Unique perfect matchings and proof nets

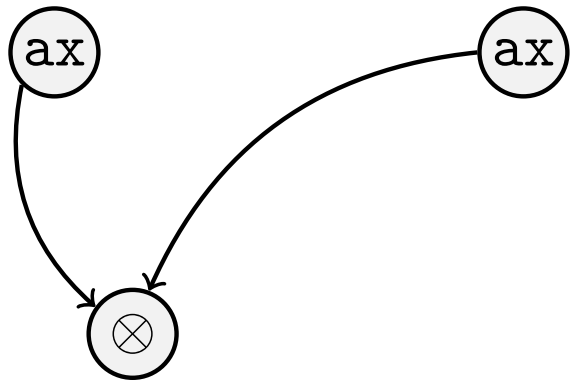
Nguyễn Lê Thành Dũng
École normale supérieure de Paris & LIPN, Université Paris 13
nltd@nguyentito.eu

Formal Structures in Computation and Deduction
(FSCD 2018, FLoC conference)
Oxford, July 9th, 2018

MLL proof nets

- We work in Multiplicative Linear Logic (MLL)
- A *proof net* is a sort of graph made of ax, $\parr$ and $\otimes$ links which represents a proof
  - i.e. translated from a sequent calculus proof
  - Equivalently, set of proof nets inductively generated

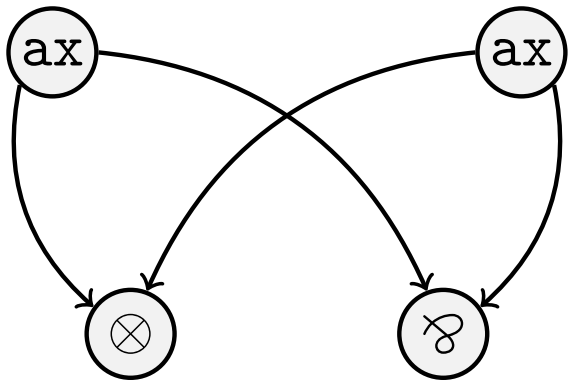
$$\dfrac{\dfrac{}{\vdash A, A^\perp}\,\mathtt{ax} \qquad \dfrac{}{\vdash B, B^\perp}\,\mathtt{ax}}{\vdash A \otimes B, A^\perp, B^\perp}\,\otimes$$



MLL proof nets

- We work in Multiplicative Linear Logic (MLL)
- A *proof net* is a sort of graph made of ax, $\parr$ and $\otimes$ links which represents a proof
  - i.e. translated from a sequent calculus proof
  - Equivalently, set of proof nets inductively generated

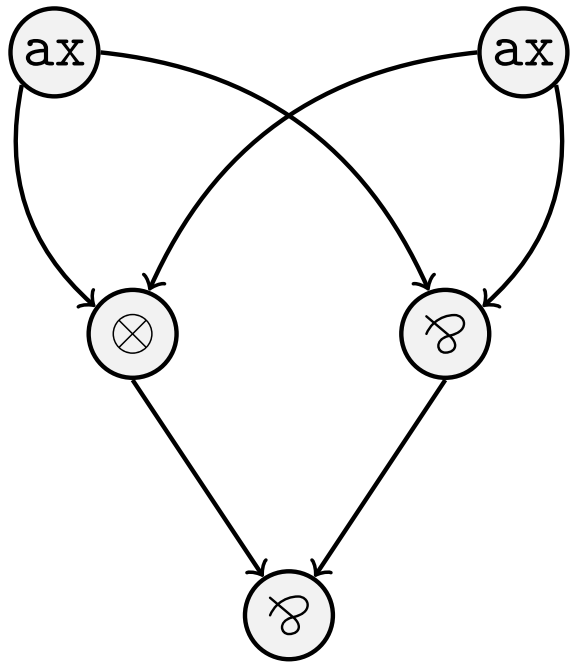
$$\dfrac{\dfrac{\dfrac{}{\vdash A, A^\perp}\,\mathtt{ax} \qquad \dfrac{}{\vdash B, B^\perp}\,\mathtt{ax}}{\vdash A \otimes B, A^\perp, B^\perp}\,\otimes}{\vdash A \otimes B, A^\perp \parr B^\perp}\,\parr$$



MLL proof nets

- We work in Multiplicative Linear Logic (MLL)
- A *proof net* is a sort of graph made of ax, $\parr$ and $\otimes$ links which represents a proof
  - i.e. translated from a sequent calculus proof
  - Equivalently, set of proof nets inductively generated

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash A, A^\perp}\,\mathtt{ax} \qquad \dfrac{}{\vdash B, B^\perp}\,\mathtt{ax}}{\vdash A \otimes B, A^\perp, B^\perp}\otimes}{\vdash A \otimes B, A^\perp \parr B^\perp}\parr}{\vdash (A \otimes B) \parr (A^\perp \parr B^\perp)}\parr$$



Proof nets vs proof structures

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

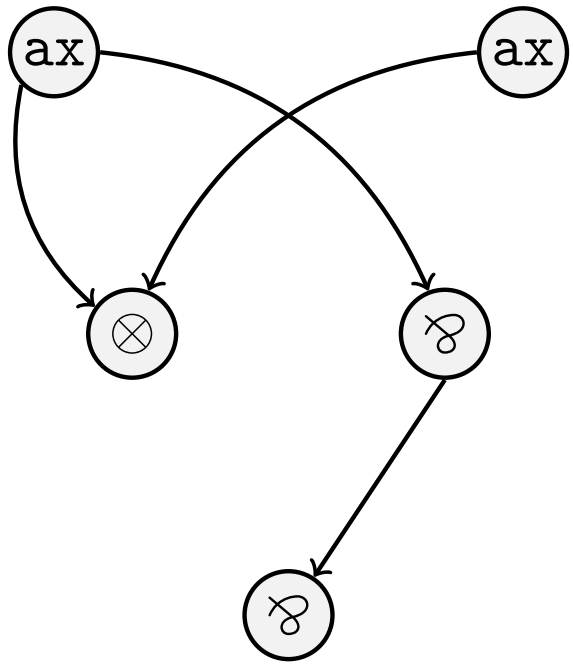
Problem (Correctness)

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

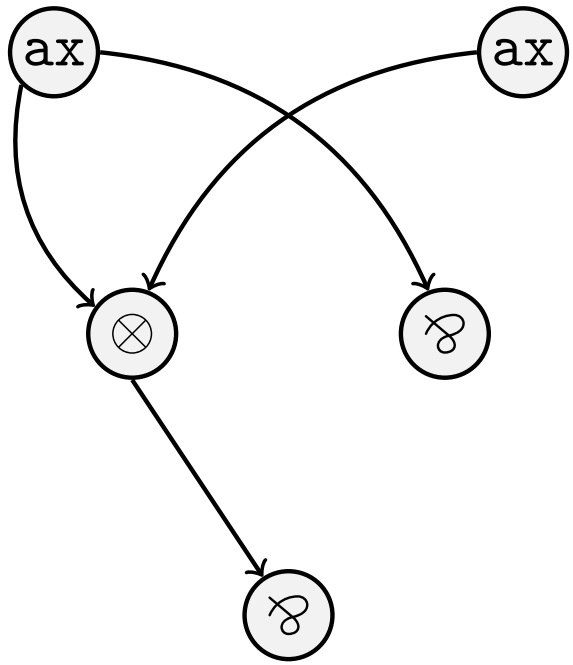
A correctness criterion for MLL

- Most common criterion: Danos–Regnier
- Delete 1 of the 2 premises of each $\parr$-link; do you always get an (undirected) *tree*?
- If so, then you've got an MLL proof net



