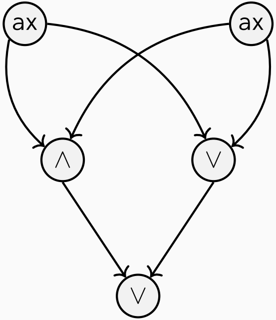
Proof nets in Multiplicative Linear Logic

Proofs-as-programs: Intuitionistic Multiplicative Linear Logic $\simeq$ linear λ-calculus
Here, we work with *classical* MLL: $A \to B = A^\perp \vee B$ (or rather $A \multimap B = A^\perp ⅋ B$)

A *proof net* is a sort of graph made of ax, $\vee$ and $\wedge$ nodes which represents a proof

- i.e. translated from a sequent calculus proof
- Equivalently, set of proof nets inductively generated

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash A, A^\perp}\,\mathsf{ax} \qquad \dfrac{}{\vdash B, B^\perp}\,\mathsf{ax}}{\vdash A \wedge B, A^\perp, B^\perp}\,\wedge}{\vdash A \wedge B, A^\perp \vee B^\perp}\,\vee}{\vdash (A \wedge B) \vee (A^\perp \vee B^\perp)}\,\vee$$



Proof nets vs proof structures

- Proof structures: graphs made of ax-nodes, $\wedge$-nodes and $\vee$-nodes
- Not all proof structures are proof nets!
  Some are not images of any sequent calculus proof

Problem (Correctness)

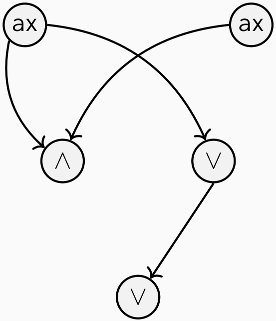
Given a proof structure, decide whether it is a proof net.

Related to *correctness criteria*:

non-inductive combinatorial characterizations of proof nets among proof structures

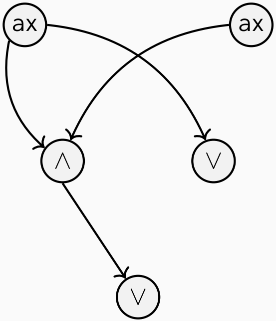
The Danos–Regnier correctness criterion for MLL

Delete 1 of the 2 premises of each $\vee$-node; do you always get an (undirected) *tree*?
If so, then you've got an MLL proof net



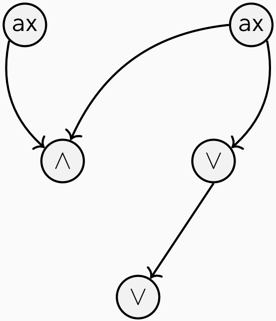
The Danos–Regnier correctness criterion for MLL

Delete 1 of the 2 premises of each $\vee$-node; do you always get an (undirected) *tree*?
If so, then you've got an MLL proof net



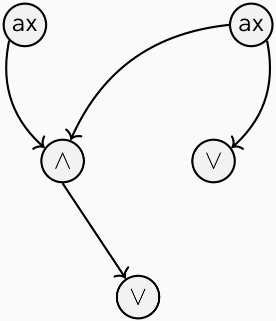
The Danos–Regnier correctness criterion for MLL

Delete 1 of the 2 premises of each $\vee$-node; do you always get an (undirected) *tree*?
If so, then you've got an MLL proof net



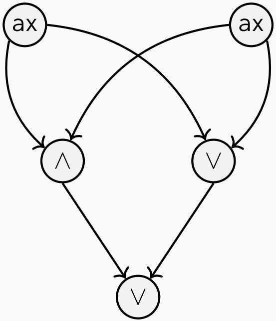
The Danos–Regnier correctness criterion for MLL

Delete 1 of the 2 premises of each $\vee$-node; do you always get an (undirected) *tree*?
If so, then you've got an MLL proof net



The Danos–Regnier correctness criterion for MLL+Mix

Delete 1 of the 2 premises of each $\vee$-node; do you always get a *tree* (resp. forest)?
If so, then you've got an MLL (resp. MLL+Mix) proof net



