Proof complexity of deep inference: a survey

Anupam Das

École Normale Supérieure de Lyon

Invited talk at LCC '16

2nd September, 2016
Marseilles

Disclaimers about this talk

Disclaimers about this talk

- Not just my own work.
- Not a comprehensive overview.
- Not mainstream proof complexity.

Disclaimers about this talk

- Not just my own work.
- Not a comprehensive overview.
- Not mainstream proof complexity.

Brace yourself...

- There will be proofs.

Disclaimers about this talk

- Not just my own work.
- Not a comprehensive overview.
- Not mainstream proof complexity.

Brace yourself...

- There will be proofs.
- There will be *formal* proofs.

Disclaimers about this talk

- Not just my own work.
- Not a comprehensive overview.
- Not mainstream proof complexity.

Brace yourself...

- There will be proofs.
- There will be *formal* proofs.

Why should you listen?

Disclaimers about this talk

- Not just my own work.
- Not a comprehensive overview.
- Not mainstream proof complexity.

Brace yourself...

- There will be proofs.
- There will be *formal* proofs.

Why should you listen?

- The proofs are nice.

Disclaimers about this talk

- Not just my own work.
- Not a comprehensive overview.
- Not mainstream proof complexity.

Brace yourself...

- There will be proofs.
- There will be *formal* proofs.

Why should you listen?

- The proofs are nice.
- There will be pictures.

Disclaimers about this talk

- Not just my own work.
- Not a comprehensive overview.
- Not mainstream proof complexity.

Brace yourself...

- There will be proofs.
- There will be *formal* proofs.

Why should you listen?

- The proofs are nice.
- There will be pictures.
- A deep area with plenty of open problems.

Outline

Proof complexity

Proof complexity

- Measures the size of proofs in terms of the size of their conclusions.

Proof complexity

- Measures the size of proofs in terms of the size of their conclusions.
- *Key dichotomy*: (quasi)polynomial vs. exponential

Proof complexity

- Measures the size of proofs in terms of the size of their conclusions.
- *Key dichotomy*: (quasi)polynomial vs. exponential

Theorem (Cook-Reckhow '74)

coNP = **NP** $\iff$ *there is a polynomially bounded proof system.*

Proof complexity

- Measures the size of proofs in terms of the size of their conclusions.
- *Key dichotomy*: (quasi)polynomial vs. exponential

Theorem (Cook-Reckhow '74)

coNP = **NP** $\iff$ *there is a polynomially bounded proof system.*

- **Cook's program:** find general methods to prove lower bounds for proofs.

Proof complexity

- Measures the size of proofs in terms of the size of their conclusions.
- *Key dichotomy*: (quasi)polynomial vs. exponential

Theorem (Cook-Reckhow '74)

coNP = **NP** $\iff$ *there is a polynomially bounded proof system.*

- **Cook's program:** find general methods to prove lower bounds for proofs.
- Proof systems can be compared by polynomial simulation.
- This induces a complexity-theoretic hierarchy of proof systems.

Proof complexity

- Measures the size of proofs in terms of the size of their conclusions.
- *Key dichotomy*: (quasi)polynomial vs. exponential

Theorem (Cook-Reckhow '74)

coNP $=$ **NP** $\iff$ *there is a polynomially bounded proof system.*

- **Cook's program:** find general methods to prove lower bounds for proofs.
- Proof systems can be compared by polynomial simulation.
- This induces a complexity-theoretic hierarchy of proof systems.
- $\rightsquigarrow$ subproblems of **coNP** vs. **NP**.

The limits of our knowledge...

The limits of our knowledge...

Cut-free LK

Tableaux

Hilbert-Frege

LK

The limits of our knowledge...

Exponential lower bounds

Cut-free LK

Tableaux

No known lower bounds

Hilbert-Frege

LK

The limits of our knowledge...

Exponential lower bounds

Cut-free LK

Tableaux

$\leq$

Resolution

No known lower bounds

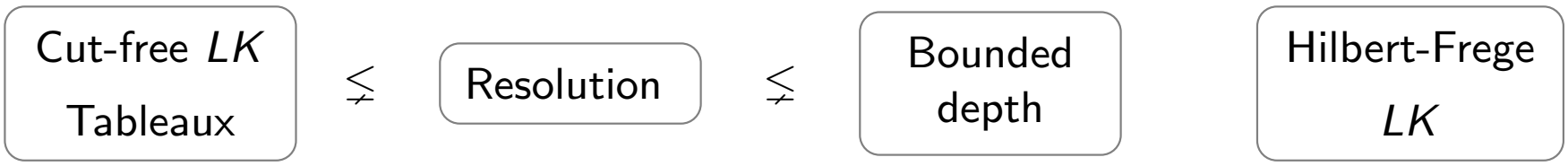
Hilbert-Frege

LK

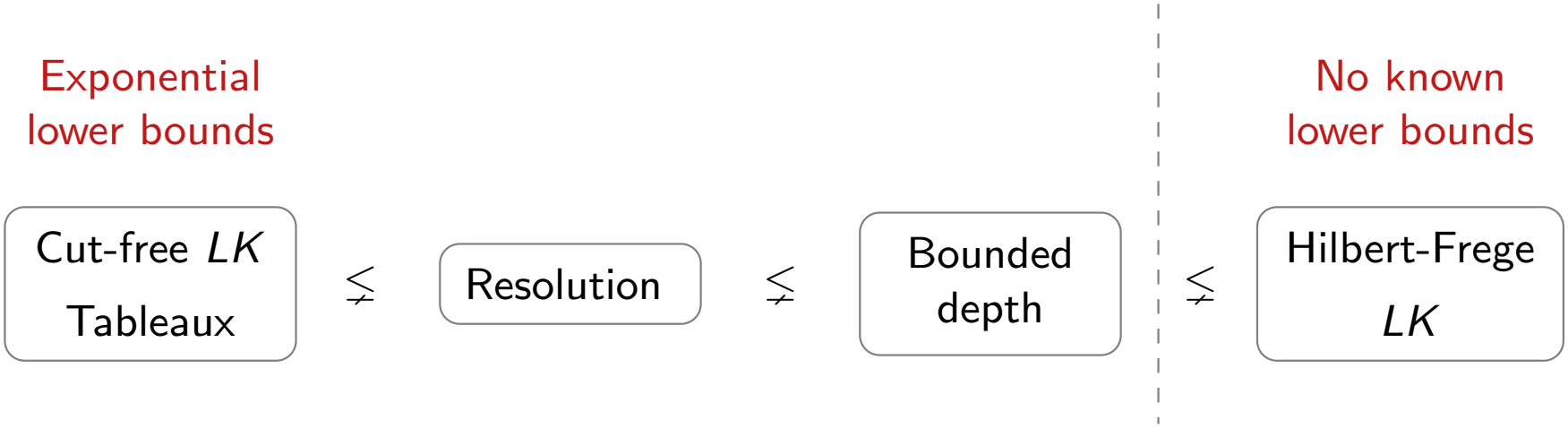
The limits of our knowledge...

Exponential lower bounds

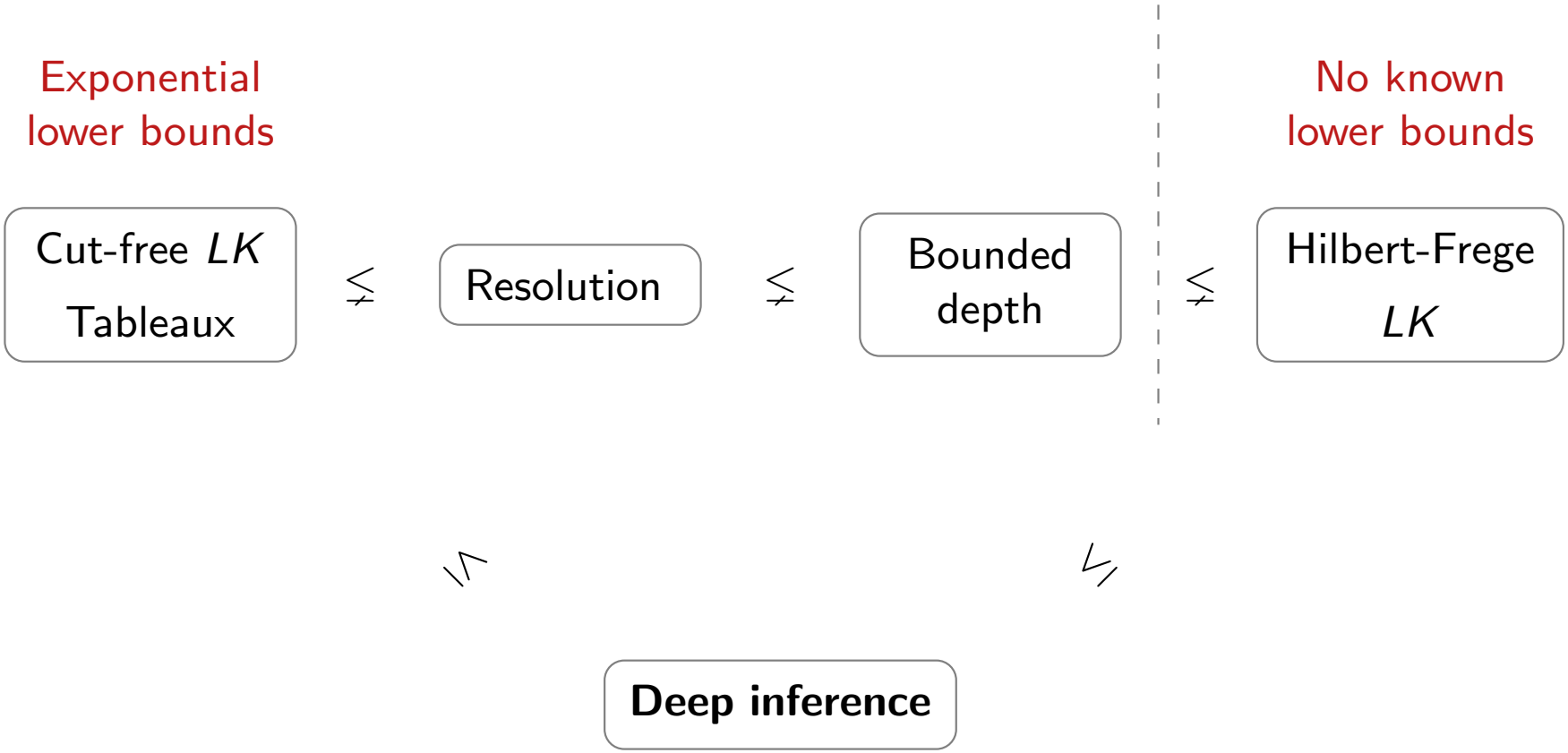
No known lower bounds



The limits of our knowledge...



The limits of our knowledge...



Deep inference: a pitch for proof theorists

Deep inference: a pitch for proof theorists

Traditionally, inference steps operate on the main connective of a formula.

Deep inference: a pitch for proof theorists

Traditionally, inference steps operate on the main connective of a formula. E.g.

$$\frac{\Gamma, A \qquad \Gamma, B}{\Gamma, A \wedge B}$$

Deep inference: a pitch for proof theorists

Traditionally, inference steps operate on the main connective of a formula. E.g.

$$\frac{\Gamma, A \qquad \Gamma, B}{\Gamma, A \wedge B}$$

In *deep inference* they may operate on any connective in a formula.

Deep inference: a pitch for proof theorists

Traditionally, inference steps operate on the main connective of a formula. E.g.

$$\frac{\Gamma, A \qquad \Gamma, B}{\Gamma, A \wedge B}$$

In *deep inference* they may operate on any connective in a formula. E.g.

$$\frac{F[A] \qquad F[B]}{F[A \wedge B]}$$

Deep inference: a pitch for proof theorists

Traditionally, inference steps operate on the main connective of a formula. E.g.

$$\frac{\Gamma, A \qquad \Gamma, B}{\Gamma, A \wedge B}$$

In *deep inference* they may operate on any connective in a formula. E.g.

$$\frac{F[A] \qquad F[B]}{F[A \wedge B]}$$

The deep inference methodology presents several advantages:

Deep inference: a pitch for proof theorists

Traditionally, inference steps operate on the main connective of a formula. E.g.

$$\frac{\Gamma, A \qquad \Gamma, B}{\Gamma, A \wedge B}$$

In *deep inference* they may operate on any connective in a formula. E.g.

$$\frac{F[A] \qquad F[B]}{F[A \wedge B]}$$

The deep inference methodology presents several advantages:

- Analytic systems for more logics.