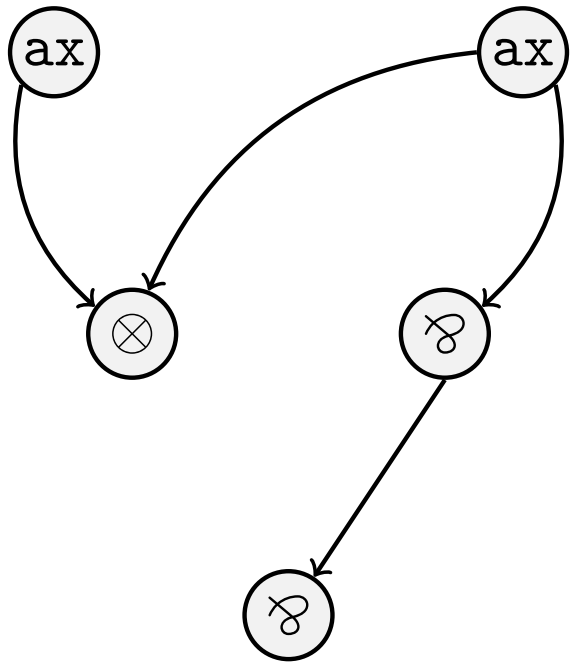
A correctness criterion for MLL

- Most common criterion: Danos–Regnier
- Delete 1 of the 2 premises of each $\parr$-link; do you always get an (undirected) *tree*?
- If so, then you've got an MLL proof net



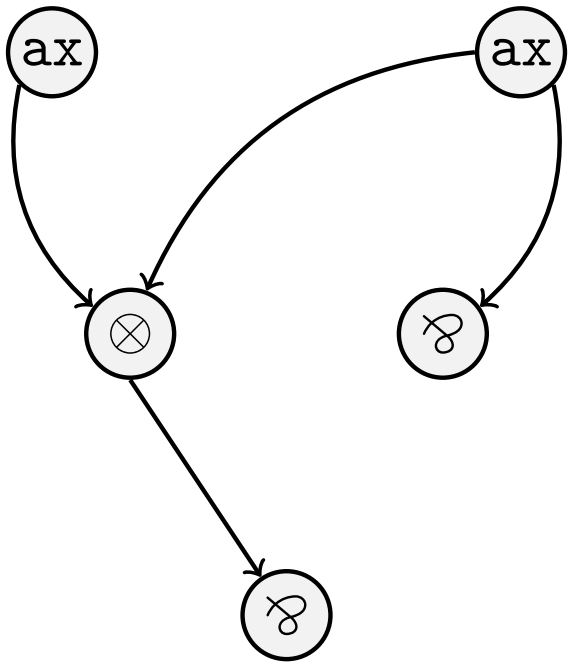
A correctness criterion for MLL

- Most common criterion: Danos–Regnier
- Delete 1 of the 2 premises of each $\parr$-link; do you always get an (undirected) *tree*?
- If so, then you've got an MLL proof net



A correctness criterion for MLL

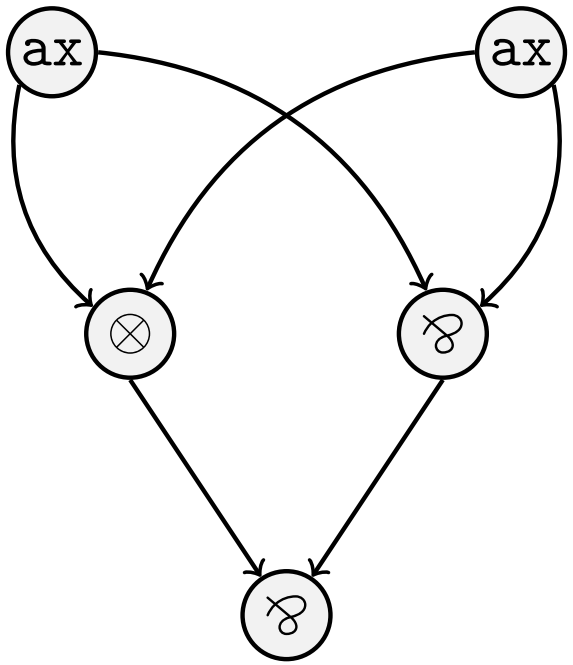
- Most common criterion: Danos–Regnier
- Delete 1 of the 2 premises of each $\parr$-link; do you always get an (undirected) *tree*?
- If so, then you've got an MLL proof net



A correctness criterion for MLL+Mix

- Most common criterion: Danos–Regnier
- Delete 1 of the 2 premises of each $\parr$-link; do you always get an (undirected) *tree* (resp. forest)?
- If so, then you've got an MLL (resp. MLL+Mix) proof net

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

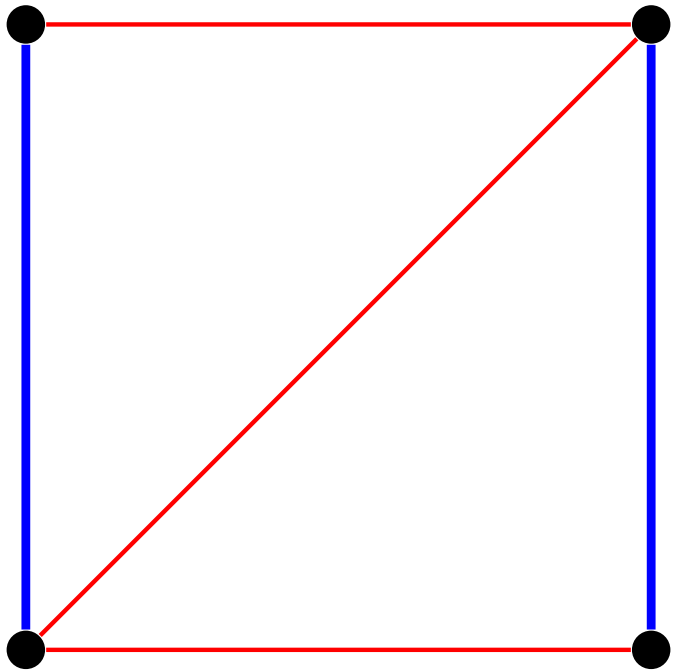


A graph-theoretic viewpoint

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

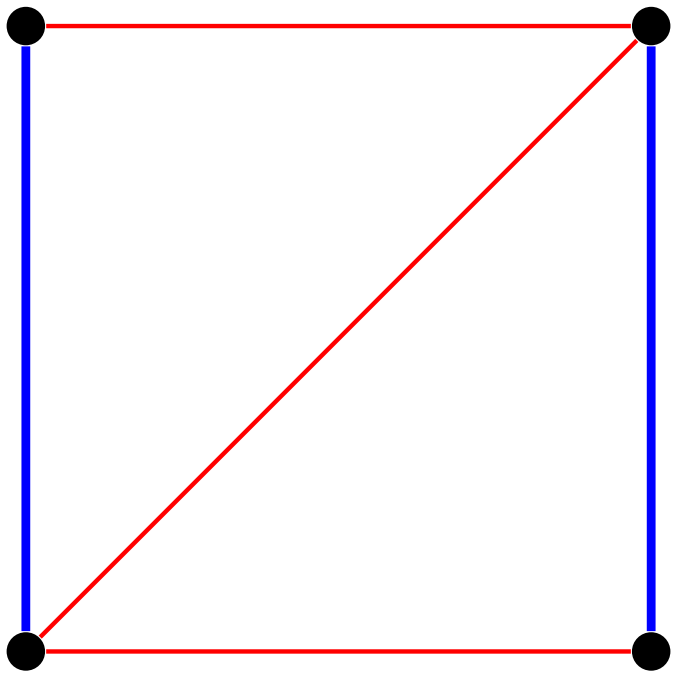
Perfect matchings (1)