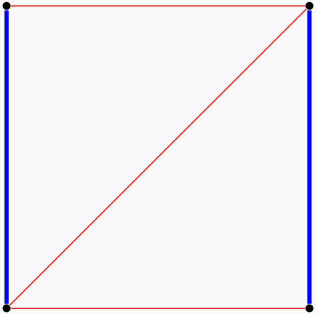
Mix rule: $\dfrac{\vdash \Gamma \quad \vdash \Delta}{\vdash \Gamma, \Delta}$

A graph-theoretic viewpoint

- Forest = acyclic graph
- MLL+Mix correct = no cycle crossing both premises of a $\vee$-node
- So this is a constrained path-finding / cycle-finding problem
  - Several such problems have been studied in graph theory
  - Next: an example

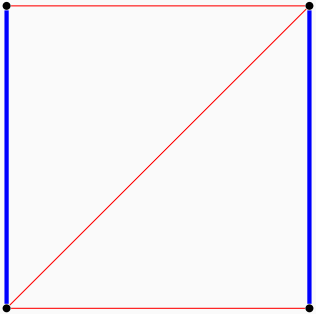
Perfect matchings (1)

- A classical topic in graph theory and combinatorial optimisation
- A *perfect matching* is a set of edges in an undirected graph such that each vertex is incident to exactly one edge in the matching
- Example below: blue edges form a perfect matching



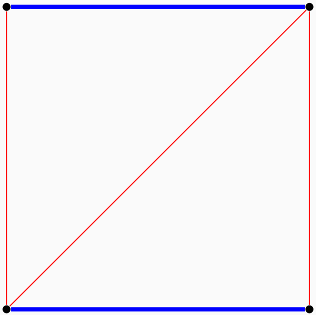
Perfect matchings (2)

- An *alternating path* is a path without vertex repetitions,
  which alternates between edges inside and outside the matching
- Analogous notion of *alternating cycle*
- *Berge's lemma:* $\exists$ alternating cycle $\iff$ the perfect matching is not *unique*



Perfect matchings (2)

- An *alternating path* is a path without vertex repetitions, which alternates between edges inside and outside the matching
- Analogous notion of *alternating cycle*
- *Berge's lemma:* $\exists$ alternating cycle $\iff$ the perfect matching is not *unique*



Proof net correctness vs perfect matching uniqueness

- Alternating paths / cycles in perfect matchings are “equivalent” to many kinds of constrained paths / cycles in graph theory
  - See e.g. Szeider, *On theorems equivalent with Kotzig’s result* [...], 2004
  - or my own arXiv note *Constrained path-finding and structure from acyclicity*
- Is it also the case for MLL+Mix correctness?

Proof net correctness vs perfect matching uniqueness

- Alternating paths / cycles in perfect matchings are "equivalent" to many kinds of constrained paths / cycles in graph theory
  - See e.g. Szeider, *On theorems equivalent with Kotzig's result* [...], 2004
  - or my own arXiv note *Constrained path-finding and structure from acyclicity*
- Is it also the case for MLL+Mix correctness? YES
- A connection was found by Christian Retoré in the 1990s
  R&B-graphs: {proof structures} $\to$ {graphs equipped with perfect matchings}

Theorem (Retoré's correctness criterion)

A proof structure is a MLL+Mix proof net iff the perfect matching of its R&B-graph is unique (i.e. has no alternating cycle).

On sequentialization theorems

- *Sequentialization theorem*: correct proof structures are proof nets, i.e. come from sequent calculus proofs
- A remark by Retoré: this can be reproved from the theorem below

Theorem (Kotzig 1959)

Every unique perfect matching contains a bridge.

On sequentialization theorems

- *Sequentialization theorem*: correct proof structures are proof nets, i.e. come from sequent calculus proofs
- A remark by Retoré: this can be reproved from the theorem below

Theorem (Kotzig 1959)

Every unique perfect matching contains a bridge.

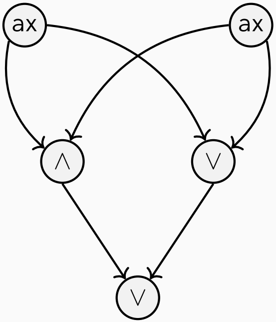
A slight mismatch: no bijection between *sequentializations*, i.e.

