A complexity gap between pomset logic and system BV

Nguyễn Lê Thành Dũng (a.k.a. Tito) — nltd@nguyentito.eu
Laboratoire d'informatique de Paris Nord, Villetaneuse, France
joint work with Lutz Straßburger (Inria Saclay)

Journée réseaux de démonstration (workshop on proof nets), 22 janvier 2021

The big picture

Reminder from Christian's talk: *pomset logic* conservatively extends MLL+Mix with '$\triangleleft$'

$$A, B ::= \mathbf{I} \mid a \mid a^{\perp} \mid A \otimes B \mid A \parr B \mid A \triangleleft B$$

non-commutativity: $A \triangleleft B \not\equiv B \triangleleft A$ self-duality: $(A \triangleleft B)^{\perp} = A^{\perp} \triangleleft B^{\perp}$ (*not* $B^{\perp} \triangleleft A^{\perp}$)

The big picture

Reminder from Christian's talk: *pomset logic* conservatively extends MLL+Mix with '$\triangleleft$'

$$A, B ::= \mathbf{I} \mid a \mid a^{\perp} \mid A \otimes B \mid A \parr B \mid A \triangleleft B$$

non-commutativity: $A \triangleleft B \not\equiv B \triangleleft A$ self-duality: $(A \triangleleft B)^{\perp} = A^{\perp} \triangleleft B^{\perp}$ (*not* $B^{\perp} \triangleleft A^{\perp}$)

System ***BV*** is another logic devised in the late 1990s with the same properties

Guglielmi 2007, *A System of Interaction and Structure* (emphasis mine):

> **It is still open whether the logic in this paper, called *BV*, is the same as pomset logic. We conjecture that it is actually the same logic**, *but one crucial step is still missing, at the time of this writing, in the equivalence proof. This paper is the first in a planned series of 3 papers dedicated to BV.* [...] *In the 3rd part, some of my colleagues will hopefully show the equivalence of BV and pomset logic,* [...]

It is known that BV provability implies pomset provability.

The big picture

Reminder from Christian's talk: *pomset logic* conservatively extends MLL+Mix with '$\triangleleft$'

$$A, B ::= \mathbf{I} \mid a \mid a^{\perp} \mid A \otimes B \mid A \parr B \mid A \triangleleft B$$

non-commutativity: $A \triangleleft B \not\equiv B \triangleleft A$ self-duality: $(A \triangleleft B)^{\perp} = A^{\perp} \triangleleft B^{\perp}$ (*not* $B^{\perp} \triangleleft A^{\perp}$)

System ***BV*** is another logic devised in the late 1990s with the same properties

Guglielmi 2007, *A System of Interaction and Structure* (emphasis mine):

> **It is still open whether the logic in this paper, called *BV*, is the same as pomset logic. We conjecture that it is actually the same logic**, *but one crucial step is still missing, at the time of this writing, in the equivalence proof. This paper is the first in a planned series of 3 papers dedicated to BV.* [...] *In the 3rd part, some of my colleagues will hopefully show the equivalence of BV and pomset logic,* [...]

It is known that BV provability implies pomset provability. **We show that the converse is false!** Explicit counterexample with 8 variables, each with 1 positive and 1 negative occurrence

The big picture

Reminder from Christian's talk: *pomset logic* conservatively extends MLL+Mix with '$\triangleleft$'

$$A, B ::= \mathbf{I} \mid a \mid a^{\perp} \mid A \otimes B \mid A \parr B \mid A \triangleleft B$$

non-commutativity: $A \triangleleft B \not\equiv B \triangleleft A$ self-duality: $(A \triangleleft B)^{\perp} = A^{\perp} \triangleleft B^{\perp}$ (*not* $B^{\perp} \triangleleft A^{\perp}$)

System ***BV*** is another logic devised in the late 1990s with the same properties

Guglielmi 2007, *A System of Interaction and Structure* (emphasis mine):