- 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



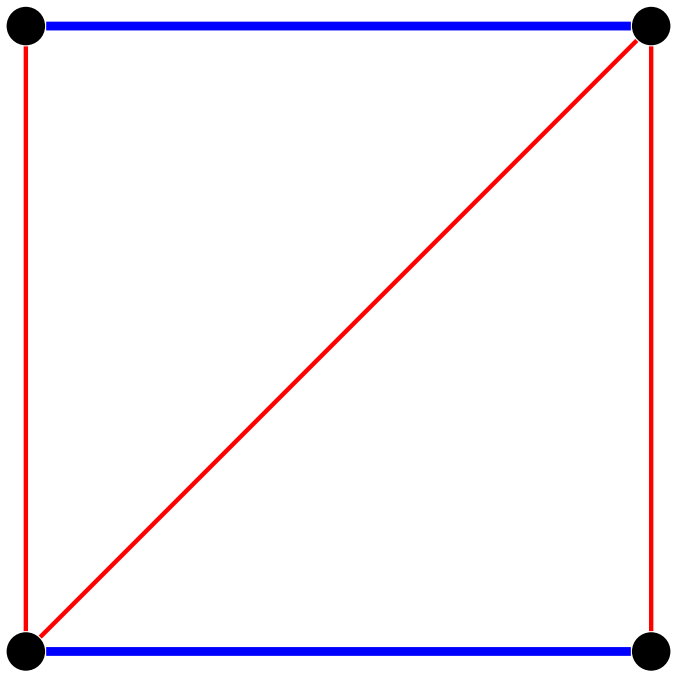
Perfect matchings (2)

- An *alternating path* is a path
  - without vertex repetitions
  - which alternates between edges inside and outside the matching
- Analogous notion of *alternating cycle*
- $\exists$ alternating cycle $\Leftrightarrow$ 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*
- $\exists$ alternating cycle $\Leftrightarrow$ the perfect matching is not *unique*



Proof net correctness vs perfect matching uniqueness

- Alternating paths / cycles in perfect matchings are equivalent to many[1] kinds of constrained paths / cycles in graph theory
- Is it also the case for MLL+Mix correctness?

[1] See e.g. Szeider, On theorems equivalent with Kotzig's result on graphs with unique 1-factors, 2004

Proof net correctness vs perfect matching uniqueness

- Alternating paths / cycles in perfect matchings are equivalent to many[1] kinds of constrained paths / cycles in graph theory
- Is it also the case for MLL+Mix correctness? YES
- A connection was found by Christian Retoré in the 90's
- *R&B-graphs*: reduction
  {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 alt. cycle).

[1] See e.g. Szeider, On theorems equivalent with Kotzig's result on graphs with unique 1-factors, 2004

An immediate application: complexity of correctness

- Correctness of MLL proof nets can be decided in linear time
- What about MLL+Mix? Not an easy extension of the MLL case
  - Combinatorial and complexity-theoretic arguments

An immediate application: complexity of correctness

- Correctness of MLL proof nets can be decided in linear time
- What about MLL+Mix? Not an easy extension of the MLL case
  - Combinatorial and complexity-theoretic arguments

Theorem (new!)

MLL+Mix correctness can be decided in linear time.

Proof.

Compute the R&B-graph, then test if it admits an alternating cycle. Both are in linear time. □

- Sophisticated algorithm for finding an alt. cycle in linear time
  - Leverage the work of graph theorists as a black box
  - Also works for MLL, less work for us than previous $O(n)$ criteria
- Everything needed already existed in 1999

On sequentialization theorems

- *Sequentialization theorem*: correct proof structures are proof nets, i.e. come from sequent calculus proofs
- A remark by Retoré: analogously, unique perfect matchings admit an inductive characterization (proof: use the theorem below)

Theorem (Kotzig 1959)

Every unique perfect matching (i.e. w/o alt cycle) 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: analogously, unique perfect matchings admit an inductive characterization (proof: use the theorem below)

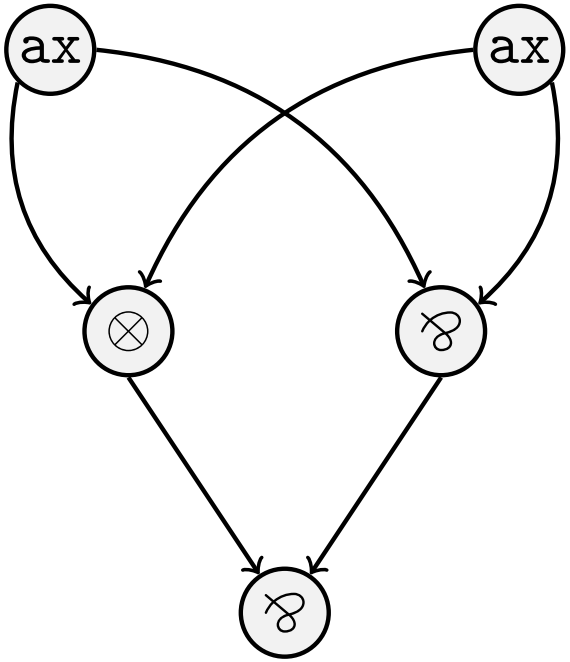
Theorem (Kotzig 1959)

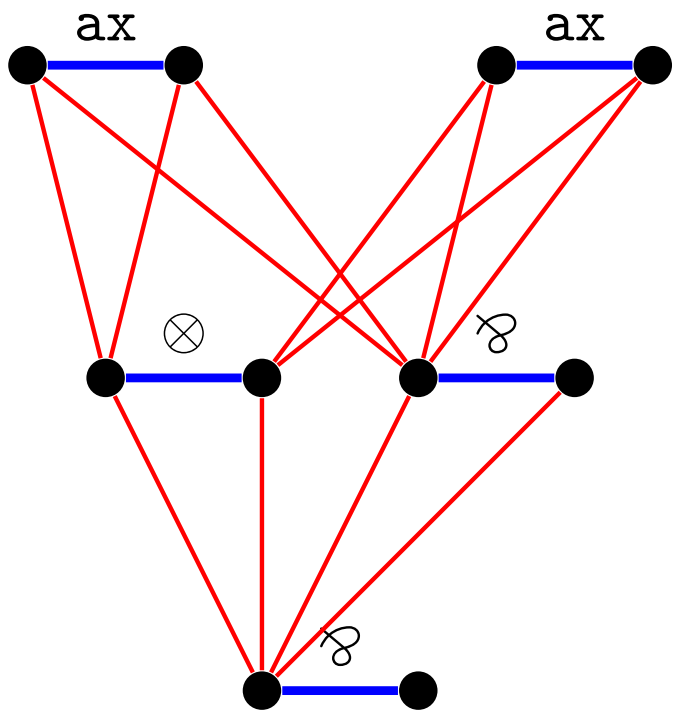
Every unique perfect matching (i.e. w/o alt cycle) contains a bridge.