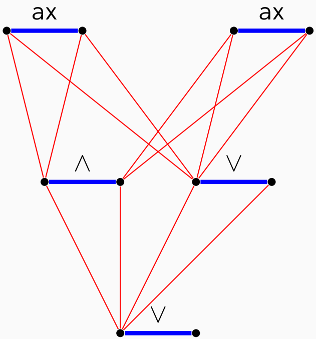
- sequent calculus proofs that map to a proof net
- ways to build up inductively a graph with unique PM by adding bridges

We fix this with another reduction $\{$proof structures$\} \to \{$graphs w/ PMs$\}$: *graphification*

Graphification of proof structures (1)

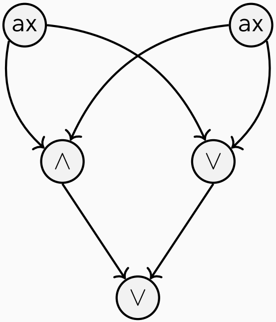
Matching edges correspond to nodes; *bridges* correspond to *splitting terminal nodes*

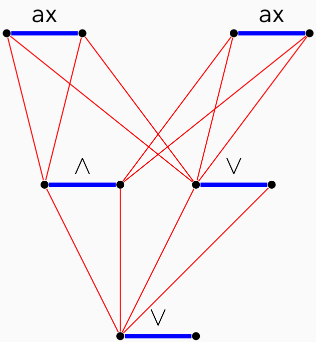




Graphification of proof structures (1)

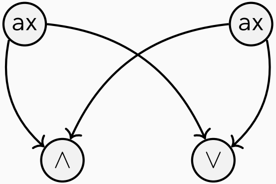
Matching edges correspond to nodes; *bridges* correspond to *splitting terminal nodes*

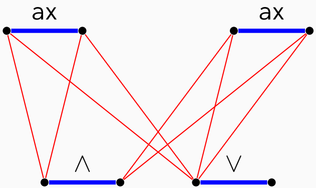




Graphification of proof structures (1)

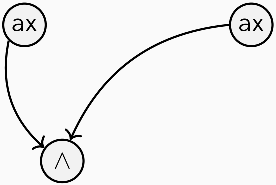
Matching edges correspond to nodes; *bridges* correspond to *splitting terminal nodes*

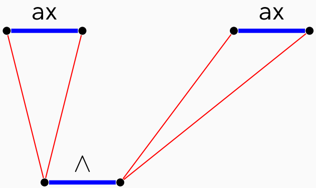




Graphification of proof structures (1)

Matching edges correspond to nodes; *bridges* correspond to *splitting terminal nodes*



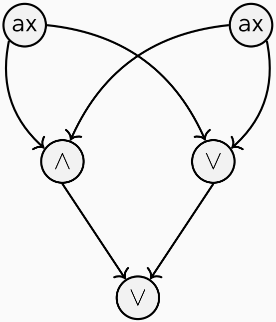


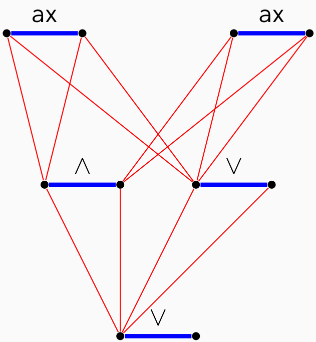
Graphification of proof structures (1)

Matching edges correspond to nodes; *bridges* correspond to *splitting terminal nodes*

Graphification of proof structures (1)

Matching edges correspond to nodes; *bridges* correspond to *splitting terminal nodes*





- Correctness criterion is still uniqueness of PM i.e. no alternating cycle

Graphifications of proof nets (2)

Theorem

The sequentializations of a proof structure are in bijection with those of its graphification.

In particular if one set is $\neq \varnothing$ so is the other, therefore:

Corollary (Sequentialization theorem for MLL+Mix)

Danos–Regnier acyclic $\iff$ MLL+Mix sequentializable.

Graphifications of proof nets (2)

Theorem

The sequentializations of a proof structure are in bijection with those of its graphification.

In particular if one set is $\neq \varnothing$ so is the other, therefore:

Corollary (Sequentialization theorem for MLL+Mix)

Danos–Regnier acyclic $\iff$ MLL+Mix sequentializable.