> **It is still open whether the logic in this paper, called *BV*, is the same as pomset logic. We conjecture that it is actually the same logic**, *but one crucial step is still missing, at the time of this writing, in the equivalence proof. This paper is the first in a planned series of 3 papers dedicated to BV.* [...] *In the 3rd part, some of my colleagues will hopefully show the equivalence of BV and pomset logic,* **this way explaining why it was impossible to find a sequent system for pomset logic.**

It is known that BV provability implies pomset provability. **We show that the converse is false!** Explicit counterexample with 8 variables, each with 1 positive and 1 negative occurrence

Proof nets vs sequent calculus vs deep inference

Pomset logic is defined through *proof nets* with a "geometric" correctness criterion
No complete *deductive* proof system known until Slavnov's sequent calculus (2019)
(deductive = inference rules + derivation trees)

Proof nets vs sequent calculus vs deep inference

Pomset logic is defined through *proof nets* with a "geometric" correctness criterion
No complete *deductive* proof system known until Slavnov's sequent calculus (2019)
(deductive = inference rules + derivation trees)

BV is defined using *deep inference* (invented for this purpose); no sequent calculus known
So if pomset logic and BV had been equivalent – *we show that they aren't!* – this would have:

- provided a deductive system for pomset logic (deep inference is deductive);
- explained the common difficulty.

Proof nets vs sequent calculus vs deep inference

Pomset logic is defined through *proof nets* with a "geometric" correctness criterion
No complete *deductive* proof system known until Slavnov's sequent calculus (2019)
(deductive = inference rules + derivation trees)

BV is defined using *deep inference* (invented for this purpose); no sequent calculus known
So if pomset logic and BV had been equivalent – *we show that they aren't!* – this would have:

- provided a deductive system for pomset logic (deep inference is deductive);
- explained the common difficulty.

> *This paper* [that is, Guglielmi 2007] *is the first in a planned series of 3 papers dedicated to BV.* [...]
> *In the 2nd part, Alwen Tiu will show why BV cannot be defined in any sequent system.*

There is indeed an obstruction that rules out traditional sequent calculi and "shallow systems"
more generally (*A System of Interaction and Structure II: The Need for Deep Inference*, 2006)
but it doesn't apply to e.g. Slavnov's "decorated sequents"

A glance at deep inference + complexity of provability

Deep inference = unary rules applied to subformulas of arbitrary depth:

$$\text{inference rule } \frac{A}{B} \qquad \rightsquigarrow \qquad \text{instances } \frac{S[A]}{S[B]} \text{ for any context } S$$

(compare with rewriting systems, or functoriality in categorical logic)

$$\text{e.g.} \qquad \frac{A \parr (B \otimes (C \parr D))}{A \parr ((B \otimes C) \parr D)}$$

A glance at deep inference + complexity of provability

Deep inference = unary rules applied to subformulas of arbitrary depth:

$$\text{inference rule } \frac{A}{B} \qquad \rightsquigarrow \qquad \text{instances } \frac{S[A]}{S[B]} \text{ for any context } S$$

(compare with rewriting systems, or functoriality in categorical logic)

$$\text{e.g.} \qquad \frac{\dfrac{(A \parr B) \otimes (C \parr D)}{A \parr (B \otimes (C \parr D))}}{A \parr ((B \otimes C) \parr D)} \qquad \Longleftrightarrow \qquad \frac{\vdash A, B \qquad \vdash C, D}{\vdash A, B \otimes C, D}$$

A glance at deep inference + complexity of provability

Deep inference = unary rules applied to subformulas of arbitrary depth:

$$\text{inference rule } \frac{A}{B} \qquad \rightsquigarrow \qquad \text{instances } \frac{S[A]}{S[B]} \text{ for any context } S$$

(compare with rewriting systems, or functoriality in categorical logic)

$$\text{e.g.} \qquad \frac{\dfrac{(A \parr B) \otimes (C \parr D)}{A \parr (B \otimes (C \parr D))}}{A \parr ((B \otimes C) \parr D)} \qquad \Longleftrightarrow \qquad \frac{\vdash A, B \qquad \vdash C, D}{\vdash A, B \otimes C, D}$$

I won't list the rules of system BV (they're related to the cograph rewriting in Christian's talk)