- A mismatch:
  {sequentializations of a proof net} $\not\cong$ {seqs of its R&B-graph}
- We fix this with another reduction
  {proof structures} $\to$ {graphs w/ PMs}: *graphification*

Graphification of proof structures (1)

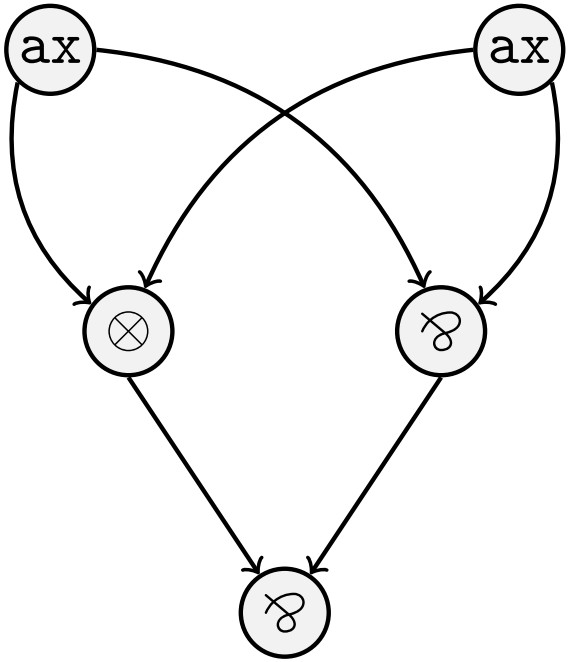
- Matching edges correspond to links
- *Bridges* correspond to *splitting terminal links*

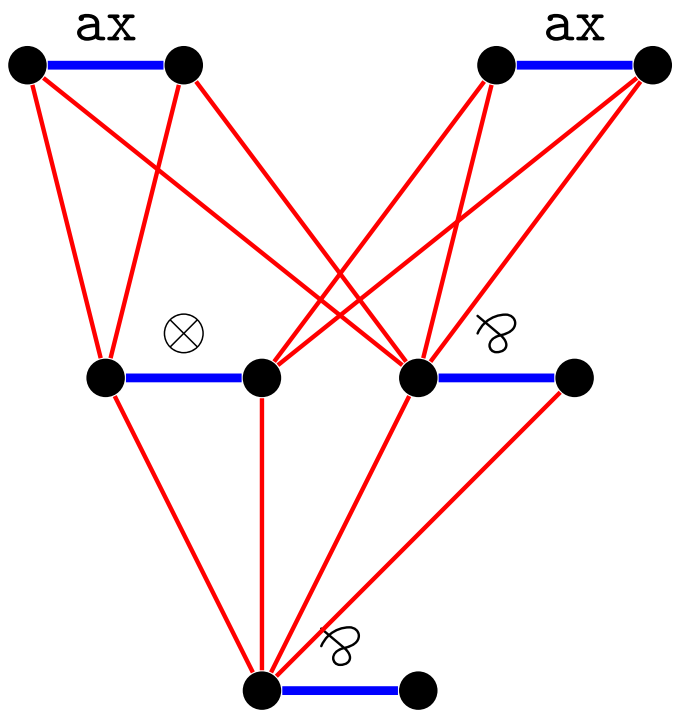




Graphification of proof structures (1)

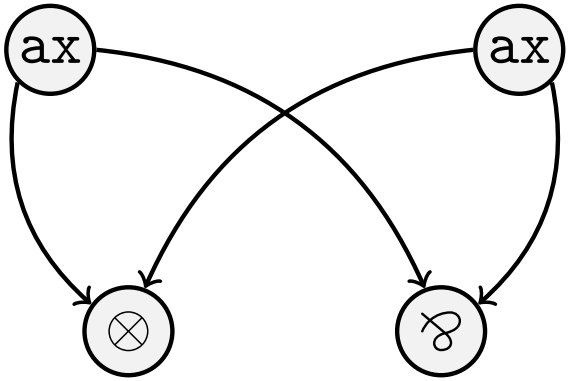
- Matching edges correspond to links
- *Bridges* correspond to *splitting terminal links*

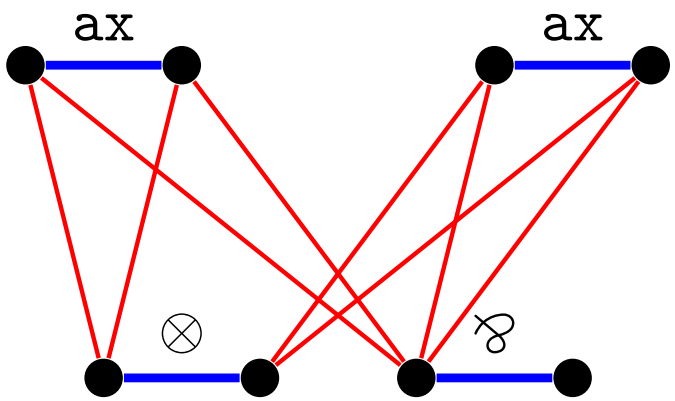




Graphification of proof structures (1)

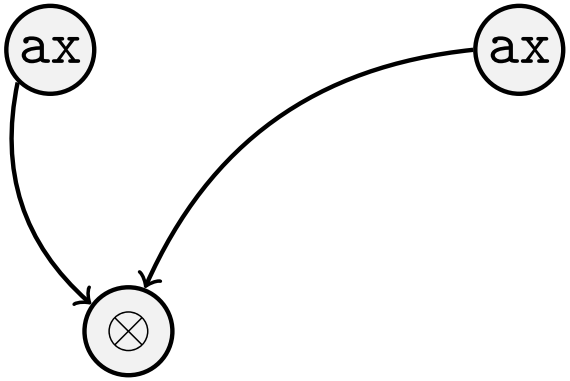
- Matching edges correspond to links
- *Bridges* correspond to *splitting terminal links*

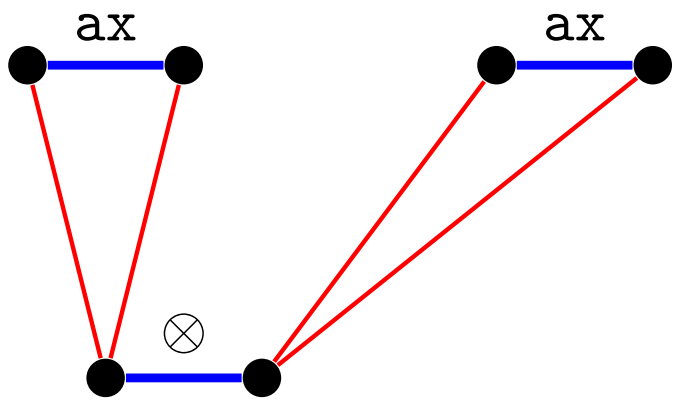




Graphification of proof structures (1)

- Matching edges correspond to links
- *Bridges* correspond to *splitting terminal links*