Bonus: quasi-linear time algorithm to *compute* a sequentialization,
relying on recent graph algorithms developments

Holm, Rotenberg & Thorup, Dynamic bridge-finding in $\tilde{O}(\log^2 n)$ amortized time, 2018

Graphifications of proof nets (2)

Theorem

The sequentializations of a proof structure are in bijection with those of its graphification.

In particular if one set is $\neq \varnothing$ so is the other, therefore:

Corollary (Sequentialization theorem for MLL+Mix)

Danos–Regnier acyclic $\iff$ *MLL+Mix sequentializable.*

Bonus: quasi-linear time algorithm to *compute* a sequentialization,
relying on recent graph algorithms developments

Holm, Rotenberg & Thorup, Dynamic bridge-finding in $\tilde{O}(\log^2 n)$ amortized time, 2018

Next: some structural combinatorics, then more complexity

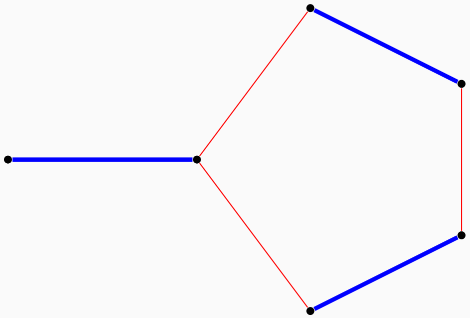
Blossoms in matching theory

A key concept in combinatorial matching algorithms, e.g. testing PM uniqueness

Edmonds, *Paths, trees and flowers*, Canadian J. Math., 1965

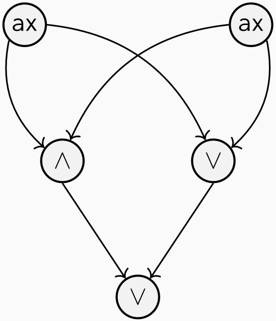
Definition

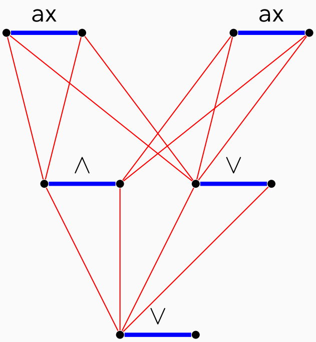
A *blossom* is a cycle with exactly one vertex matched outside the cycle.



Blossoms vs. dependencies

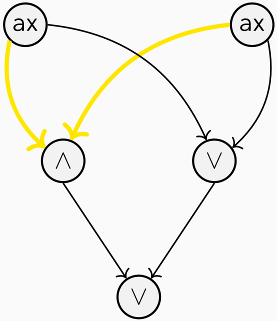
Blossoms of graphification $\rightsquigarrow$ subformulae and dependencies

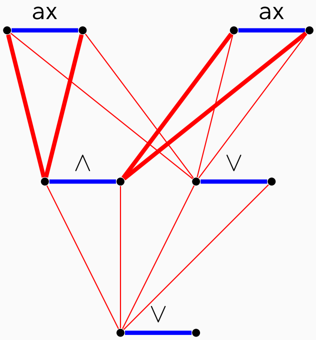




Blossoms vs. dependencies

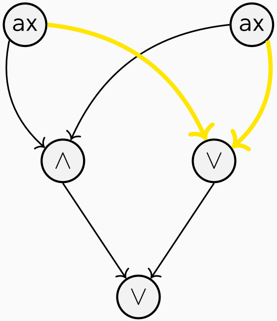
Blossoms of graphification $\rightsquigarrow$ **subformulae** and dependencies

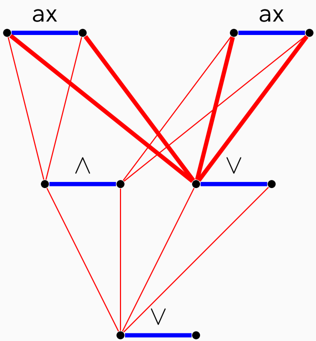




Blossoms vs. dependencies

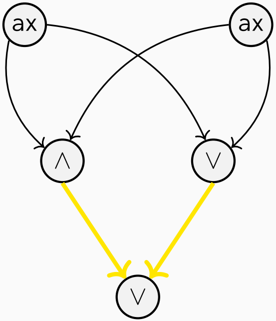
Blossoms of graphification $\rightsquigarrow$ **subformulae** and dependencies