Important: *proofs of A are of length $O(|A|^2)$*, therefore **provability in BV is a NP problem**

(actually NP-complete [Kahramanoğulları 2008])

A glance at deep inference + complexity of provability

Deep inference = unary rules applied to subformulas of arbitrary depth:

$$\text{inference rule } \frac{A}{B} \qquad \rightsquigarrow \qquad \text{instances } \frac{S[A]}{S[B]} \text{ for any context } S$$

(compare with rewriting systems, or functoriality in categorical logic)

$$\text{e.g.} \quad \frac{\dfrac{(A \parr B) \otimes (C \parr D)}{A \parr (B \otimes (C \parr D))}}{A \parr ((B \otimes C) \parr D)} \qquad \Longleftrightarrow \qquad \frac{\vdash A, B \qquad \vdash C, D}{\vdash A, B \otimes C, D}$$

I won't list the rules of system BV (they're related to the cograph rewriting in Christian's talk)

Important: *proofs of A are of length $O(|A|^2)$*, therefore **provability in BV is a NP problem**

(actually NP-complete [Kahramanoğulları 2008])

Theorem (the actual main topic of the talk)

Provability in pomset logic is Σ_2^{p}-complete.

(2nd level of the polynomial hierarchy, *strictly harder than* NP unless NP = coNP)

Complexity of proof net correctness

Theorem

Provability in pomset logic is Σ_2^{p}-complete.

Along the way, we proved a simpler result with the same tools:

Theorem

Proof net correctness (given a proof structure, is it a correct proof net?) in pomset logic is coNP-complete.

The rest of the talk proves this!
I will assume that you know about MLL+Mix proof nets

Important ingredient: *perfect matchings*, generalized to *directed* graphs
+ inspiration from the literature on *edge-colored graphs*
(Gourvès et al. 2013, *Complexity of trails, paths and circuits in arc-colored digraphs*)

Perfect matchings

Definition

A *perfect matching* is a set of edges in a graph such that each vertex is incident to exactly one edge in the matching.

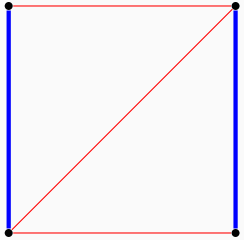
A classical topic in combinatorics!

Definition

An *æ-cycle* is a cycle which

- *alternates* between edges inside and outside the matching
- is *elementary*: has no vertex repetitions

Remark: $\exists$ æ-cycle $\Leftrightarrow$ the perfect matching is not *unique*
(Often just called "alternating cycles", elementary by default)



Perfect matchings

Definition

A *perfect matching* is a set of edges in a graph such that each vertex is incident to exactly one edge in the matching.

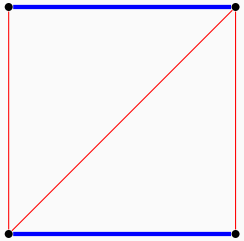
A classical topic in combinatorics!

Definition

An *æ-cycle* is a cycle which

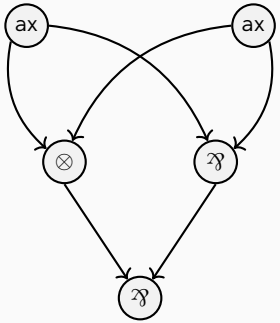
- *alternates* between edges inside and outside the matching
- is *elementary*: has no vertex repetitions

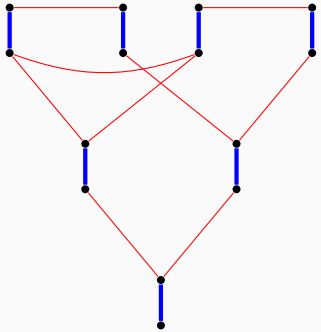
Remark: $\exists$ æ-cycle $\Leftrightarrow$ the perfect matching is not *unique*
(Often just called "alternating cycles", elementary by default)



Perfect matchings for MLL+Mix proof nets

An old translation proof structures → graphs with perfect matchings by Christian Retoré
(*beware*: not to be confused with cographs & chordless æ-cycles!)