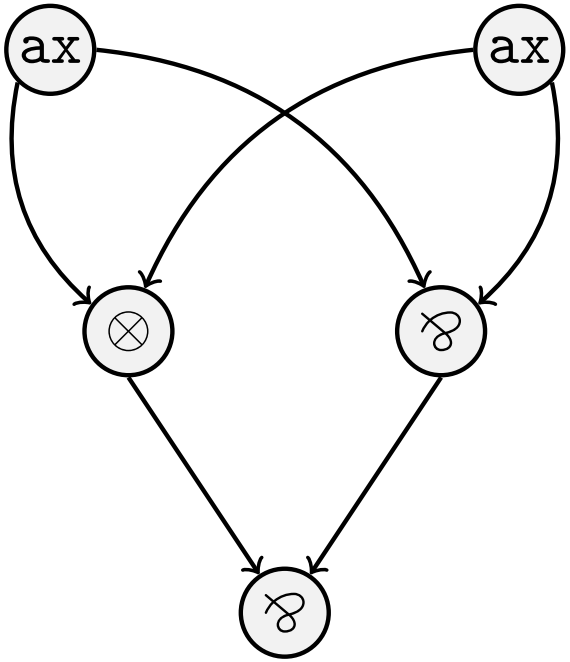


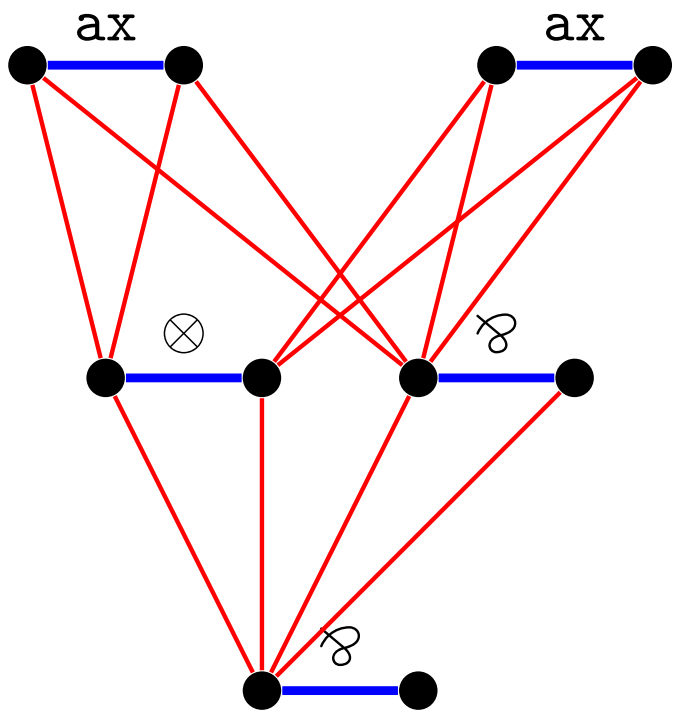
Graphification of proof structures (1)

- Matching edges correspond to links
- *Bridges* correspond to *splitting terminal links*

Graphification of proof structures (1)

- Matching edges correspond to links
- *Bridges* correspond to *splitting terminal links*





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

Graphifications of proof nets (2)

Theorem

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

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

Corollary (Sequentialization theorem for MLL+Mix)

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

- New proof, immediate from graph-theoretic analogue

Graphifications of proof nets (2)

Theorem

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

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

Corollary (Sequentialization theorem for MLL+Mix)

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

- New proof, immediate from graph-theoretic analogue
- Next, let's *compute* a sequentialization
  - Proof nets don't record their order of construction

Problem (Sequentialization)

Given a MLL+Mix proof net π, find a sequent proof which translates into π.

The naive sequentialization algorithm

1. Find a splitting link
2. Remove it
3. Recurse on remaining sub-proof net(s)

- Obvious implementation: quadratic time
  - Linear-time traversal to find a splitting link at each step

The naive sequentialization algorithm

1. Find a bridge in the graphification
2. Remove it
3. Recurse on remaining sub-proof net(s)

- Obvious implementation: quadratic time
  - Linear-time traversal to find a bridge at each step

The naive sequentialization algorithm

1. Find a bridge in the graphification
2. Remove it
3. Recurse on remaining sub-proof net(s)

- Obvious implementation: quadratic time
  - Linear-time traversal to find a bridge at each step
- Efficient implementation: how to find bridges quickly?
  - This has been studied by graph theorists

Quasi-linear sequentialization

- How to find bridges quickly? Using a dedicated data structure
  - Fixed vertex set, edge insertions/deletions, queries for bridges and connected components
  - Latest improvement: SODA 2018 paper[2]
- $O((\log |V|)^2 (\log \log |V|)^2)$ amortized complexity operations

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

Quasi-linear sequentialization

- How to find bridges quickly? Using a dedicated data structure
  - Fixed vertex set, edge insertions/deletions, queries for bridges and connected components
  - Latest improvement: SODA 2018 paper[2]
- $O((\log |V|)^2 (\log \log |V|)^2)$ amortized complexity operations

Theorem

MLL+Mix proof nets can be sequentialized in $O(n(\log n)^2(\log \log n)^2)$ *time.*

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

Quasi-linear sequentialization

- How to find bridges quickly? Using a dedicated data structure
  - Fixed vertex set, edge insertions/deletions, queries for bridges and connected components
  - Latest improvement: SODA 2018 paper[2]
- $O((\log |V|)^2 (\log \log |V|)^2)$ amortized complexity operations

Theorem

MLL+Mix proof nets can be sequentialized in $O(n(\log n)^2(\log \log n)^2)$ time.

- MLL without Mix: linear-time sequentialization
  - But the method cannot be extended to MLL+Mix

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

Mix makes things harder

- MLL proof nets are well-studied
  - Linear-time correctness and sequentialization
  - Correctness is NL-complete
- Much less was known about MLL+Mix
- Recap of new results:
  correctness in linear time, but sequentialization in *quasi*-linear time

Mix makes things harder

- MLL proof nets are well-studied
  - Linear-time correctness and sequentialization
  - Correctness is NL-complete
- Much less was known about MLL+Mix
- Recap of new results:
  correctness in linear time, but sequentialization in *quasi*-linear time
- MLL+Mix correctness $\in$ NL would solve an open problem in graph theory
  - Proof: reduce uniqueness of PMs to MLL+Mix correctness
  - So the problems are actually *equivalent*
- $\rightarrow$ Both MLL+Mix correctness and sequentialization seem "harder" in some informal sense