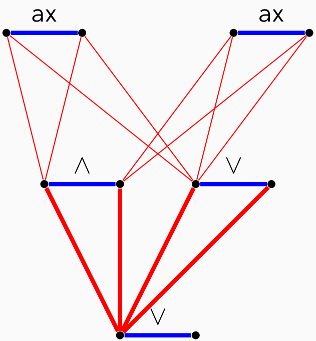




Blossoms vs. dependencies

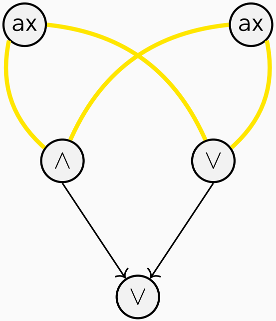
Blossoms of graphification $\rightsquigarrow$ **subformulae** and dependencies

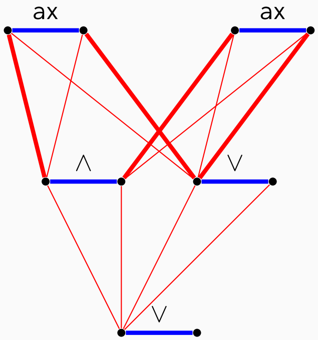




Blossoms vs. dependencies

Blossoms of graphification $\rightsquigarrow$ subformulae and **dependencies**

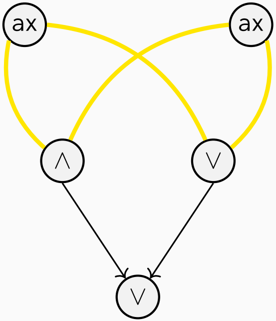


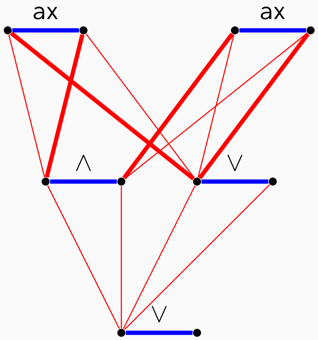


__**Definition:**__ A $\vee$-node l *depends upon* a node l' if there is a Danos–Regnier path between the premises of l going through l'.

Blossoms vs. dependencies

Blossoms of graphification $\rightsquigarrow$ subformulae and **dependencies**





<u>Definition:</u> A $\vee$-node l *depends upon* a node l' if there is a Danos–Regnier path between the premises of l going through l'.

Kingdom ordering of proof nets and unique PMs

Let π be an MLL+Mix proof net.

Definition (Kingdom ordering of a proof net)

We define $l \ll_\pi l$ iff every sequentialization of π introduces l above l'.

Theorem (Bellin 1997 (rediscovered by Bagnol, Doumane & Saurin 2015))

$\ll_\pi =$ *transitive closure of* (*subformula relation*) $\cup$ (*dependency relation*)

Kingdom ordering of proof nets and unique PMs

Let π be an MLL+Mix proof net.

Definition (Kingdom ordering of a proof net)

We define $l \ll_\pi l$ iff every sequentialization of π introduces l above l'.

Theorem (Bellin 1997 (rediscovered by Bagnol, Doumane & Saurin 2015))

$\ll_\pi =$ *transitive closure of* (*subformula relation*) $\cup$ (*dependency relation*)

The kingdom ordering can be defined for unique perfect matchings

(Natural concept, similar things studied in combinatorics

e.g. perfect elimination orderings of chordal graphs)

Theorem (Equivalent graph-theoretic version)

Kingdom ordering = "blossom reachability"

$\rightsquigarrow$ A non-artificial graph-theoretic result coming from linear logic!

The statement is even simplified by moving from proofs to graphs

Complexity of correctness

Uniqueness of a perfect matching can be tested in linear time (Gabow, Kaplan & Tarjan 1999) so using graphification – or Retoré's R&B-graphs from the 1990s – we have:

Corollary (first stated in my FSCD'18 paper?)