Deep inference: a pitch for proof theorists

Traditionally, inference steps operate on the main connective of a formula. E.g.

$$\frac{\Gamma, A \qquad \Gamma, B}{\Gamma, A \wedge B}$$

In *deep inference* they may operate on any connective in a formula. E.g.

$$\frac{F[A] \qquad F[B]}{F[A \wedge B]}$$

The deep inference methodology presents several advantages:

- Analytic systems for more logics.
- Inference steps can be local.

Deep inference: a pitch for proof theorists

Traditionally, inference steps operate on the main connective of a formula. E.g.

$$\frac{\Gamma, A \qquad \Gamma, B}{\Gamma, A \wedge B}$$

In *deep inference* they may operate on any connective in a formula. E.g.

$$\frac{F[A] \qquad F[B]}{F[A \wedge B]}$$

The deep inference methodology presents several advantages:

- Analytic systems for more logics.
- Inference steps can be local.
- Systems have smaller proofs.

Systems for classical propositional logic (Guglielmi, Brünnler & Tiu '01)

The system SKS for propositional logic:

$$\mathsf{i}{\uparrow}\ \frac{A \wedge \bar{A}}{\bot} \qquad \mathsf{w}{\uparrow}\ \frac{A}{\top} \qquad \mathsf{c}{\uparrow}\ \frac{A}{A \wedge A} \qquad \mathsf{s}\ \frac{A \wedge (B \vee C)}{(A \wedge B) \vee C}$$

$$\mathsf{i}{\downarrow}\ \frac{\top}{A \vee \bar{A}} \qquad \mathsf{w}{\downarrow}\ \frac{\bot}{A} \qquad \mathsf{c}{\downarrow}\ \frac{A \vee A}{A} \qquad \mathsf{m}\ \frac{(A \wedge B) \vee (C \wedge D)}{(A \vee C) \wedge (B \vee D)}$$

Systems for classical propositional logic (Guglielmi, Brünnler & Tiu '01)

The system SKS for propositional logic:

$$\mathsf{i}{\uparrow}\ \frac{A \wedge \bar{A}}{\bot} \qquad \mathsf{w}{\uparrow}\ \frac{A}{\top} \qquad \mathsf{c}{\uparrow}\ \frac{A}{A \wedge A} \qquad \mathsf{s}\ \frac{A \wedge (B \vee C)}{(A \wedge B) \vee C}$$

$$\mathsf{i}{\downarrow}\ \frac{\top}{A \vee \bar{A}} \qquad \mathsf{w}{\downarrow}\ \frac{\bot}{A} \qquad \mathsf{c}{\downarrow}\ \frac{A \vee A}{A} \qquad \mathsf{m}\ \frac{(A \wedge B) \vee (C \wedge D)}{(A \vee C) \wedge (B \vee D)}$$

- Formulae are equivalent modulo all linear equations.

Systems for classical propositional logic (Guglielmi, Brünnler & Tiu '01)

The system SKS for propositional logic:

$$\mathsf{i}{\uparrow}\,\frac{A \wedge \bar{A}}{\bot} \qquad \mathsf{w}{\uparrow}\,\frac{A}{\top} \qquad \mathsf{c}{\uparrow}\,\frac{A}{A \wedge A} \qquad \mathsf{s}\,\frac{A \wedge (B \vee C)}{(A \wedge B) \vee C}$$

$$\mathsf{i}{\downarrow}\,\frac{\top}{A \vee \bar{A}} \qquad \mathsf{w}{\downarrow}\,\frac{\bot}{A} \qquad \mathsf{c}{\downarrow}\,\frac{A \vee A}{A} \qquad \mathsf{m}\,\frac{(A \wedge B) \vee (C \wedge D)}{(A \vee C) \wedge (B \vee D)}$$

- Formulae are equivalent modulo all linear equations.
- Proofs are obtained by sequential and parallel composition:

$$\Phi, \Psi \quad ::= \quad A \quad | \quad \Phi \star \Psi \quad | \quad \rho\,\frac{\Phi}{\Psi}$$

where $\dfrac{\mathsf{cn}(\Phi)}{\mathsf{pr}(\Psi)}$ is an instance of the rule ρ.

Systems for classical propositional logic (Guglielmi, Brünnler & Tiu '01)

The system SKS for propositional logic:

$$\mathsf{i}{\uparrow}\,\frac{A \wedge \bar{A}}{\bot} \qquad \mathsf{w}{\uparrow}\,\frac{A}{\top} \qquad \mathsf{c}{\uparrow}\,\frac{A}{A \wedge A} \qquad \mathsf{s}\,\frac{A \wedge (B \vee C)}{(A \wedge B) \vee C}$$

$$\mathsf{i}{\downarrow}\,\frac{\top}{A \vee \bar{A}} \qquad \mathsf{w}{\downarrow}\,\frac{\bot}{A} \qquad \mathsf{c}{\downarrow}\,\frac{A \vee A}{A} \qquad \mathsf{m}\,\frac{(A \wedge B) \vee (C \wedge D)}{(A \vee C) \wedge (B \vee D)}$$

- Formulae are equivalent modulo all linear equations.
- Proofs are obtained by sequential and parallel composition:

$$\Phi, \Psi \quad ::= \quad A \quad | \quad \Phi \star \Psi \quad | \quad \rho\,\frac{\Phi}{\Psi}$$

  where $\dfrac{\mathsf{cn}(\Phi)}{\mathsf{pr}(\Psi)}$ is an instance of the rule ρ.

- Proofs without $\mathsf{i}{\uparrow}$ and $\mathsf{w}{\uparrow}$ are considered analytic.

An example derivation

$$
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{p}{p \wedge p}\, c\uparrow \quad \vee \quad \dfrac{\dfrac{\bot}{p}\, w\uparrow}{p \wedge p}\, c\uparrow
}{p \vee p}\, m
}{p}\, c\downarrow
\quad = \quad
\dfrac{p}{\dfrac{\bot}{q}\, w\downarrow}
}{p \vee p}\, s
\quad \wedge \quad c\downarrow \dfrac{p \vee p}{p}
}{
\dfrac{p}{p \wedge p}\, c\uparrow \quad \vee \quad = \quad \dfrac{q \wedge w\uparrow \dfrac{p}{\top}}{q}
}
$$

Locality and complexity

- Structural steps can be reduced to atomic form.

Locality and complexity

- Structural steps can be reduced to atomic form. E.g.

$$\mathsf{c}{\downarrow}\,\frac{(A \wedge B) \vee (A \wedge B)}{A \wedge B} \quad \rightarrow \quad \mathsf{m}\,\frac{(A \wedge B) \vee (A \wedge B)}{\mathsf{c}{\downarrow}\,\frac{A \vee A}{A} \wedge \mathsf{c}{\downarrow}\,\frac{B \vee B}{B}}$$

Locality and complexity

- Structural steps can be reduced to atomic form. E.g.

$$\mathsf{c}{\downarrow}\,\frac{(A \wedge B) \vee (A \wedge B)}{A \wedge B} \quad \rightarrow \quad \mathsf{m}\,\frac{(A \wedge B) \vee (A \wedge B)}{\mathsf{c}{\downarrow}\,\frac{A \vee A}{A} \wedge \mathsf{c}{\downarrow}\,\frac{B \vee B}{B}}$$

- Logical rules are linear, so all steps are local.

Locality and complexity

- Structural steps can be reduced to atomic form. E.g.

$$\mathsf{c}{\downarrow}\,\frac{(A \wedge B) \vee (A \wedge B)}{A \wedge B} \quad \rightarrow \quad \mathsf{m}\,\frac{(A \wedge B) \vee (A \wedge B)}{\mathsf{c}{\downarrow}\,\frac{A \vee A}{A} \wedge \mathsf{c}{\downarrow}\,\frac{B \vee B}{B}}$$

- Logical rules are linear, so all steps are local.

- **Slogan:** Deep inference simultaneously optimises checking complexity and communication complexity.

A useful abstraction of proofs

A useful abstraction of proofs

Definition

The **atomic flow** of a proof is the graph obtained by tracing the path of each atom through the proof, designating nodes at structural steps.

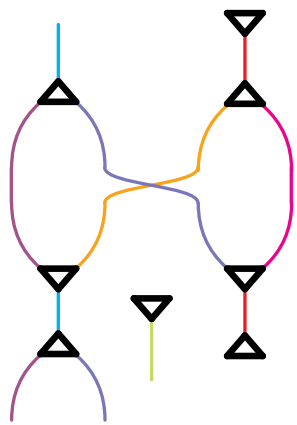
A useful abstraction of proofs

Definition

The **atomic flow** of a proof is the graph obtained by tracing the path of each atom through the proof, designating nodes at structural steps.

E.g.

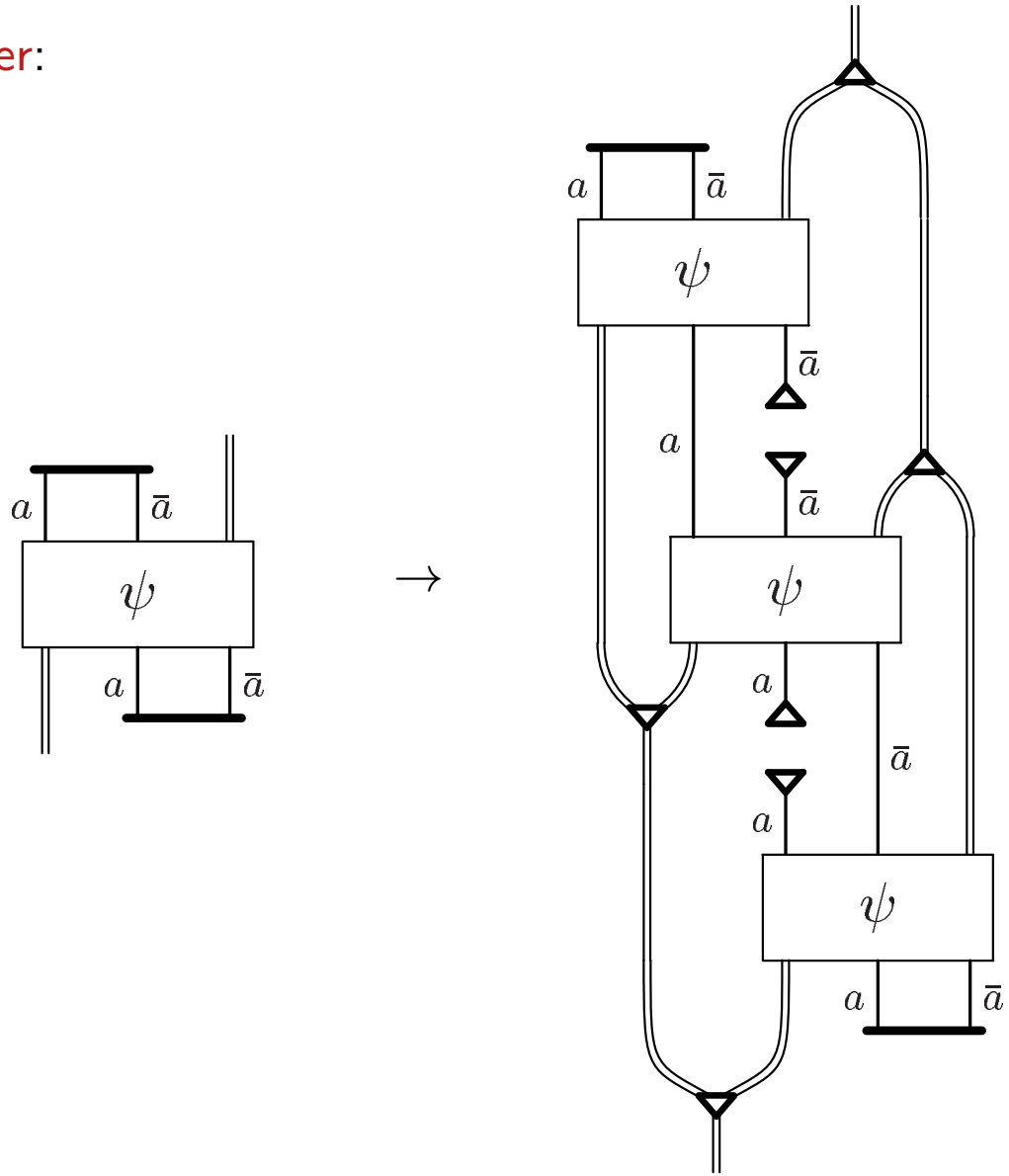
$$\dfrac{\dfrac{\dfrac{\dfrac{p}{p \wedge p}\, c\uparrow}{p \vee p}\vee \quad \dfrac{\dfrac{\bot}{p}\, w\uparrow}{p \wedge p}\, c\uparrow}{p}\, m}{\dfrac{p \vee p}{p}\, c\downarrow = \dfrac{p}{p \vee \dfrac{\bot}{q}\, w\downarrow} \wedge \dfrac{p \vee p}{p}\, c\downarrow}\, s \qquad \dfrac{\dfrac{p}{p \wedge p}\, c\uparrow}{p}\vee = \dfrac{q \wedge \dfrac{p}{\top}\, w\uparrow}{q}$$