Mix makes things harder

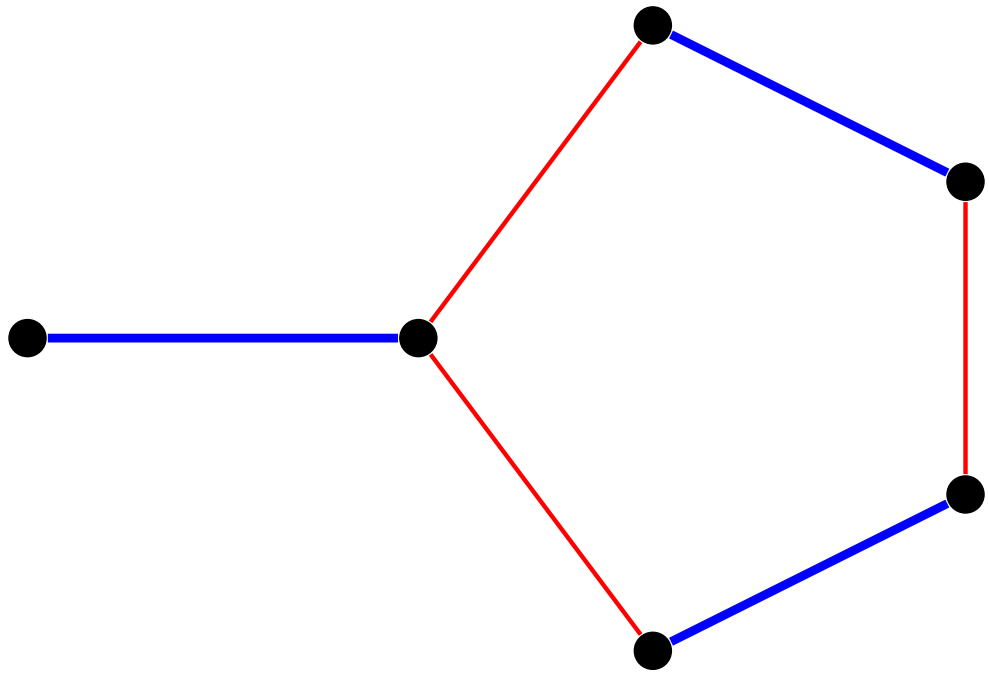
- MLL proof nets are well-studied
  - Linear-time correctness and sequentialization
  - Correctness is NL-complete
- Much less was known about MLL+Mix
- Recap of new results:
  correctness in linear time, but sequentialization in *quasi*-linear time
- MLL+Mix correctness $\in$ NL would solve an open problem in graph theory
  - Proof: reduce uniqueness of PMs to MLL+Mix correctness
  - So the problems are actually *equivalent*
- $\rightarrow$ Both MLL+Mix correctness and sequentialization seem "harder" in some informal sense
- We've applied graph algorithms to linear logic
  - Next: a theorem on graphs inspired by linear logic

Blossoms in matching theory

- A key concept in combinatorial matching algorithms, e.g. testing PM uniqueness: *blossoms*[3]

Definition

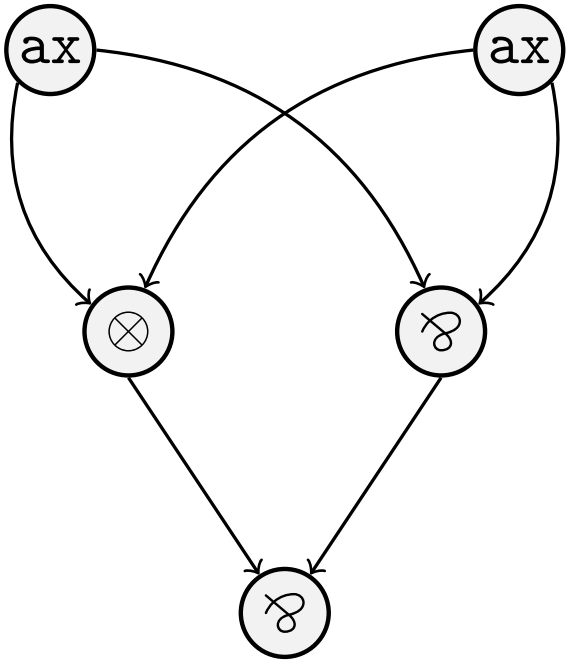
A *blossom* is a cycle with exactly one vertex matched outside the cycle.

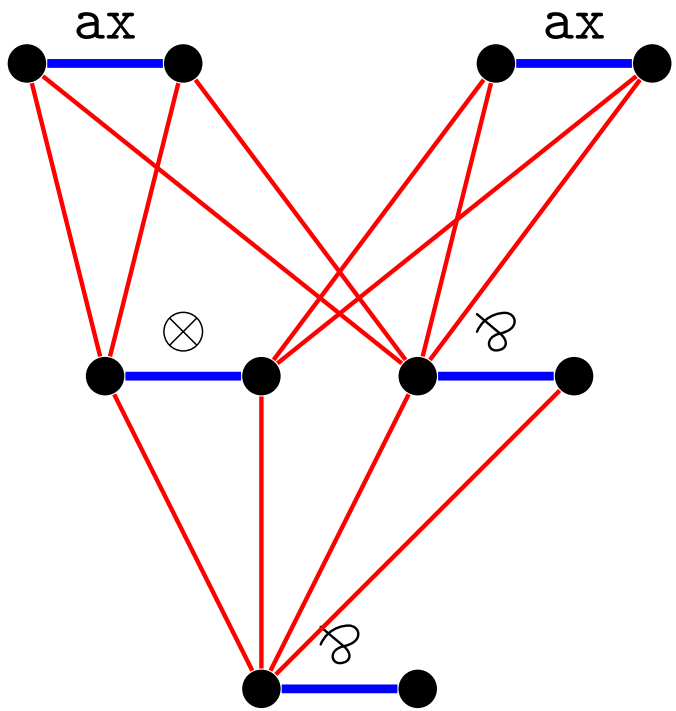


[3] Edmonds, *Paths, trees and flowers*, Canadian J. Math., 1965

Blossoms vs. dependencies

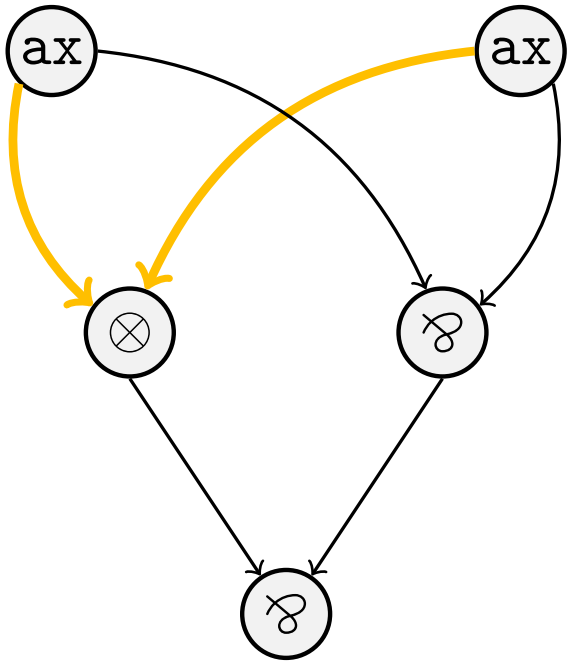
- Blossoms of graphification $\rightsquigarrow$ subformulae and dependencies