MLL+Mix correctness is decidable in linear time.

Previously in the literature: very sophisticated approaches for MLL without Mix
(Guerrini 1999, Murawski & Ong 2000)

Complexity of correctness

Uniqueness of a perfect matching can be tested in linear time (Gabow, Kaplan & Tarjan 1999) so using graphification – or Retoré's R&B-graphs from the 1990s – we have:

Corollary (first stated in my FSCD'18 paper?)

MLL+Mix correctness is decidable in linear time.

Previously in the literature: very sophisticated approaches for MLL without Mix

(Guerrini 1999, Murawski & Ong 2000)

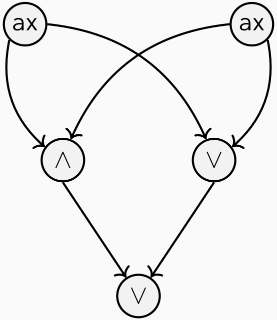
A polynomial-time algorithm for PM uniqueness is already non-trivial

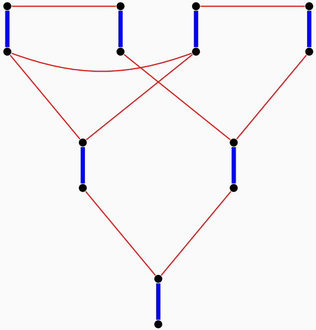
(essentially the blossoms paper Edmonds 1965 – note: [GKT99] also use blossoms)

and breaks down in *directed* graphs → final topic of this talk

R&B-graphs and pomset logic (from Retoré's PhD thesis)

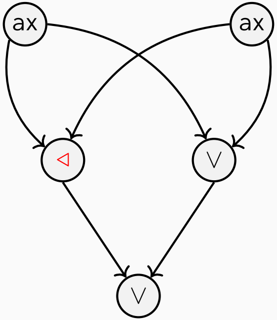
Pomset logic proof nets are best explained through Retoré's R&B-graphs:

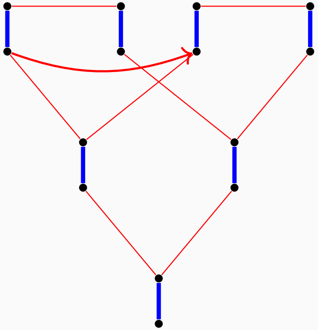




R&B-graphs and pomset logic (from Retoré's PhD thesis)

Pomset logic proof nets are best explained through Retoré's R&B-graphs:



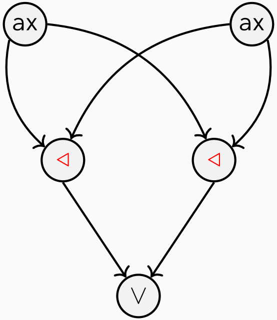


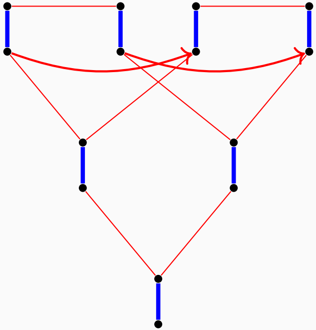
Add *directed* edges to handle a *non-commutative* connective $\lhd$

$$A \wedge B \implies A \lhd B \implies A \vee B$$

R&B-graphs and pomset logic (from Retoré's PhD thesis)

Pomset logic proof nets are best explained through Retoré's R&B-graphs:





Add *directed* edges to handle a *non-commutative* connective $\lhd$

$$A \land B \implies A \lhd B \implies A \lor B$$

Complexity of pomset logic

So we have a reduction $\{$pomset proof structures$\} \to \{$*directed* graphs w/ PMs$\}$.
There's also a converse reduction, so:

Theorem

Pomset logic proof net correctness is coNP-*complete.*

Proof.