Cut-elimination via flows (Guglielmi & Gundersen '08)

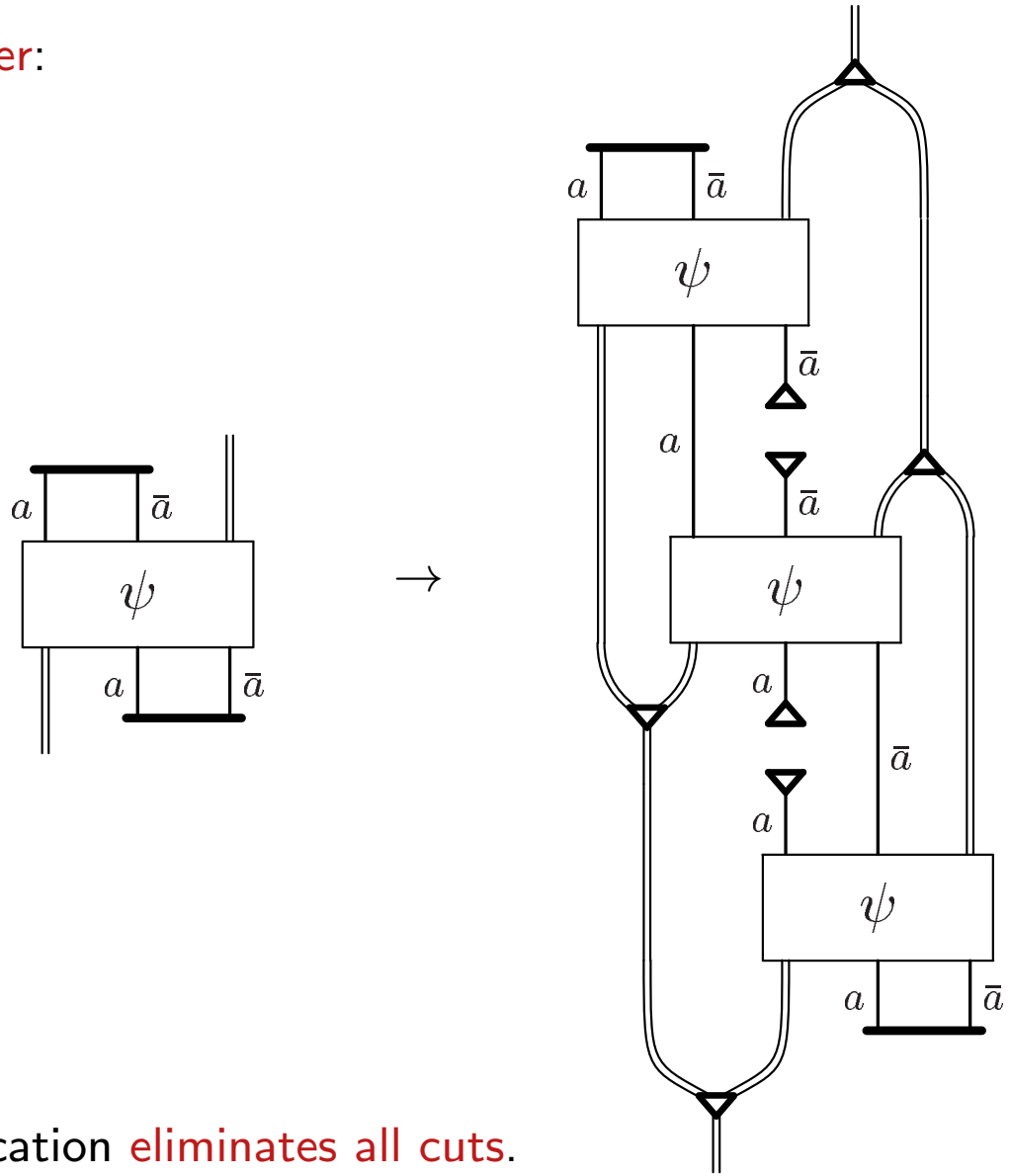
Cut-elimination via flows (Guglielmi & Gundersen '08)

The path breaker:



Cut-elimination via flows (Guglielmi & Gundersen '08)

The path breaker:



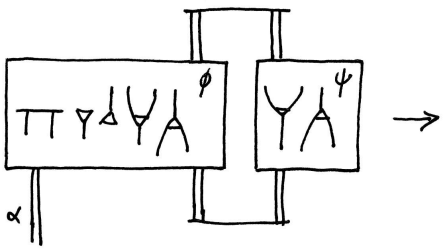
Recursive application eliminates all cuts.

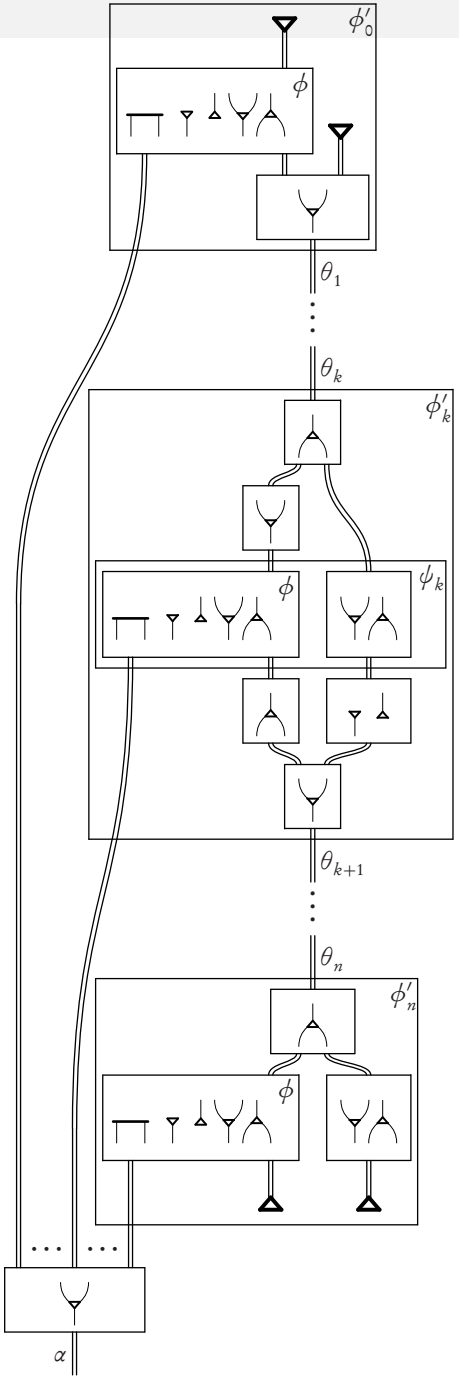
Outline

Doing better than the pathbreaker

Doing better than the pathbreaker

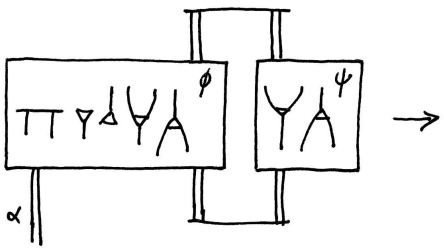
We can obtain a length-width tradeoff:

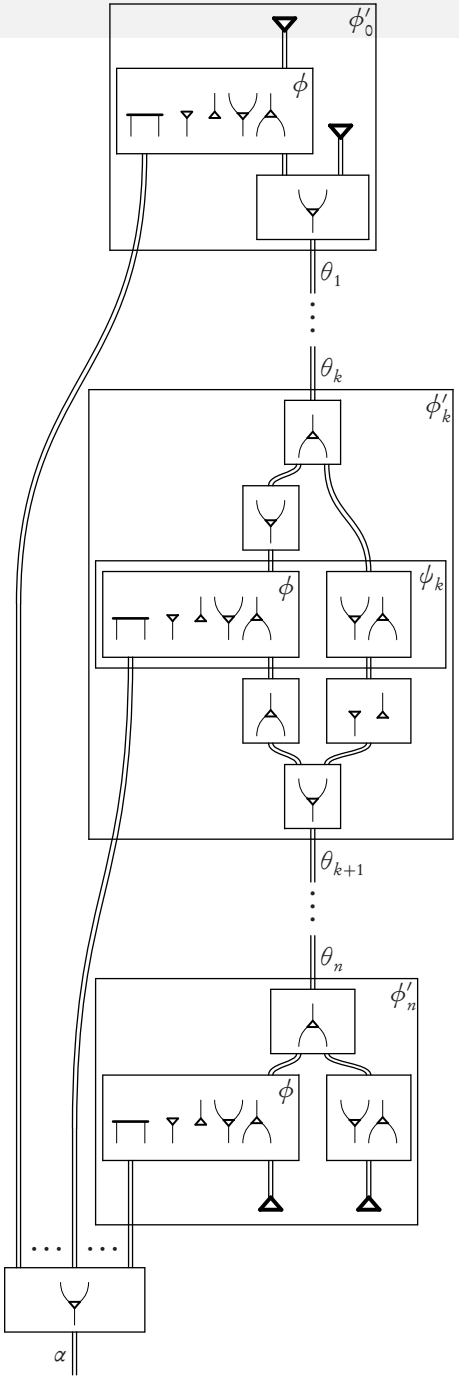




Doing better than the pathbreaker

We can obtain a length-width tradeoff:





We rely on a logical catalyst...

Formulae for counting

Definition (Threshold functions)

The Boolean functions $TH_k^n : \{0,1\}^n \to \{0,1\}$ are defined by

$$\sigma \mapsto 1 \quad \text{if} \quad \sum_{i=1}^{n} \sigma_i \geq k \quad .$$

Definition (Threshold functions)

The Boolean functions $TH_k^n : \{0,1\}^n \to \{0,1\}$ are defined by

$$\sigma \mapsto 1 \quad \text{if} \quad \sum_{i=1}^{n} \sigma_i \geq k \quad .$$

- These functions are monotone, i.e.

$$\tau \geq \sigma \implies TH_k^n(\tau) \geq TH_k^n(\sigma) \quad .$$

Formulae for counting

Definition (Threshold functions)

The Boolean functions $TH_k^n : \{0,1\}^n \to \{0,1\}$ are defined by

$$\sigma \mapsto 1 \quad \text{if} \quad \sum_{i=1}^{n} \sigma_i \geq k \quad .$$

- These functions are monotone, i.e.

$$\tau \geq \sigma \implies TH_k^n(\tau) \geq TH_k^n(\sigma) \quad .$$

- We can compute them over $\{\bot, \top, \vee, \wedge\}$, e.g.

$$\bigvee_{\sum_i \sigma_i \geq k} \bigwedge_{\sigma_i = 1} a_i \quad .$$

Formulae for counting

Definition (Threshold functions)

The Boolean functions $TH_k^n : \{0,1\}^n \to \{0,1\}$ are defined by

$$\sigma \mapsto 1 \quad \text{if} \quad \sum_{i=1}^{n} \sigma_i \geq k \quad .$$

- These functions are monotone, i.e.

$$\tau \geq \sigma \implies TH_k^n(\tau) \geq TH_k^n(\sigma) \quad .$$

- We can compute them over $\{\bot, \top, \vee, \wedge\}$, e.g.

$$\bigvee_{\sum_i \sigma_i \geq k} \bigwedge_{\sigma_i = 1} a_i \quad .$$

- Unfortunately too big: they have size $\approx 2^n$.

Formulae for counting

However, notice the following identity:

$$TH_k^{2n}(\mathbf{a}, \mathbf{b}) \quad = \quad \bigvee_{i=0}^{k} TH_i^n(\mathbf{a}) \wedge TH_{k-i}^n(\mathbf{b})$$

Formulae for counting

However, notice the following identity:

$$TH_k^{2n}(\mathbf{a},\mathbf{b}) \quad = \quad \bigvee_{i=0}^{k} TH_i^n(\mathbf{a}) \wedge TH_{k-i}^n(\mathbf{b})$$

This yields formulae of depth $O(\log n)$ using gates of size k.