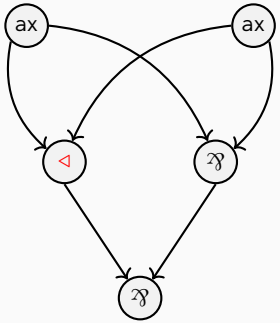


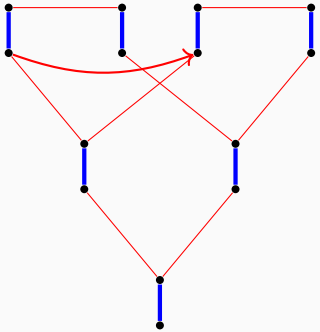


correctness = no Danos–Regnier switching cycle ⟺ no æ-cycle

Perfect matchings for MLL+Mix proof nets

An old translation proof structures → graphs with perfect matchings by Christian Retoré (*beware*: not to be confused with cographs & chordless æ-cycles!)



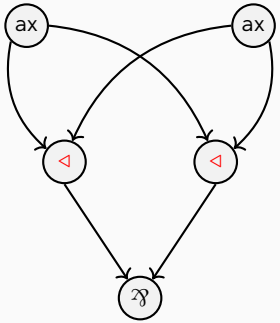


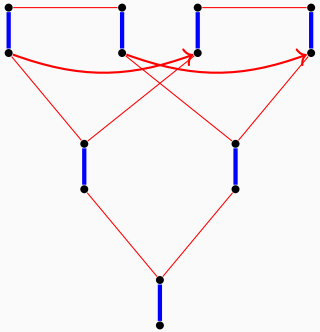
correctness = no Danos–Regnier switching cycle ⟺ no æ-cycle

Pomset logic: add *directed* edges to handle non-commutativity

Perfect matchings for MLL+Mix proof nets

An old translation proof structures → graphs with perfect matchings by Christian Retoré
(*beware*: not to be confused with cographs & chordless æ-cycles!)



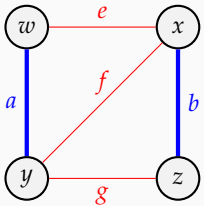


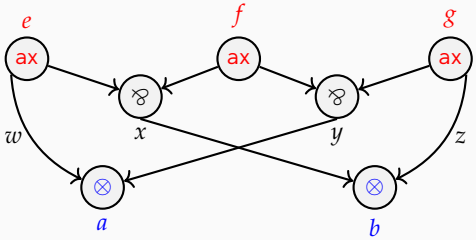
correctness = no Danos–Regnier switching cycle ⟺ no æ-cycle

Pomset logic: add *directed* edges to handle non-commutativity

Reduction perfect matchings → proof structures

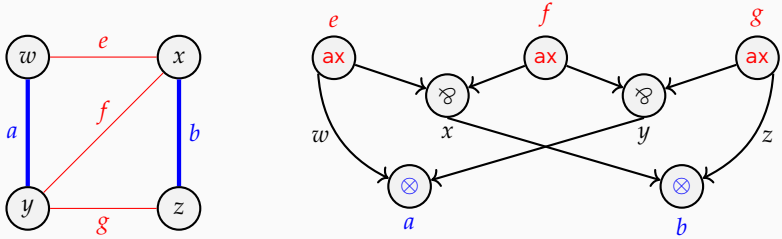
The converse direction (from my paper *Unique perfect matchings and proof nets*, 2018):





Reduction perfect matchings → proof structures

The converse direction (from my paper *Unique perfect matchings and proof nets*, 2018):