Reduction 3SAT $\to$ directed alternating cycle $\to$ pomset proof net *in*correctness. $\square$

Main inspiration: literature on edge-colored graphs, closely related to matchings

(here: Gourvès et al., *Complexity of trails, paths and circuits in arc-colored digraphs*, 2013)

Complexity of pomset logic

So we have a reduction {pomset proof structures} $\to$ {*directed* graphs w/ PMs}.
There's also a converse reduction, so:

Theorem

Pomset logic proof net correctness is coNP*-complete.*

Proof.

Reduction 3SAT $\to$ directed alternating cycle $\to$ pomset proof net *in*correctness. $\square$

Main inspiration: literature on edge-colored graphs, closely related to matchings

(here: Gourvès et al., *Complexity of trails, paths and circuits in arc-colored digraphs*, 2013)

And with a bit more work:

Theorem (N. & Straßburger, upcoming journal paper)

Deciding the provability *of a pomset logic formula is* Σ_2^p*-complete.*

(*second level of the polynomial hierarchy*)

Pomset Logic vs system BV

Guglielmi's *system BV* is a logic over the same language of formulas as pomset logic (PL), historically important as the origin of the *deep inference* paradigm in proof theory

A two-decades-old conjecture

These logics are equivalent, i.e. prove the same formulas.

It was known that $(\mathrm{BV} \vdash A) \implies (\mathrm{PL} \vdash A)$.

Pomset Logic vs system BV

Guglielmi's *system BV* is a logic over the same language of formulas as pomset logic (PL), historically important as the origin of the *deep inference* paradigm in proof theory

A two-decades-old conjecture

These logics are equivalent, i.e. prove the same formulas.

It was known that $(\mathrm{BV} \vdash A) \implies (\mathrm{PL} \vdash A)$.

But BV provability is NP-complete: strictly easier than for PL unless $\mathsf{NP} = \mathsf{coNP}$!

Pomset Logic vs system BV

Guglielmi's *system BV* is a logic over the same language of formulas as pomset logic (PL), historically important as the origin of the *deep inference* paradigm in proof theory

A two-decades-old conjecture

These logics are equivalent, i.e. prove the same formulas.

It was known that $(\mathrm{BV} \vdash A) \implies (\mathrm{PL} \vdash A)$.

But BV provability is NP-complete: strictly easier than for PL unless $\mathsf{NP} = \mathsf{coNP}$!

Unconditional <u>refutation</u> of the conjecture (N. & Straßburger, CSL'22)

There is some formula A such that $\mathrm{BV} \nvdash A$ *but* $\mathrm{PL} \vdash A$.

$$A = ((a \lhd b) \wedge (c \lhd d)) \vee ((e \lhd f) \wedge (g \lhd h)) \vee (a^\perp \lhd h^\perp) \vee (e^\perp \lhd b^\perp) \vee (g^\perp \lhd d^\perp) \vee (c^\perp \lhd f^\perp)$$

Conclusion

MLL(+*Mix*) *proof nets:* a graphical syntax for proofs

(not literally) "linear λ-terms without a distinguished spanning tree"

The question of distinguishing *correct* proof nets leads to rich combinatorics,

closely related to classical topics (perfect matchings, blossoms)

Conclusion

MLL(+*Mix*) *proof nets:* a graphical syntax for proofs
(not literally) "linear λ-terms without a distinguished spanning tree"

The question of distinguishing *correct* proof nets leads to rich combinatorics,
closely related to classical topics (perfect matchings, blossoms)

We turned an obscure result on proof nets into a nice theorem on graphs

Conversely, by leveraging the literature on graphs, we got a few surprises

- including a refutation of a conjecture in proof theory
- another thing: if MLL+Mix correctness were as easy as for MLL (NL-complete) then
it would solve a conjecture on matchings by Lovász from the 1990s

Conclusion

MLL(+*Mix*) *proof nets:* a graphical syntax for proofs
(not literally) "linear λ-terms without a distinguished spanning tree"

The question of distinguishing *correct* proof nets leads to rich combinatorics,
closely related to classical topics (perfect matchings, blossoms)

We turned an obscure result on proof nets into a nice theorem on graphs

Conversely, by leveraging the literature on graphs, we got a few surprises

- including a refutation of a conjecture in proof theory
- another thing: if MLL+Mix correctness were as easy as for MLL (NL-complete) then it would solve a conjecture on matchings by Lovász from the 1990s

Thanks for your attention!