Proposition

There are $n^{O(\log n)}$-size monotone formulae th_k^n computing TH_k^n.

Formulae for counting

However, notice the following identity:

$$TH_k^{2n}(\mathbf{a},\mathbf{b}) \quad = \quad \bigvee_{i=0}^{k} TH_i^n(\mathbf{a}) \wedge TH_{k-i}^n(\mathbf{b})$$

This yields formulae of depth $O(\log n)$ using gates of size k.

Proposition

There are $n^{O(\log n)}$-size monotone formulae th_k^n computing TH_k^n.

Theorem

The following have $n^{O(\log n)}$-size derivations in $\{\mathsf{w}\uparrow, \mathsf{w}\downarrow, \mathsf{s}\}$:

$$\begin{array}{ccc} \top & th_{n+1}^n(\mathbf{a}) & th_k^n(\cdots \bot \cdots) \\ (1)\,\| & (2)\,\| & (3)\,\| \\ th_0^n(\mathbf{a}) & \bot & th_{k+1}^n(\cdots \top \cdots) \end{array}$$

Formulae for counting

However, notice the following identity:

$$TH_k^{2n}(\mathbf{a},\mathbf{b}) \quad = \quad \bigvee_{i=0}^{k} TH_i^n(\mathbf{a}) \wedge TH_{k-i}^n(\mathbf{b})$$

This yields formulae of depth $O(\log n)$ using gates of size k.

Proposition

There are $n^{O(\log n)}$-size monotone formulae th_k^n computing TH_k^n.

Theorem

The following have $n^{O(\log n)}$-size derivations in $\{\mathrm{w}\uparrow, \mathrm{w}\downarrow, \mathrm{s}\}$:

$$\begin{array}{ccc} \top & th_{n+1}^n(\mathbf{a}) & th_k^n(\cdots \bot \cdots) \\ (1)\,\| & (2)\,\| & (3)\,\| \\ th_0^n(\mathbf{a}) & \bot & th_{k+1}^n(\cdots \top \cdots) \end{array}$$

Proof.

(1) and (2) are simple. (3) requires a bit of work... □

Increasing thresholds: proof of (3)

Proof is by induction on n

Increasing thresholds: proof of (3)

Proof is by induction on n :

$$= \frac{th_k^{2n}(\cdots \bot \cdots, \boldsymbol{b})}{\bigvee_{i=0}^{k} \left(\begin{array}{c} th_i^n(\cdots \bot \cdots) \\ IH \,\| \\ th_{i+1}^n(\cdots \top \cdots) \end{array} \wedge th_{k-i}^n(\boldsymbol{b}) \right) \vee \mathrm{w}{\downarrow} \dfrac{\bot}{th_0^n(\cdots \top \cdots) \wedge th_{k+1}^n(\boldsymbol{b})}}$$

$$= \frac{}{th_{k+1}^{2n}(\cdots \top \cdots, \boldsymbol{b})}$$

Increasing thresholds: proof of (3)

Proof is by induction on n :

$$= \dfrac{th_k^{2n}(\cdots \bot \cdots, \boldsymbol{b})}{\bigvee_{i=0}^{k} \begin{pmatrix} th_i^n(\cdots \bot \cdots) \\ IH \,\|\, \\ th_{i+1}^n(\cdots \top \cdots) \end{pmatrix} \wedge th_{k-i}^n(\boldsymbol{b}) \; \vee \mathrm{w}{\downarrow} \dfrac{\bot}{th_0^n(\cdots \top \cdots) \wedge th_{k+1}^n(\boldsymbol{b})}}$$

$$= \dfrac{}{th_{k+1}^{2n}(\cdots \top \cdots, \boldsymbol{b})}$$

(**NB:** Notice the crucial use of deep inference!)

Main result

Lemma

There is a quasipolynomial transformation of the following format,

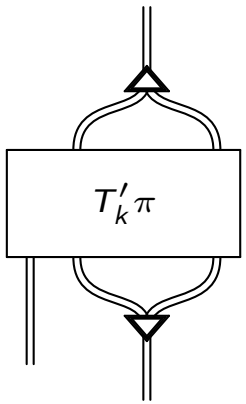
$$\pi \Big\| \text{SKS} \quad \tau \qquad \longrightarrow \qquad \begin{array}{c} th_k^n(\mathbf{a}) \\ T_k\pi \Big\| \textit{cut-free} \\ \tau \vee th_{k+1}^n(\mathbf{a}) \end{array}$$

Main result

Lemma

There is a quasipolynomial transformation of the following format,

$$\pi \Big\| \text{SKS} \quad \tau \qquad \longrightarrow \qquad \begin{array}{c} th^n_k(\mathbf{a}) \\ T_k\pi \,\Big\|\, \textit{cut-free} \\ \tau \vee th^n_{k+1}(\mathbf{a}) \end{array}$$



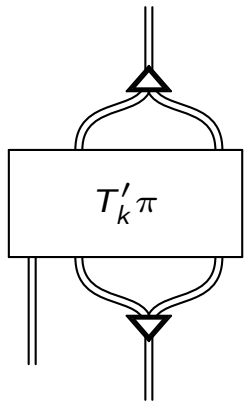
with flows of constant length.

Main result

Lemma

There is a quasipolynomial transformation of the following format,

$$\pi \Big\| \text{SKS} \quad \tau \qquad \longrightarrow \qquad \begin{array}{c} th^n_k(\mathbf{a}) \\ T_k\pi \Big\| \textit{cut-free} \\ \tau \vee th^n_{k+1}(\mathbf{a}) \end{array}$$

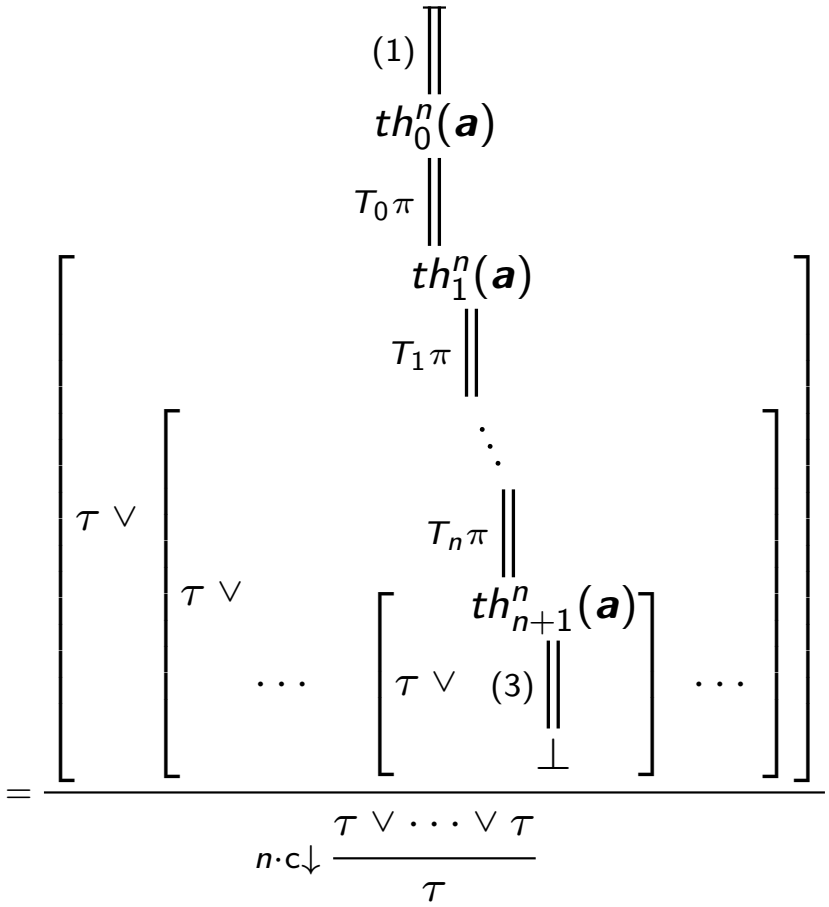


with flows of constant length.

Theorem (Jeřábek '09, Bruscoli et al. '10)

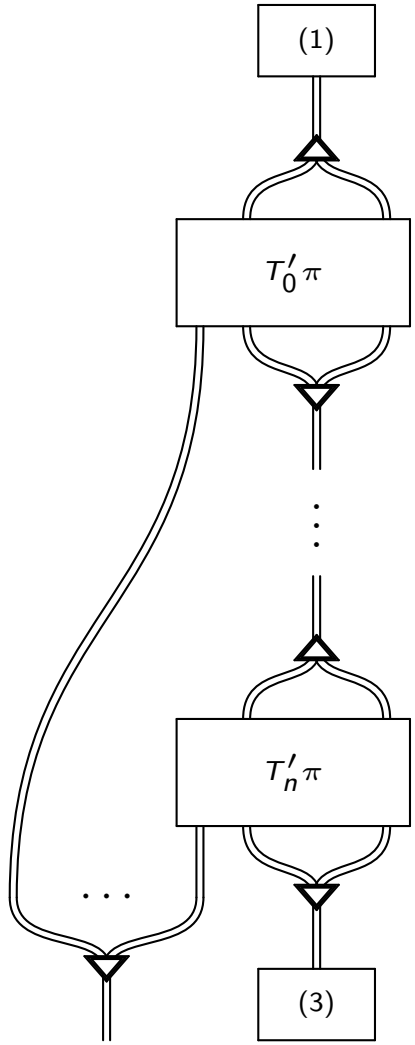
SKS *has quasipolynomial-time cut-elimination.*

Proof of main result



Proof of main result

$$
= \dfrac{\begin{array}{c}
(1)\ \| \\
th_0^n(\boldsymbol{a}) \\
T_0\pi\ \| \\
th_1^n(\boldsymbol{a}) \\
T_1\pi\ \| \\
\vdots \\
T_n\pi\ \| \\
\left[\tau \vee \left[\tau \vee \cdots \vee \left[\tau \vee (3)\ \| \right] \cdots\right]\right]
\end{array}}{n\cdot \mathrm{c}\!\downarrow \dfrac{\tau \vee \cdots \vee \tau}{\tau}}
$$



Outline

Analytic subsystems of deep inference

Recall the system SKS:

$$\mathsf{i}\uparrow \frac{A \wedge \bar{A}}{\bot} \qquad \mathsf{w}\uparrow \frac{A}{\top} \qquad \mathsf{c}\uparrow \frac{A}{A \wedge A} \qquad \mathsf{s} \frac{A \wedge (B \vee C)}{(A \wedge B) \vee C}$$

$$\mathsf{i}\downarrow \frac{\top}{A \vee \bar{A}} \qquad \mathsf{w}\downarrow \frac{\bot}{A} \qquad \mathsf{c}\downarrow \frac{A \vee A}{A} \qquad \mathsf{m} \frac{(A \wedge B) \vee (C \wedge D)}{(A \vee C) \wedge (B \vee D)}$$

Analytic subsystems of deep inference

Recall the system SKS:

$$\mathsf{i}\uparrow \frac{A \wedge \bar{A}}{\bot} \qquad \mathsf{w}\uparrow \frac{A}{\top} \qquad \mathsf{c}\uparrow \frac{A}{A \wedge A} \qquad \mathsf{s} \frac{A \wedge (B \vee C)}{(A \wedge B) \vee C}$$

$$\mathsf{i}\downarrow \frac{\top}{A \vee \bar{A}} \qquad \mathsf{w}\downarrow \frac{\bot}{A} \qquad \mathsf{c}\downarrow \frac{A \vee A}{A} \qquad \mathsf{m} \frac{(A \wedge B) \vee (C \wedge D)}{(A \vee C) \wedge (B \vee D)}$$

Definition

- $\mathsf{KS}^+ := \mathsf{SKS} \setminus \{\mathsf{i}\uparrow\}$. (cut-free deep inference)
- $\mathsf{KS} := \mathsf{SKS} \setminus \{\mathsf{i}\uparrow, \mathsf{w}\uparrow, \mathsf{c}\uparrow\}$. (minimal deep inference)

From KS^+ to KS

- KS^+ is quite powerful: almost as strong as Hilbert-Frege systems.