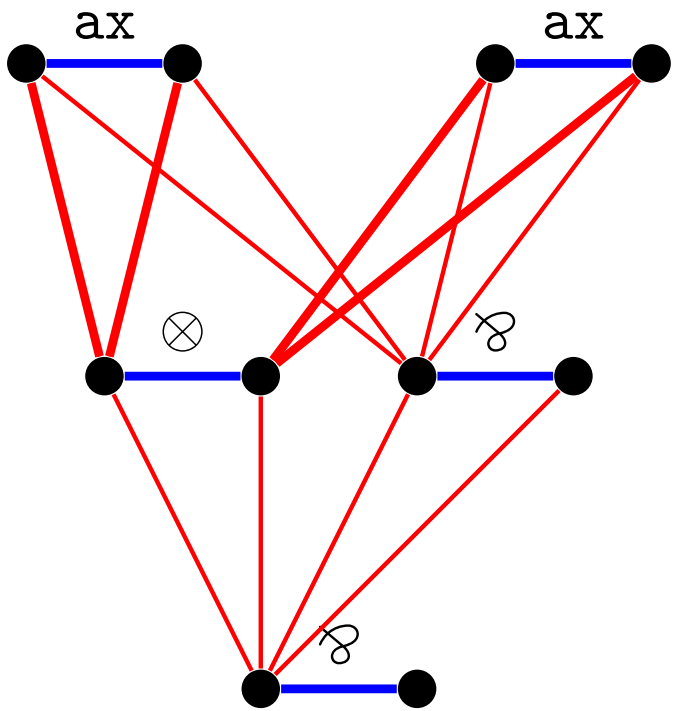




Blossoms vs. dependencies

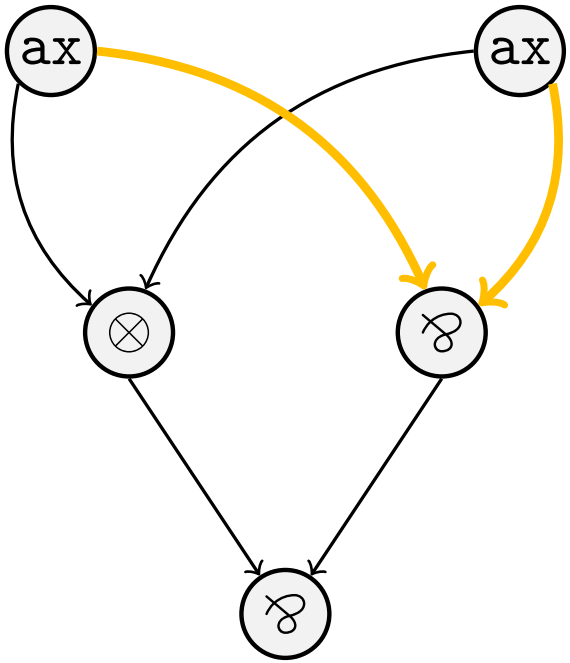
- Blossoms of graphification $\rightsquigarrow$ **subformulae** and dependencies

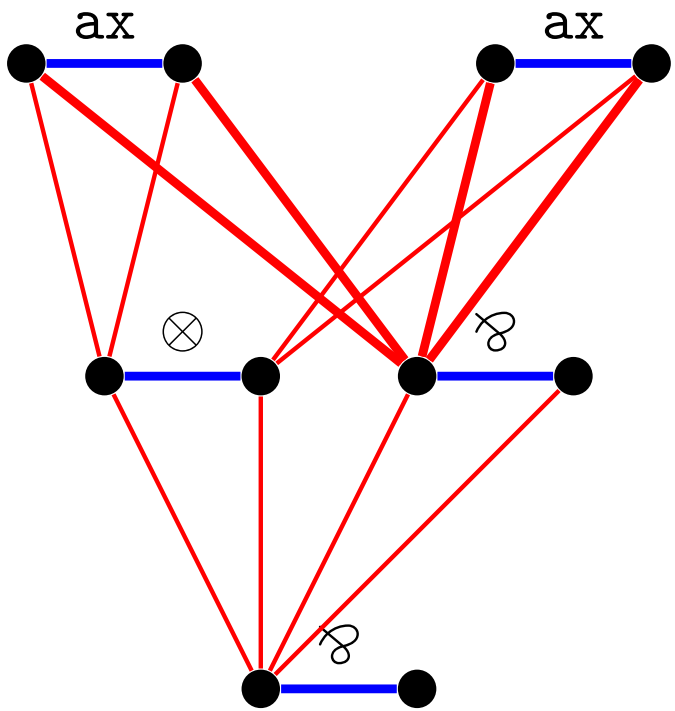




Blossoms vs. dependencies

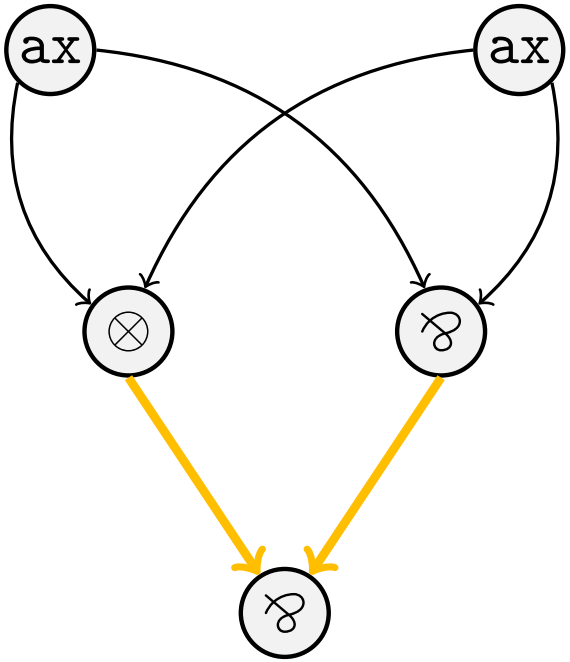
- Blossoms of graphification $\rightsquigarrow$ **subformulae** and dependencies

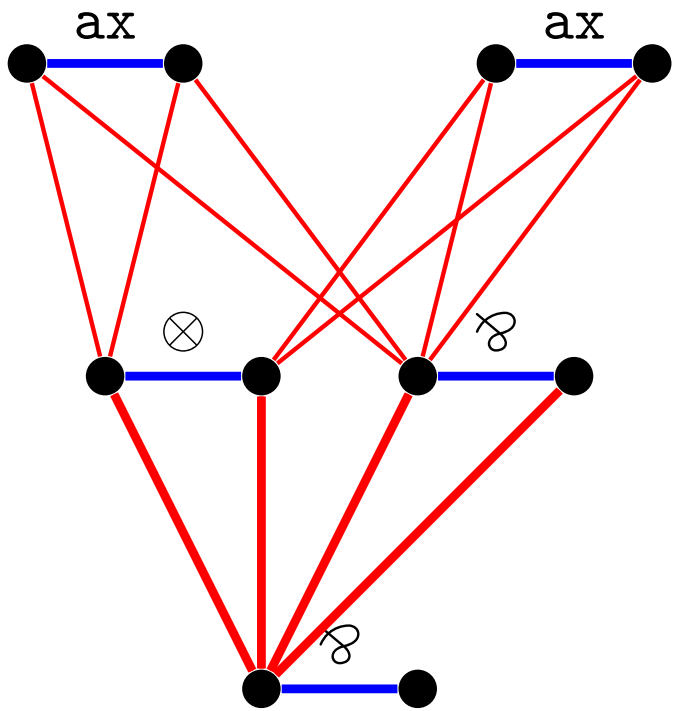




Blossoms vs. dependencies

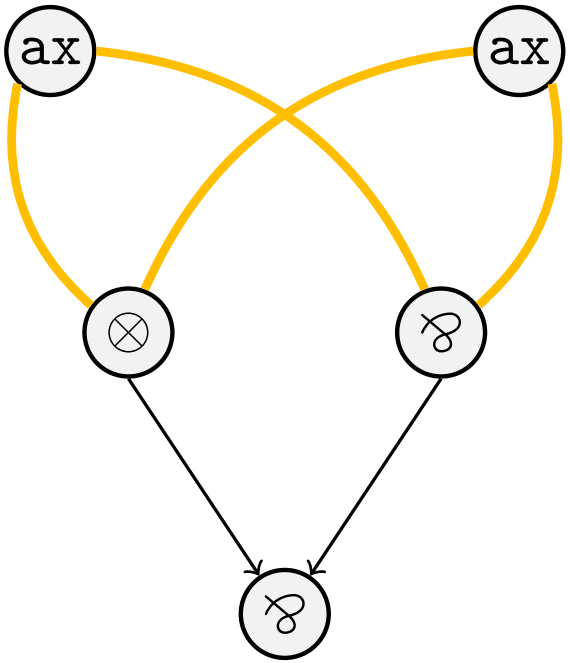
- Blossoms of graphification $\rightsquigarrow$ **subformulae** and dependencies