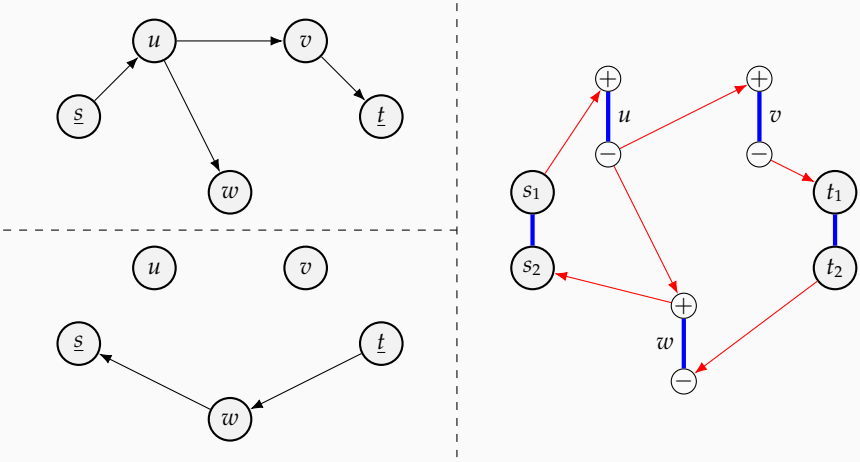
Extends to perfect matchings in general *directed* graphs → *pomset logic* proof structures, using the non-commutative ◁ to build a "directed ax" gadget. We therefore have:

Lemma

There is a polynomial-time reduction from ∃ of directed æ-cycles to pomset logic proof net incorrectness.

Goal: show that finding directed æ-cycles is NP-hard

Reduction from another graph-theoretic problem ("elementary round-trip")



æ-cycle on the right $\iff$ go from s to t in top graph, then back from t to s in bottom graph, *without visiting the same vertex twice* (elementary $\in$ æ)

Reduction CNF-SAT $\rightarrow$ elementary round-trip

Consider the SAT instance

$$(x \vee y \vee z) \wedge (\neg x \vee y) \wedge (\neg y \vee \neg z)$$

We consider two graphs whose vertices are the literal occurrences in the clauses:

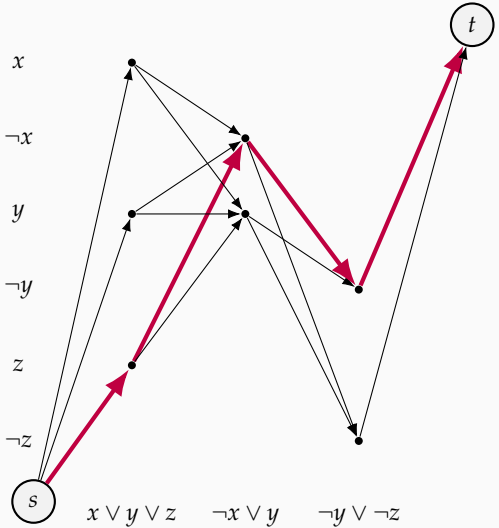
Reduction CNF-SAT $\rightarrow$ elementary round-trip

Consider the SAT instance

$$(x \vee y \vee z) \wedge (\neg x \vee y) \wedge (\neg y \vee \neg z)$$

We consider two graphs whose vertices are the literal occurrences in the clauses:

- paths $s \rightarrow t$ in 1st graph =
  choose one literal in each clause



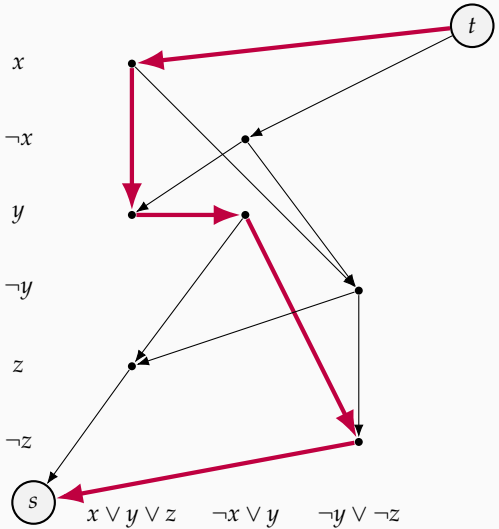
Reduction CNF-SAT $\rightarrow$ elementary round-trip

Consider the SAT instance

$$(x \vee y \vee z) \wedge (\neg x \vee y) \wedge (\neg y \vee \neg z)$$

We consider two graphs whose vertices are the literal occurrences in the clauses:

- paths $s \rightarrow t$ in 1st graph =
  choose one literal in each clause
- paths $t \rightarrow s$ in 2nd graph =
  choose assignment and visit all `false` literals
  (here $x =$ `false`, $y =$ `false`, $z =$ `true`)



Reduction CNF-SAT $\to$ elementary round-trip

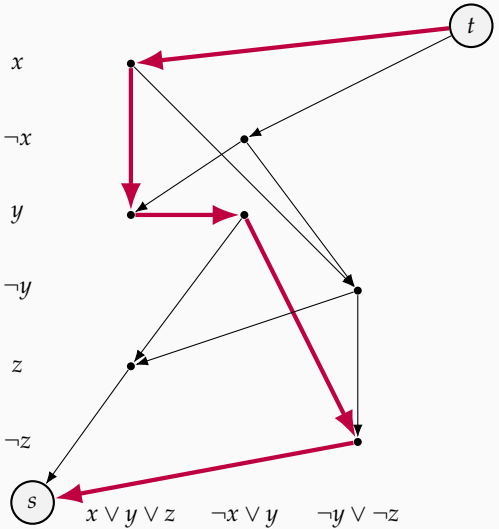
Consider the SAT instance

$$(x \vee y \vee z) \wedge (\neg x \vee y) \wedge (\neg y \vee \neg z)$$

We consider two graphs whose vertices are the literal occurrences in the clauses:

- paths $s \to t$ in 1st graph =
  choose one literal in each clause
- paths $t \to s$ in 2nd graph =
  choose assignment and visit all `false` literals
  (here $x =$ `false`, $y =$ `false`, $z =$ `true`)

Non-intersecting pair (cf. previous slide)
= satisfying assignment



Conclusion

Provability is a strictly harder problem for pomset logic than for BV unless NP = coNP:

- it is NP-complete for system BV [Kahramanoğulları 2008]
- and Σ_2^{P}-complete for pomset logic (new!)
  - related result: proof net correctness is coNP-complete in pomset logic (also new!)

Conclusion

Provability is a strictly harder problem for pomset logic than for BV unless NP = coNP:

- it is NP-complete for system BV [Kahramanoğulları 2008]
- and Σ_2^{p}-complete for pomset logic (new!)
  - related result: proof net correctness is coNP-complete in pomset logic (also new!)

Unconditional refutation of the conjectured equivalence of pomset logic and BV:

$$A = ((a \triangleleft b) \otimes (c \triangleleft d)) \parr ((e \triangleleft f) \otimes (g \triangleleft h)) \parr (a^{\perp} \triangleleft h^{\perp}) \parr (e^{\perp} \triangleleft b^{\perp}) \parr (g^{\perp} \triangleleft d^{\perp}) \parr (c^{\perp} \triangleleft f^{\perp})$$

- A is provable in pomset logic: the only possible proof structure is a correct proof net
- for any valid BV inference $\dfrac{B}{A}$, the premise B is not provable in pomset logic (*a fortiori* in BV)

Conclusion

Provability is a strictly harder problem for pomset logic than for BV unless NP = coNP:

- it is NP-complete for system BV [Kahramanoğulları 2008]
- and Σ_2^{p}-complete for pomset logic (new!)
  - related result: proof net correctness is coNP-complete in pomset logic (also new!)

Unconditional refutation of the conjectured equivalence of pomset logic and BV:

$$A = ((a \triangleleft b) \otimes (c \triangleleft d)) \parr ((e \triangleleft f) \otimes (g \triangleleft h)) \parr (a^{\perp} \triangleleft h^{\perp}) \parr (e^{\perp} \triangleleft b^{\perp}) \parr (g^{\perp} \triangleleft d^{\perp}) \parr (c^{\perp} \triangleleft f^{\perp})$$

- A is provable in pomset logic: the only possible proof structure is a correct proof net
- for any valid BV inference $\dfrac{B}{A}$, the premise B is not provable in pomset logic (*a fortiori* in BV)

Thank you for your attention!