From KS^+ to KS

- KS^+ is quite powerful: almost as strong as Hilbert-Frege systems.
- What about KS?

From KS^+ to KS

- KS^+ is quite powerful: almost as strong as Hilbert-Frege systems.
- What about KS? Crucially, the following rule is absent:

$$\mathsf{c}{\uparrow}\ \frac{A}{A \wedge A}$$

From KS^+ to KS

- KS^+ is quite powerful: almost as strong as Hilbert-Frege systems.
- What about KS? Crucially, the following rule is absent:

$$\mathsf{c}{\uparrow}\ \frac{A}{A \wedge A}$$

$\rightsquigarrow$ no sharing.

From KS^+ to KS

- KS^+ is quite powerful: almost as strong as Hilbert-Frege systems.
- What about KS? Crucially, the following rule is absent:

$$\mathsf{c}{\uparrow}\ \frac{A}{A \wedge A}$$

  $\rightsquigarrow$ no sharing.

- We consider the question of KS^+ vs. KS via normalisation procedures.

From KS^+ to KS

- KS^+ is quite powerful: almost as strong as Hilbert-Frege systems.
- What about KS? Crucially, the following rule is absent:

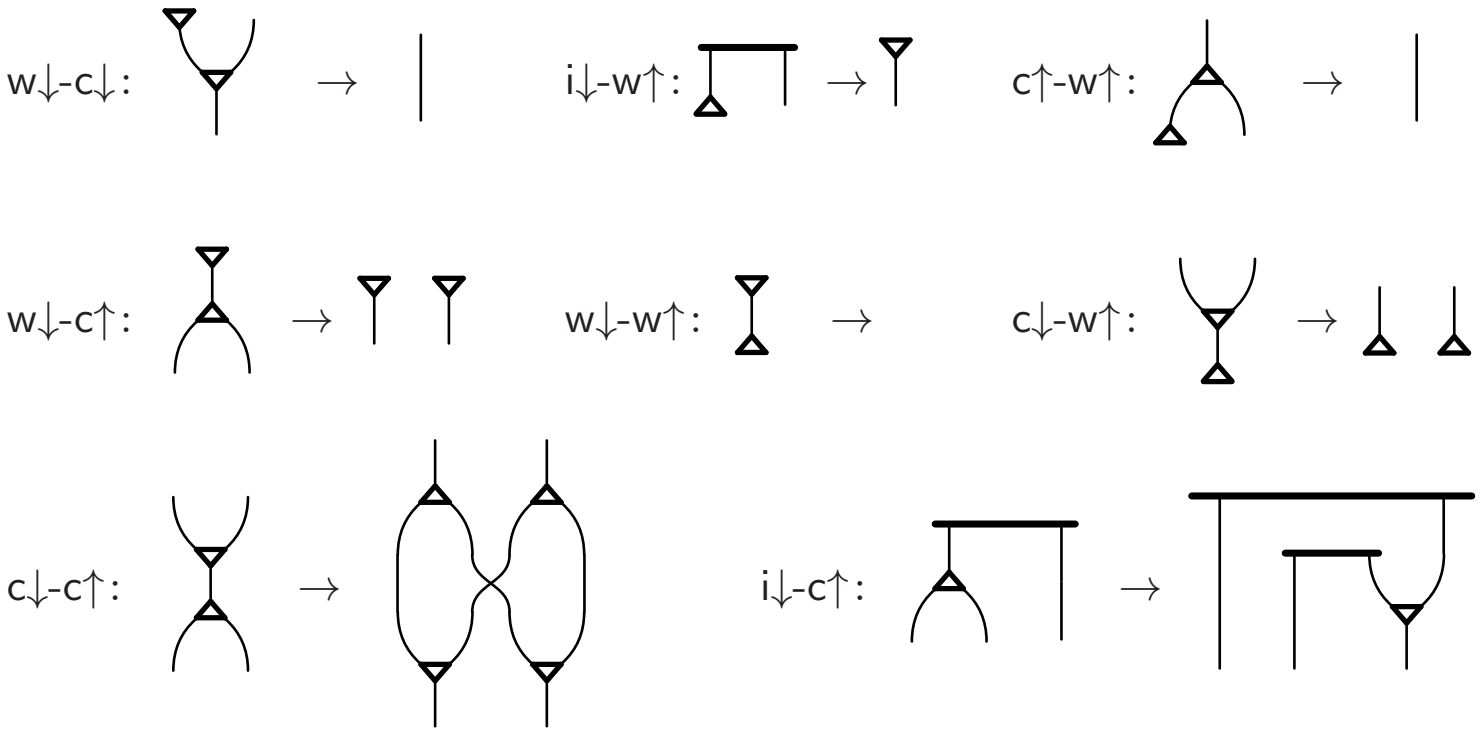
$$\mathsf{c}{\uparrow}\ \frac{A}{A \wedge A}$$

  $\rightsquigarrow$ no sharing.

- We consider the question of KS^+ vs. KS via normalisation procedures.
- In particular, we rely on local graph rewriting on atomic flows...

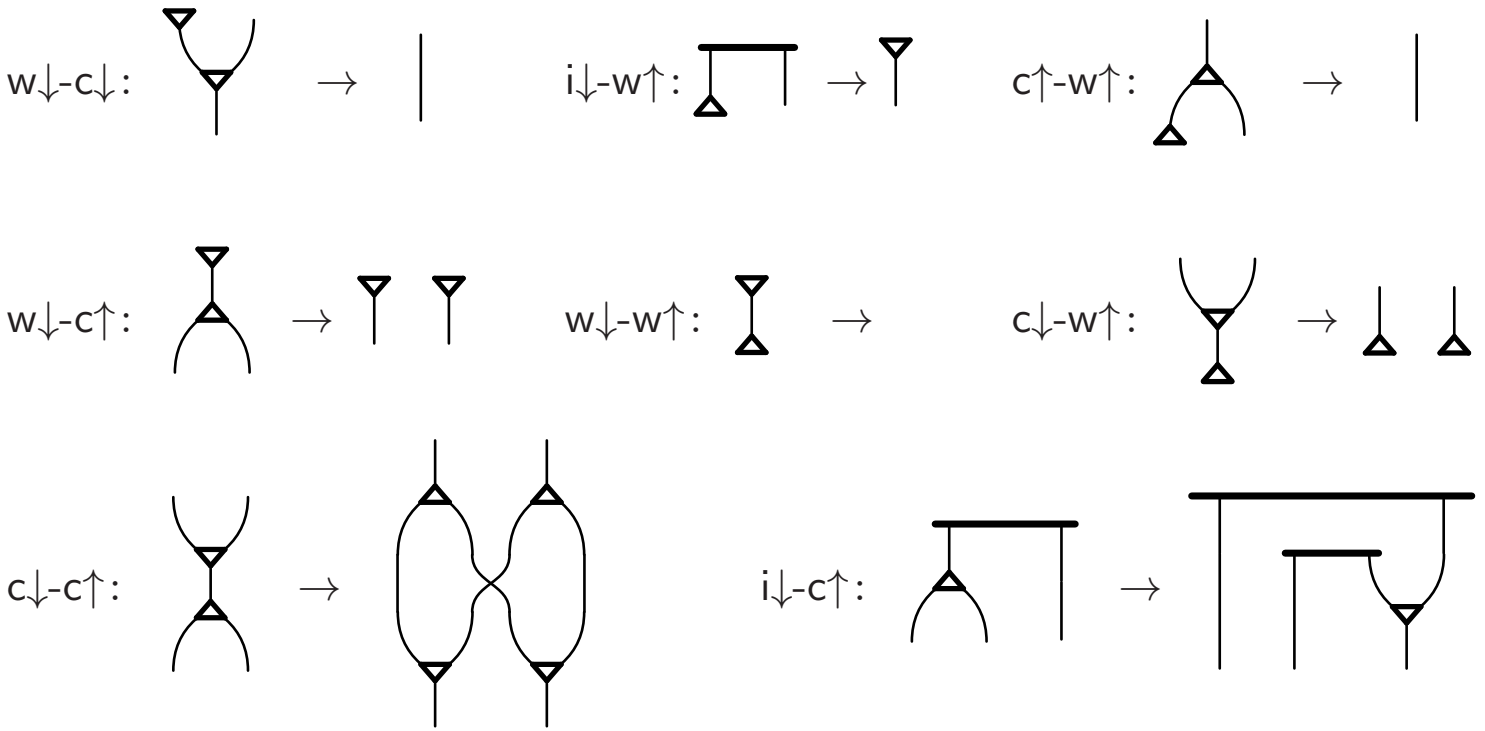
Normalisation procedures

Consider the rewrite system **norm** on atomic flows:



Normalisation procedures

Consider the rewrite system **norm** on atomic flows:

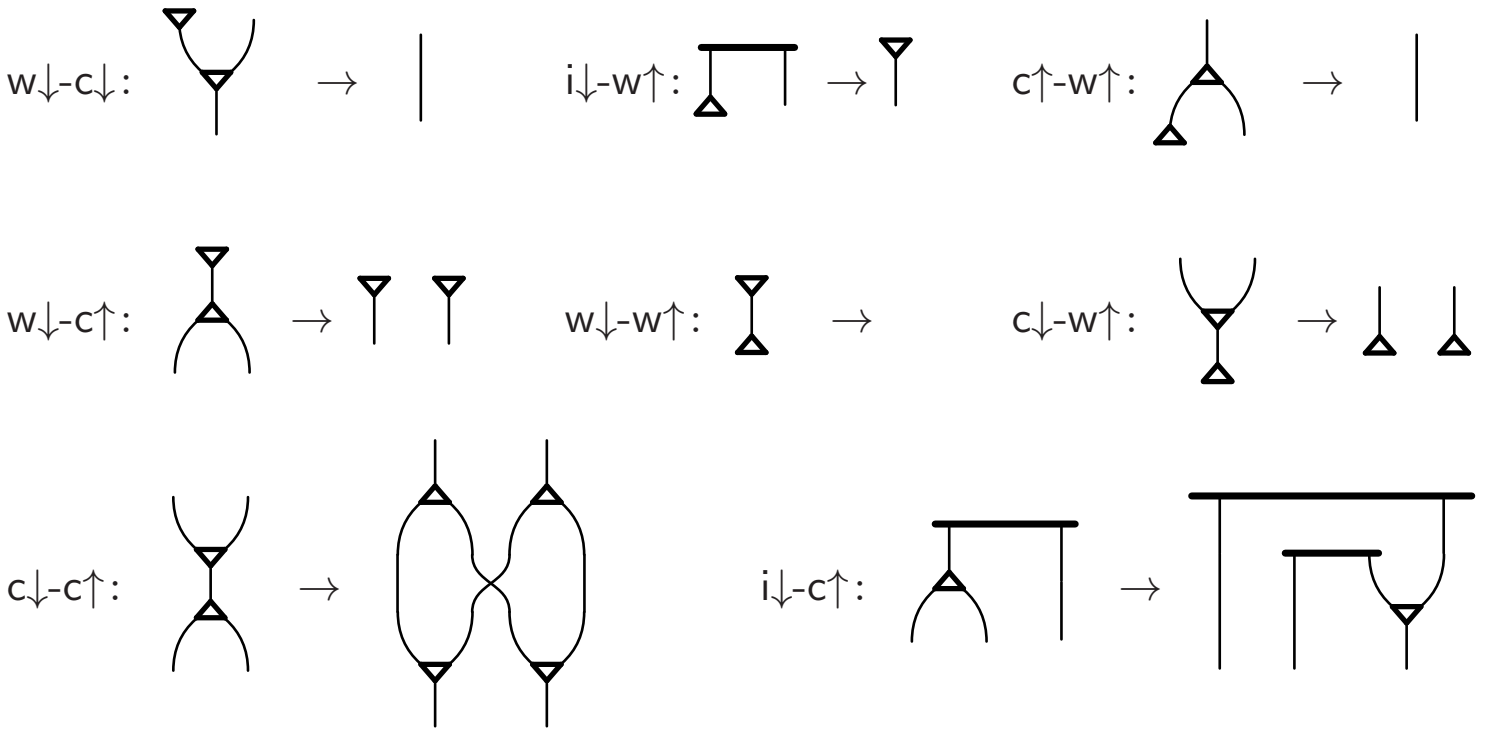


Theorem

1. **norm** *is terminating and confluent.*

Normalisation procedures

Consider the rewrite system **norm** on atomic flows:

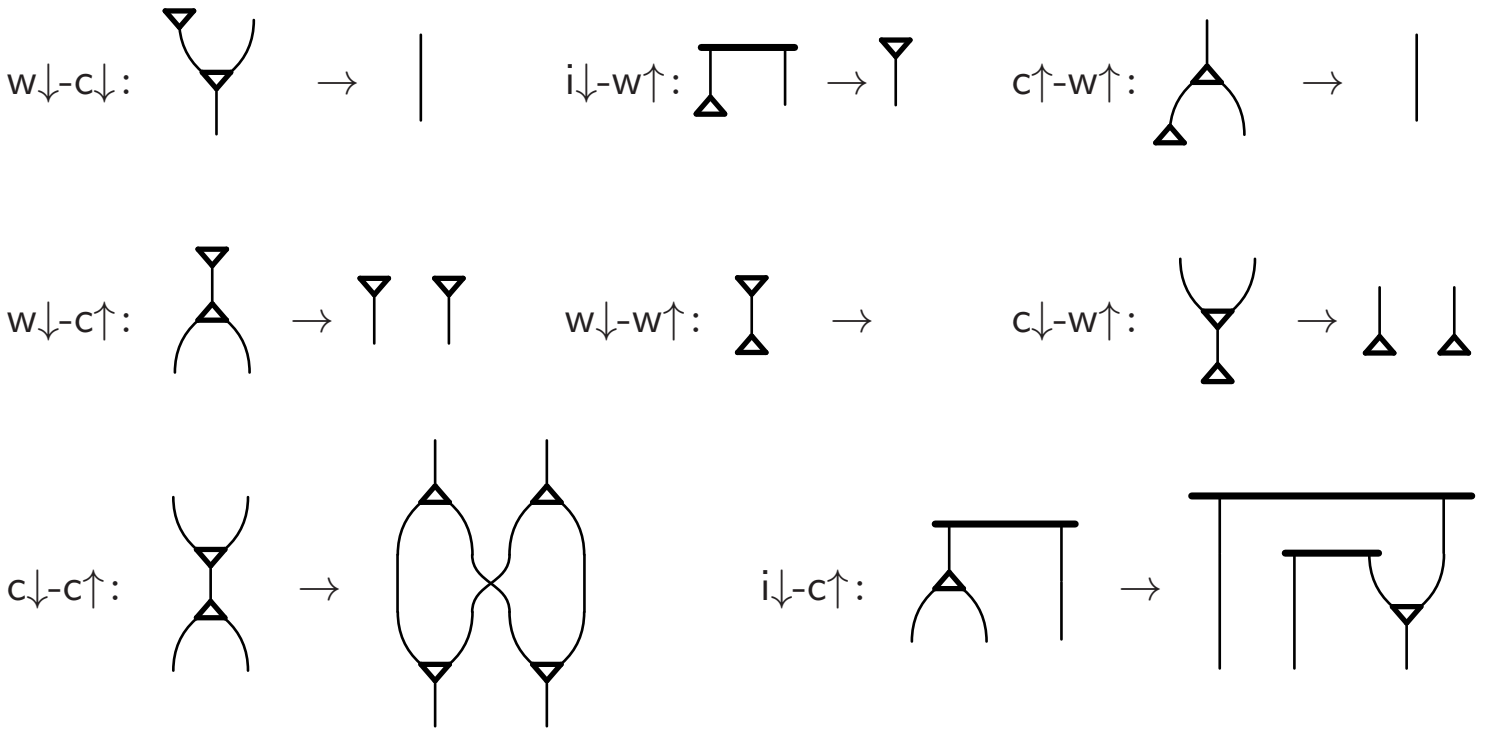


Theorem

1. **norm** *is terminating and confluent.*
2. *Induces normalisation procedure from* KS^+ *to* KS*.*

Normalisation procedures

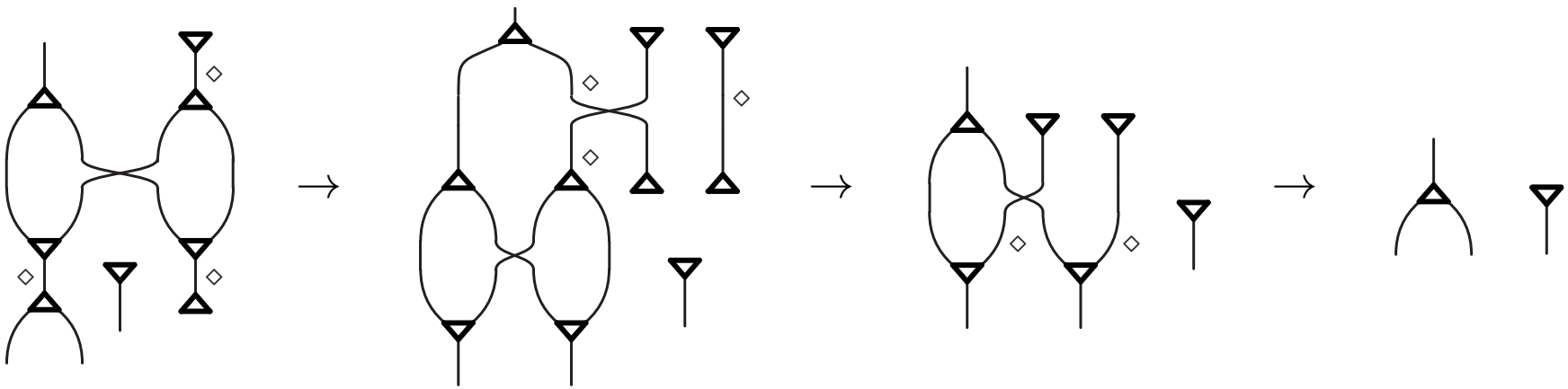
Consider the rewrite system **norm** on atomic flows:



Theorem

1. **norm** *is terminating and confluent.*
2. *Induces normalisation procedure from* KS^+ *to* KS.
3. *Complexity is polynomial time in number of paths of an input flow.*

Example of a flow reduction

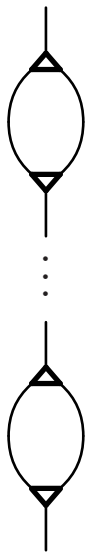


Complexity issues

- When is normalisation efficient?

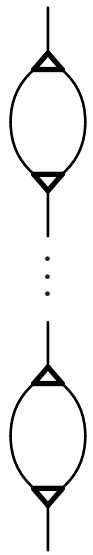
Complexity issues

- When is normalisation efficient?
- A typical problem:



Complexity issues

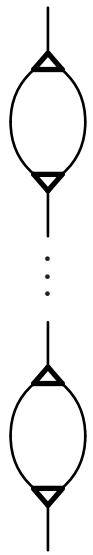
- When is normalisation efficient?
- A typical problem:



- If there are n loops then the number of (maximal) paths is 2^n.

Complexity issues

- When is normalisation efficient?
- A typical problem:



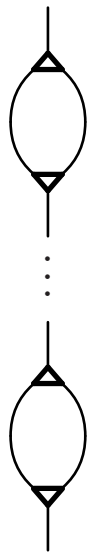
- If there are n loops then the number of (maximal) paths is 2^n.

Observation

norm *takes time exponential in the length of an input flow.*

Complexity issues

- When is normalisation efficient?
- A typical problem:



- If there are n loops then the number of (maximal) paths is 2^n.

Observation

norm *takes time exponential in the length of an input flow.*

- **NB:** Quasipolynomial-time normalisation uses flows of length $O(n)$.

Outline

The pigeonhole principle

The pigeonhole principle

- The *pigeonhole principle* is a fundamental tool in combinatorics:

  "if there are m pigeons in n holes, and $m > n$, then there must be two pigeons in some hole".

The pigeonhole principle

- The *pigeonhole principle* is a fundamental tool in combinatorics:

  "if there are m pigeons in n holes, and $m > n$, then there must be two pigeons in some hole".

- It can be encoded as a class of propositional tautologies:

$$\mathrm{PHP}_n^m := \bigwedge_{i=1}^{m} \bigvee_{j=1}^{n} p_{ij} \rightarrow \bigvee_{j=1}^{n} \bigvee_{i \neq i'} (p_{ij} \wedge p_{i'j})$$

(Read p_{ij} as "pigeon i sits in hole j".)

The pigeonhole principle

- The *pigeonhole principle* is a fundamental tool in combinatorics:

  "if there are m pigeons in n holes, and $m > n$, then there must be two pigeons in some hole".

- It can be encoded as a class of propositional tautologies:

$$\mathrm{PHP}_n^m := \bigwedge_{i=1}^{m} \bigvee_{j=1}^{n} p_{ij} \rightarrow \bigvee_{j=1}^{n} \bigvee_{i \neq i'} (p_{ij} \wedge p_{i'j})$$

  (Read p_{ij} as "pigeon i sits in hole j".)

- PHP_n^{n+1} is a benchmark class of tautologies in proof complexity.

The pigeonhole principle

- The *pigeonhole principle* is a fundamental tool in combinatorics:

  "if there are m pigeons in n holes, and $m > n$, then there must be two pigeons in some hole".

- It can be encoded as a class of propositional tautologies:

$$\mathrm{PHP}_n^m := \bigwedge_{i=1}^{m} \bigvee_{j=1}^{n} p_{ij} \rightarrow \bigvee_{j=1}^{n} \bigvee_{i \neq i'} (p_{ij} \wedge p_{i'j})$$

  (Read p_{ij} as "pigeon i sits in hole j".)

- PHP_n^{n+1} is a benchmark class of tautologies in proof complexity.
- We construct KS^+-proofs of PHP_n^{n+1} of polylogarithmic length,

The pigeonhole principle

- The *pigeonhole principle* is a fundamental tool in combinatorics:

  "if there are m pigeons in n holes, and $m > n$, then there must be two pigeons in some hole".

- It can be encoded as a class of propositional tautologies:

$$\mathrm{PHP}_n^m := \bigwedge_{i=1}^{m} \bigvee_{j=1}^{n} p_{ij} \rightarrow \bigvee_{j=1}^{n} \bigvee_{i \neq i'} (p_{ij} \wedge p_{i'j})$$

  (Read p_{ij} as "pigeon i sits in hole j".)

- PHP_n^{n+1} is a benchmark class of tautologies in proof complexity.
- We construct KS^+-proofs of PHP_n^{n+1} of polylogarithmic length, hence:

Theorem (D. '14)

There are quasipolynomial-size proofs of PHP_n^{n+1} *in* KS.

Flashback to threshold functions

Flashback to threshold functions

- Remember those threshold formulae?

$$th_k^{2n}(\mathbf{a},\mathbf{b}) \quad := \quad \bigvee_{i=0}^{k} th_i^n(\mathbf{a}) \wedge th_{k-i}^n(\mathbf{b})$$

Flashback to threshold functions

- Remember those threshold formulae?

$$th_k^{2n}(\mathbf{a},\mathbf{b}) \quad := \quad \bigvee_{i=0}^{k} th_i^n(\mathbf{a}) \wedge th_{k-i}^n(\mathbf{b})$$

- Notice that counting functions are symmetric:

$$TH_k^n(\mathbf{a}) = TH_k^n(\sigma(\mathbf{a}))$$

  for any permutation $\sigma \in S_n$.

Flashback to threshold functions

- Remember those threshold formulae?

$$th_k^{2n}(\mathbf{a},\mathbf{b}) \quad := \quad \bigvee_{i=0}^{k} th_i^n(\mathbf{a}) \wedge th_{k-i}^n(\mathbf{b})$$

- Notice that counting functions are symmetric:

$$TH_k^n(\mathbf{a}) = TH_k^n(\sigma(\mathbf{a}))$$

  for any permutation $\sigma \in S_n$.

- But what is the complexity of proving this?

Flashback to threshold functions

- Remember those threshold formulae?

$$th_k^{2n}(\mathbf{a},\mathbf{b}) \quad := \quad \bigvee_{i=0}^{k} th_i^n(\mathbf{a}) \wedge th_{k-i}^n(\mathbf{b})$$

- Notice that counting functions are symmetric:

$$TH_k^n(\mathbf{a}) = TH_k^n(\sigma(\mathbf{a}))$$

  for any permutation $\sigma \in S_n$.

- But what is the complexity of proving this?

Question

What is the size of proofs of $th_k^n(\mathbf{a}) \to th_k^n(\sigma(\mathbf{a}))$*?*

Proof idea

- Let LPHP_n^{n+1} and RPHP_n^{n+1} be the LHS and RHS of PHP_n^{n+1}, respectively.