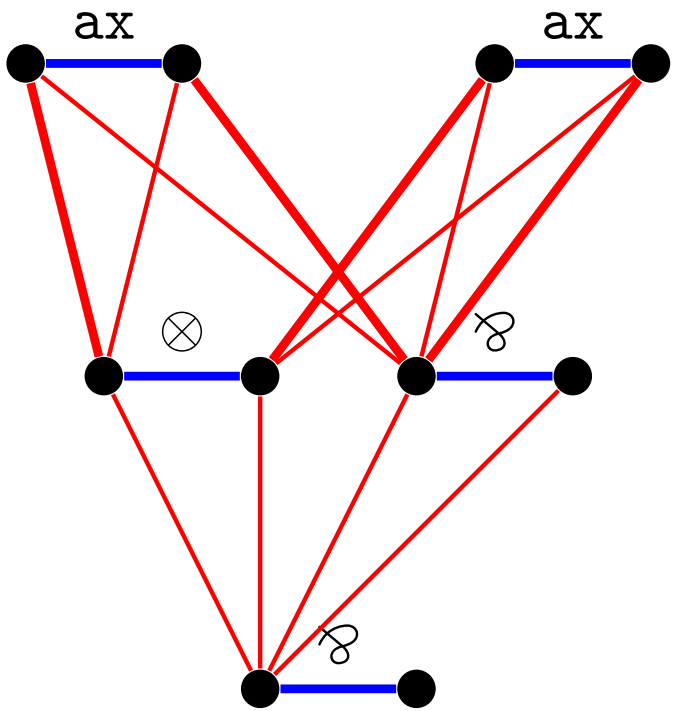




Blossoms vs. dependencies

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



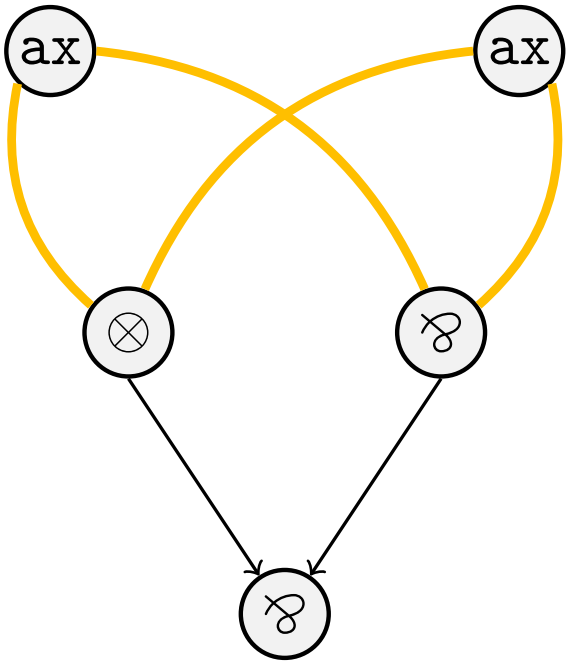


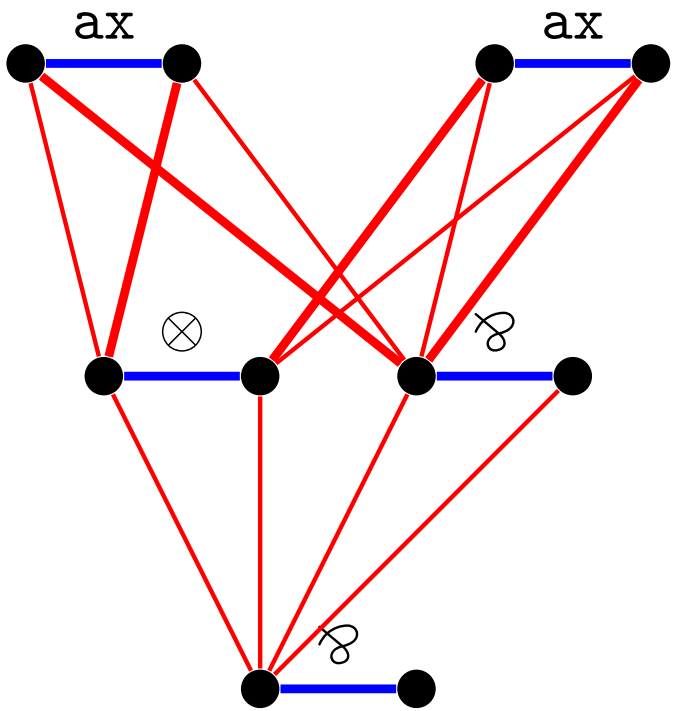
Definition

A $\parr$-link l *depends upon* a link 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**





Definition

A $\parr$-link l *depends upon* a link l' if there is a Danos–Regnier path between the premises of l going through l'.

Kingdom ordering of proof nets and unique PMs

Definition (Kingdom ordering of a proof net)

Let l, l' be links of a MLL+Mix proof net π. We define $l \ll_\pi l$ iff every sequentialization of π introduces l above l'.

Theorem (Bellin 1997)

$\ll_\pi = ((\textit{subformula relation}) \cup (\textit{dependency relation}))^*$

Kingdom ordering of proof nets and unique PMs

Definition (Kingdom ordering of a proof net)

Let l, l' be links of a MLL+Mix proof net π. We define $l \ll_\pi l$ iff every sequentialization of π introduces l above l'.

Theorem (Bellin 1997)

$\ll_\pi = ((\textit{subformula relation}) \cup (\textit{dependency relation}))^*$

- 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"

- A non-artificial graph-theoretic result coming from linear logic
  - Simpler statement: transitive closure of 1 relation instead of 2

Summary and open questions

- Unique perfect matchings: the right mainstream graph-theoretic counterpart for the statics of MLL+Mix proof nets
  - Not a combinatorial bijection, but both algorithmic reductions and transfer of structural properties
- Consequences:
  - Progress on central problems on proof nets in the MLL+Mix case
  - New results in graph theory

- What about MLL?
  - Goal: extract the combinatorial essence of MLL correctness and sequentialization, by *forgetting about logic* (no dynamics)
  - Perfect matchings are "simpler" than proof structures

Summary and open questions

- Unique perfect matchings: the right mainstream graph-theoretic counterpart for the statics of MLL+Mix proof nets
  - Not a combinatorial bijection, but both algorithmic reductions and transfer of structural properties
- Consequences:
  - Progress on central problems on proof nets in the MLL+Mix case
  - New results in graph theory

- What about MLL?
  - Goal: extract the combinatorial essence of MLL correctness and sequentialization, by *forgetting about logic* (no dynamics)
  - Perfect matchings are "simpler" than proof structures

Thank you for your attention!