Proof idea

- Let LPHP_n^{n+1} and RPHP_n^{n+1} be the LHS and RHS of PHP_n^{n+1}, respectively.
- We will use the following outline of a proof of PHP_n^{n+1}:

$$\mathrm{LPHP}_n^{n+1} \quad \overset{(1)}{\Longrightarrow} \quad th_{n+1}^{n(n+1)}(p_{ij})$$

Proof idea

- Let LPHP_n^{n+1} and RPHP_n^{n+1} be the LHS and RHS of PHP_n^{n+1}, respectively.
- We will use the following outline of a proof of PHP_n^{n+1}:

$$\begin{aligned} \mathrm{LPHP}_n^{n+1} &\overset{(1)}{\Longrightarrow} th_{n+1}^{n(n+1)}(p_{ij}) \\ &\overset{(2)}{\Longrightarrow} th_{n+1}^{n(n+1)}(p_{ji}) \end{aligned}$$

Proof idea

- Let LPHP_n^{n+1} and RPHP_n^{n+1} be the LHS and RHS of PHP_n^{n+1}, respectively.
- We will use the following outline of a proof of PHP_n^{n+1}:

$$\begin{aligned} \mathrm{LPHP}_n^{n+1} &\overset{(1)}{\Longrightarrow} th_{n+1}^{n(n+1)}(p_{ij}) \\ &\overset{(2)}{\Longrightarrow} th_{n+1}^{n(n+1)}(p_{ji}) \\ &\overset{(3)}{\Longrightarrow} \mathrm{RPHP}_n^{n+1}. \end{aligned}$$

Proof idea

- Let LPHP_n^{n+1} and RPHP_n^{n+1} be the LHS and RHS of PHP_n^{n+1}, respectively.
- We will use the following outline of a proof of PHP_n^{n+1}:

$$\begin{aligned} \mathrm{LPHP}_n^{n+1} &\overset{(1)}{\Longrightarrow} th_{n+1}^{n(n+1)}(p_{ij}) \\ &\overset{(2)}{\Longrightarrow} th_{n+1}^{n(n+1)}(p_{ji}) \\ &\overset{(3)}{\Longrightarrow} \mathrm{RPHP}_n^{n+1}. \end{aligned}$$

- This basic proof template was first used in Buss '87, and adapted by Atserias et al. '00 for monotone proofs.

Proof idea

- Let LPHP_n^{n+1} and RPHP_n^{n+1} be the LHS and RHS of PHP_n^{n+1}, respectively.
- We will use the following outline of a proof of PHP_n^{n+1}:

$$\begin{aligned} \mathrm{LPHP}_n^{n+1} &\overset{(1)}{\Longrightarrow} th_{n+1}^{n(n+1)}(p_{ij}) \\ &\overset{(2)}{\Longrightarrow} th_{n+1}^{n(n+1)}(p_{ji}) \\ &\overset{(3)}{\Longrightarrow} \mathrm{RPHP}_n^{n+1}. \end{aligned}$$

- This basic proof template was first used in Buss '87, and adapted by Atserias et al. '00 for monotone proofs.
- (1) and (3) have simple proofs with flows of constant length.

Proof idea

- Let LPHP_n^{n+1} and RPHP_n^{n+1} be the LHS and RHS of PHP_n^{n+1}, respectively.
- We will use the following outline of a proof of PHP_n^{n+1}:

$$\begin{aligned} \mathrm{LPHP}_n^{n+1} &\overset{(1)}{\Longrightarrow} th_{n+1}^{n(n+1)}(p_{ij}) \\ &\overset{(2)}{\Longrightarrow} th_{n+1}^{n(n+1)}(p_{ji}) \\ &\overset{(3)}{\Longrightarrow} \mathrm{RPHP}_n^{n+1}. \end{aligned}$$

- This basic proof template was first used in Buss '87, and adapted by Atserias et al. '00 for monotone proofs.
- (1) and (3) have simple proofs with flows of constant length.
  We focus on (2).

Proof idea

- Let LPHP_n^{n+1} and RPHP_n^{n+1} be the LHS and RHS of PHP_n^{n+1}, respectively.
- We will use the following outline of a proof of PHP_n^{n+1}:

$$\begin{aligned} \mathrm{LPHP}_n^{n+1} &\overset{(1)}{\Longrightarrow} th_{n+1}^{n(n+1)}(p_{ij}) \\ &\overset{(2)}{\Longrightarrow} th_{n+1}^{n(n+1)}(p_{ji}) \\ &\overset{(3)}{\Longrightarrow} \mathrm{RPHP}_n^{n+1}. \end{aligned}$$

- This basic proof template was first used in Buss '87, and adapted by Atserias et al. '00 for monotone proofs.
- (1) and (3) have simple proofs with flows of constant length.
  We focus on (2).
- In particular we formulate divide-and-conquer inductions to control the length of atomic flows.

Decomposition of matrix transposition

- The permutation of (2) corresponds to the transposition of a matrix.

Decomposition of matrix transposition

- The permutation of (2) corresponds to the transposition of a matrix.

Observation

If $\boldsymbol{B}$ and $\boldsymbol{C}$ are matrices of equal dimensions, then

$$\begin{pmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{pmatrix}^{\mathsf{T}} = \begin{pmatrix} \boldsymbol{A}^{\mathsf{T}} & \boldsymbol{C}^{\mathsf{T}} \\ \boldsymbol{B}^{\mathsf{T}} & \boldsymbol{D}^{\mathsf{T}} \end{pmatrix}.$$

Decomposition of matrix transposition

- The permutation of (2) corresponds to the transposition of a matrix.

Observation

If $\boldsymbol{B}$ *and* $\boldsymbol{C}$ *are matrices of equal dimensions, then*

$$\begin{pmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{pmatrix}^{\mathsf{T}} = \begin{pmatrix} \boldsymbol{A}^{\mathsf{T}} & \boldsymbol{C}^{\mathsf{T}} \\ \boldsymbol{B}^{\mathsf{T}} & \boldsymbol{D}^{\mathsf{T}} \end{pmatrix}.$$

- Attempting a divide-and-conquer we obtain

$$\begin{pmatrix} \boldsymbol{A} & \boldsymbol{B} \end{pmatrix}^{\mathsf{T}} = \begin{pmatrix} \boldsymbol{A}^{\mathsf{T}} \\ \boldsymbol{B}^{\mathsf{T}} \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} \boldsymbol{C} & \boldsymbol{D} \end{pmatrix}^{\mathsf{T}} = \begin{pmatrix} \boldsymbol{C}^{\mathsf{T}} \\ \boldsymbol{D}^{\mathsf{T}} \end{pmatrix}.$$

Decomposition of matrix transposition

- The permutation of (2) corresponds to the transposition of a matrix.

Observation

If $\boldsymbol{B}$ *and* $\boldsymbol{C}$ *are matrices of equal dimensions, then*

$$\begin{pmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{pmatrix}^{\mathsf{T}} = \begin{pmatrix} \boldsymbol{A}^{\mathsf{T}} & \boldsymbol{C}^{\mathsf{T}} \\ \boldsymbol{B}^{\mathsf{T}} & \boldsymbol{D}^{\mathsf{T}} \end{pmatrix}.$$

- Attempting a divide-and-conquer we obtain

$$\begin{pmatrix} \boldsymbol{A} & \boldsymbol{B} \end{pmatrix}^{\mathsf{T}} = \begin{pmatrix} \boldsymbol{A}^{\mathsf{T}} \\ \boldsymbol{B}^{\mathsf{T}} \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} \boldsymbol{C} & \boldsymbol{D} \end{pmatrix}^{\mathsf{T}} = \begin{pmatrix} \boldsymbol{C}^{\mathsf{T}} \\ \boldsymbol{D}^{\mathsf{T}} \end{pmatrix}.$$

- To achieve the full transposition from these, we need to interleave the rows of these two matrices.

Decomposition of matrix transposition

- Let $\boldsymbol{a} \odot \boldsymbol{b}$ denote the perfect interleaving of vectors $\boldsymbol{a}, \boldsymbol{b}$.

Decomposition of matrix transposition

- Let $\boldsymbol{a} \odot \boldsymbol{b}$ denote the perfect interleaving of vectors $\boldsymbol{a}, \boldsymbol{b}$.

Observation

$$(\boldsymbol{a}, \boldsymbol{b}) \odot (\boldsymbol{c}, \boldsymbol{d}) \quad = \quad (\boldsymbol{a} \odot \boldsymbol{c}, \boldsymbol{b} \odot \boldsymbol{d})$$

- $\rightsquigarrow$ a divide-and-conquer strategy for interleaving vectors.

Decomposition of matrix transposition

- Let $\boldsymbol{a} \odot \boldsymbol{b}$ denote the perfect interleaving of vectors $\boldsymbol{a}, \boldsymbol{b}$.

Observation

$$(\boldsymbol{a}, \boldsymbol{b}) \odot (\boldsymbol{c}, \boldsymbol{d}) \quad = \quad (\boldsymbol{a} \odot \boldsymbol{c}, \boldsymbol{b} \odot \boldsymbol{d})$$

- $\rightsquigarrow$ a divide-and-conquer strategy for interleaving vectors.

Lemma

There are KS*-derivations of* $th_k^n(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}) \to th_k^n(\boldsymbol{a}, \boldsymbol{c}, \boldsymbol{b}, \boldsymbol{d})$ *of size* $n^{O(\log n)}$.

Main results

Theorem

There are KS^+ *proofs of* $th_k^n(\boldsymbol{A}) \to th_k^n(\boldsymbol{A}^\intercal)$ *of size* $n^{O(\log n)}$ *and whose flows have length* $O(\log^2 n)$.

Main results

Theorem

There are KS^+ *proofs of* $th^n_k(\boldsymbol{A}) \to th^n_k(\boldsymbol{A}^\intercal)$ *of size* $n^{O(\log n)}$ *and whose flows have length* $O(\log^2 n)$.

(The $\log^2 n$ comes from the fact that we have used two divide-and-conquer inductions, one nested inside the other).

Main results

Theorem

There are KS^+ *proofs of* $th_k^n(\boldsymbol{A}) \to th_k^n(\boldsymbol{A}^\intercal)$ *of size* $n^{O(\log n)}$ *and whose flows have length* $O(\log^2 n)$.

(The $\log^2 n$ comes from the fact that we have used two divide-and-conquer inductions, one nested inside the other).

Corollary

There are KS *proofs of* $th_k^n(\boldsymbol{A}) \to th_k^n(\boldsymbol{A}^\intercal)$ *of size* $n^{O(\log^2 n)}$.

Main results

Theorem

There are KS^+ *proofs of* $th_k^n(\boldsymbol{A}) \to th_k^n(\boldsymbol{A}^\intercal)$ *of size* $n^{O(\log n)}$ *and whose flows have length* $O(\log^2 n)$.

(The $\log^2 n$ comes from the fact that we have used two divide-and-conquer inductions, one nested inside the other).

Corollary

There are KS *proofs of* $th_k^n(\boldsymbol{A}) \to th_k^n(\boldsymbol{A}^\intercal)$ *of size* $n^{O(\log^2 n)}$.

Corollary

There are KS *proofs of* PHP_n^m *of size quasipolynomial in* m *and* n.

Further results

Further results

From previously known lower bounds, we obtain the following:

Corollary

KS *is exponentially separated from:*

- *Cut-free sequent calculi, tree-like or dag-like.*
- *All variants of Resolution systems.*
- *Bounded-depth Hilbert-Frege systems.*

Further results

From previously known lower bounds, we obtain the following:

Corollary

KS *is exponentially separated from:*

- *Cut-free sequent calculi, tree-like or dag-like.*
- *All variants of Resolution systems.*
- *Bounded-depth Hilbert-Frege systems.*

In fact, our proof structure yields a new proof for the weak pigeonhole principle, via threshold approximators:

Theorem (D. '14)

For all $\varepsilon = \frac{1}{\log^c n}$, *there are monotone proofs of* $\mathrm{PHP}_n^{(1+\varepsilon)n}$ *of size* $n^{O(c)\log\log n}$.

Further results

From previously known lower bounds, we obtain the following:

Corollary

KS *is exponentially separated from:*

- *Cut-free sequent calculi, tree-like or dag-like.*
- *All variants of Resolution systems.*
- *Bounded-depth Hilbert-Frege systems.*

In fact, our proof structure yields a new proof for the weak pigeonhole principle, via threshold approximators:

Theorem (D. '14)

For all $\varepsilon = \frac{1}{\log^c n}$, *there are monotone proofs of* $\mathrm{PHP}_n^{(1+\varepsilon)n}$ *of size* $n^{O(c)\log\log n}$.

This is the first time developments in deep inference yielded improved bounds for mainstream proof complexity.

Outline

Bounding the depth of inference

What if we only allow inference up to bounded depth?

Bounding the depth of inference

What if we only allow inference up to bounded depth?

Theorem (D. '11)

1-KS$^+$ *polynomially simulates* KS$^+$

Bounding the depth of inference

What if we only allow inference up to bounded depth?

Theorem (D. '11)

1-KS$^+$ *polynomially simulates* KS$^+$ *and so quasipolynomially simulates* SKS.

Bounding the depth of inference

What if we only allow inference up to bounded depth?

Theorem (D. '11)

1-KS$^+$ *polynomially simulates* KS$^+$ *and so quasipolynomially simulates* SKS.

In fact, one can show the following:

Corollary

Dag-like cut-free sequent calculus with elimination rules,

$$\frac{\Gamma, \bot}{\Gamma} \qquad \frac{\Gamma, A \vee B}{\Gamma, A, B} \qquad \frac{\Gamma, A_1 \wedge A_2}{\Gamma, A_i}$$

quasipolynomially simulates Hilbert-Frege systems.

Bounding the depth of inference

What if we only allow inference up to bounded depth?

Theorem (D. '11)

1-KS$^+$ *polynomially simulates* KS$^+$ *and so quasipolynomially simulates* SKS.

In fact, one can show the following:

Corollary

Dag-like cut-free sequent calculus with elimination rules,

$$\frac{\Gamma, \bot}{\Gamma} \qquad \frac{\Gamma, A \vee B}{\Gamma, A, B} \qquad \frac{\Gamma, A_1 \wedge A_2}{\Gamma, A_i}$$

quasipolynomially simulates Hilbert-Frege systems.

Open problem

Does d-KS *(quasi)polynomially simulate* KS *for some* d*?*

Relative complexity of KS

How does KS compare with other systems?

Relative complexity of KS

How does KS compare with other systems?

Theorem (Various authors '00-'15)

KS *polynomially simulates truth tables.*

Relative complexity of KS

How does KS compare with other systems?

Theorem (Various authors '00-'15)

KS *polynomially simulates truth tables.*

Theorem (D. '12-'15)

KS *polynomially simulates*

- *Tree-like Resolution with clauses-as-sets.*
- *Dag-like Resolution with clauses-as-multisets.*

Relative complexity of KS

How does KS compare with other systems?

Theorem (Various authors '00-'15)

KS *polynomially simulates truth tables.*

Theorem (D. '12-'15)

KS *polynomially simulates*

- *Tree-like Resolution with clauses-as-sets.*
- *Dag-like Resolution with clauses-as-multisets.*

(The two proofs are quite different!)

Relative complexity of KS

How does KS compare with other systems?

Theorem (Various authors '00-'15)

KS *polynomially simulates truth tables.*

Theorem (D. '12-'15)

KS *polynomially simulates*

- *Tree-like Resolution with clauses-as-sets.*
- *Dag-like Resolution with clauses-as-multisets.*

(The two proofs are quite different!)

Open problem

KS *vs. bounded-depth Hilbert-Frege systems, in general.*

A uniform approach

A uniform approach

Bounded arithmetic

Propositional systems as nonuniform versions of weak theories of arithmetic.

A uniform approach

Bounded arithmetic
Propositional systems as nonuniform versions of weak theories of arithmetic.

E.g.

Theorem (Paris & Wilkie '81)

1. $I\Delta_0 \vdash \forall x.A(x) \Rightarrow$ *small bounded-depth proofs of* $\langle A(n)\rangle_{n\in\mathbb{N}}$.
2. $I\Delta_0 \vdash$ *"bounded-depth Hilbert-Frege is sound".*

A uniform approach

Bounded arithmetic
Propositional systems as nonuniform versions of weak theories of arithmetic.

E.g.

Theorem (Paris & Wilkie '81)

1. $I\Delta_0 \vdash \forall x.A(x) \Rightarrow$ *small bounded-depth proofs of* $\langle A(n)\rangle_{n\in\mathbb{N}}$.
2. $I\Delta_0 \vdash$ *"bounded-depth Hilbert-Frege is sound".*

Theorem (D. '16)

Intuitionistic U^1_2 *restricted to type-1 induction corresponds to* KS^+.

A uniform approach

Bounded arithmetic
Propositional systems as nonuniform versions of weak theories of arithmetic.

E.g.

Theorem (Paris & Wilkie '81)

1. $I\Delta_0 \vdash \forall x.A(x) \Rightarrow$ *small bounded-depth proofs of* $\langle A(n)\rangle_{n\in\mathbb{N}}$.
2. $I\Delta_0 \vdash$ *"bounded-depth Hilbert-Frege is sound".*

Theorem (D. '16)

Intuitionistic U^1_2 *restricted to type-1 induction corresponds to* KS^+.

Open problem

Find a theory corresponding to KS.

A uniform approach

Bounded arithmetic
Propositional systems as nonuniform versions of weak theories of arithmetic.

E.g.

Theorem (Paris & Wilkie '81)

1. $I\Delta_0 \vdash \forall x.A(x) \Rightarrow$ *small bounded-depth proofs of* $\langle A(n)\rangle_{n\in\mathbb{N}}$.
2. $I\Delta_0 \vdash$ *"bounded-depth Hilbert-Frege is sound".*

Theorem (D. '16)

Intuitionistic U^1_2 *restricted to type-1 induction corresponds to* KS^+.

Open problem

Find a theory corresponding to KS.

$\rightsquigarrow$ more simulations, superpolynomial lower bounds,... ?

Outline

Conclusions

Summary

- This is rich, technical and fun area of research!
- Many open problems yet to be solved.
- At the frontier of our knowledge on proof and computational complexity.

Conclusions

Summary

- This is rich, technical and fun area of research!
- Many open problems yet to be solved.
- At the frontier of our knowledge on proof and computational complexity.

Impact

- New tools for old problems in proof complexity.
- New approaches and connections in the area.

Conclusions

Summary

- This is rich, technical and fun area of research!
- Many open problems yet to be solved.
- At the frontier of our knowledge on proof and computational complexity.

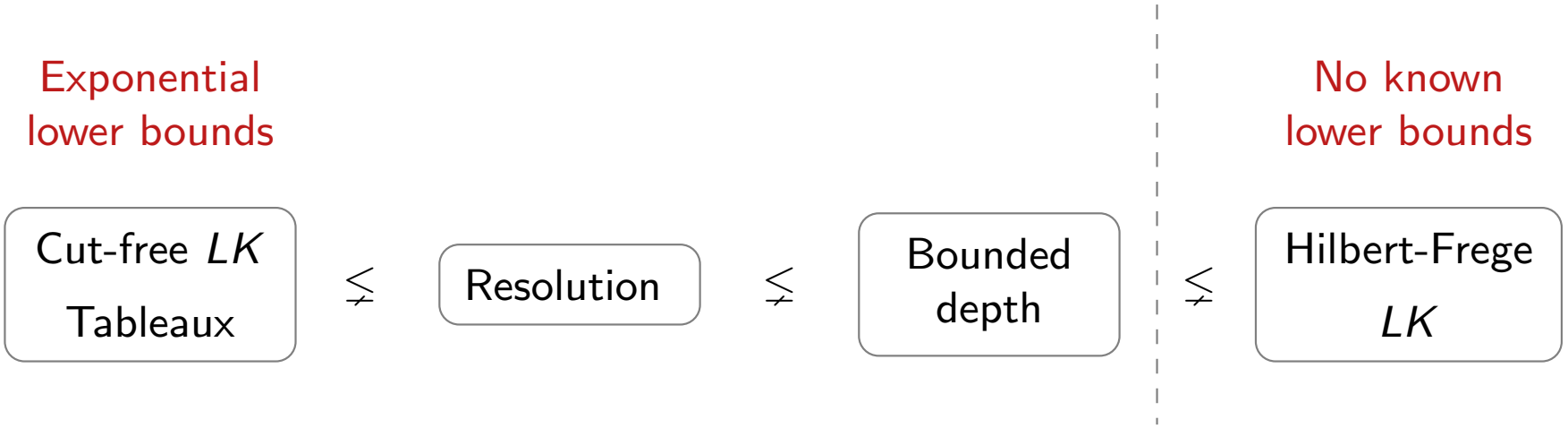
Impact

- New tools for old problems in proof complexity.
- New approaches and connections in the area.

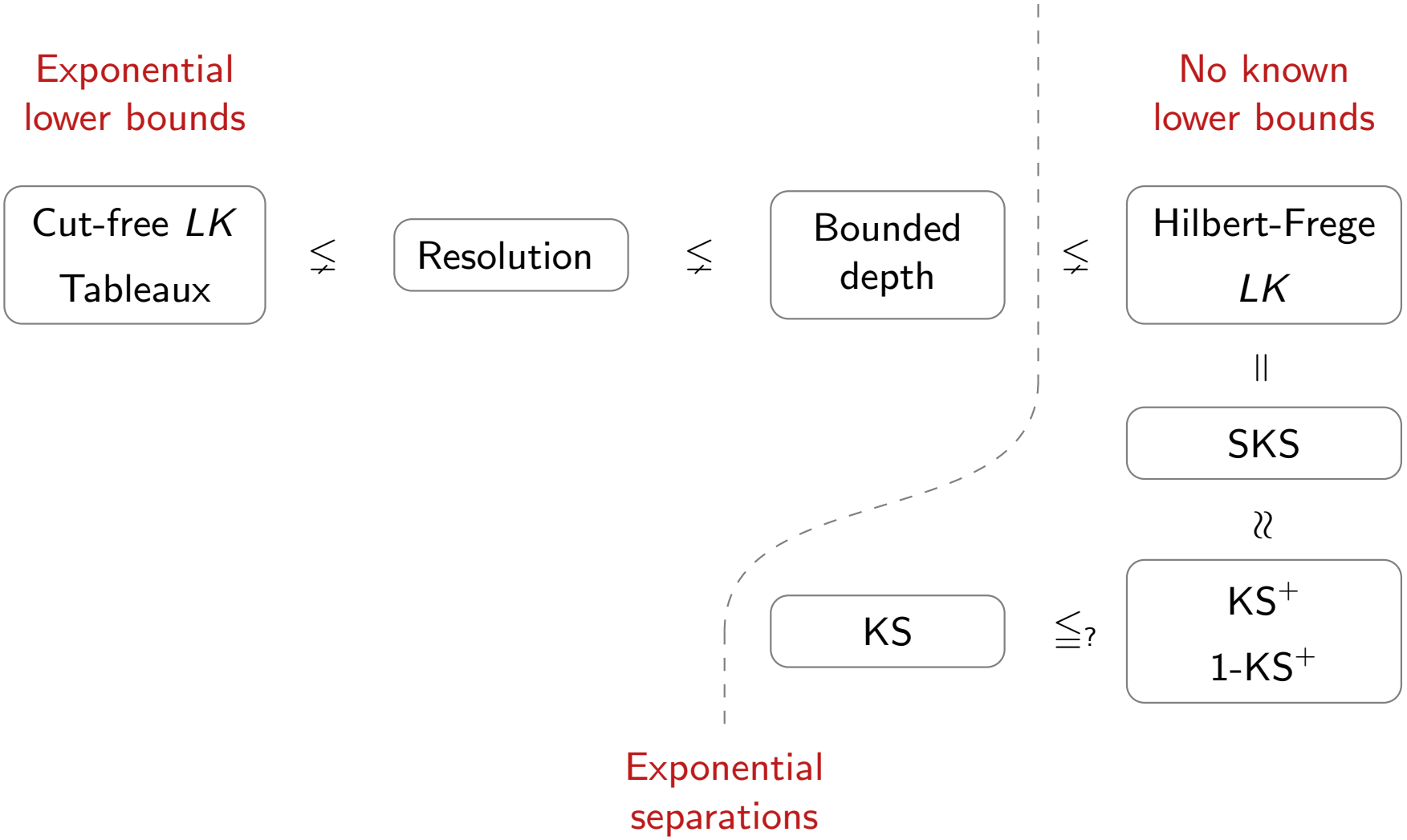
What lies ahead...

- Tighter integration with *structural* proof theory.
- Calibration with powerful tools from combinatorics.

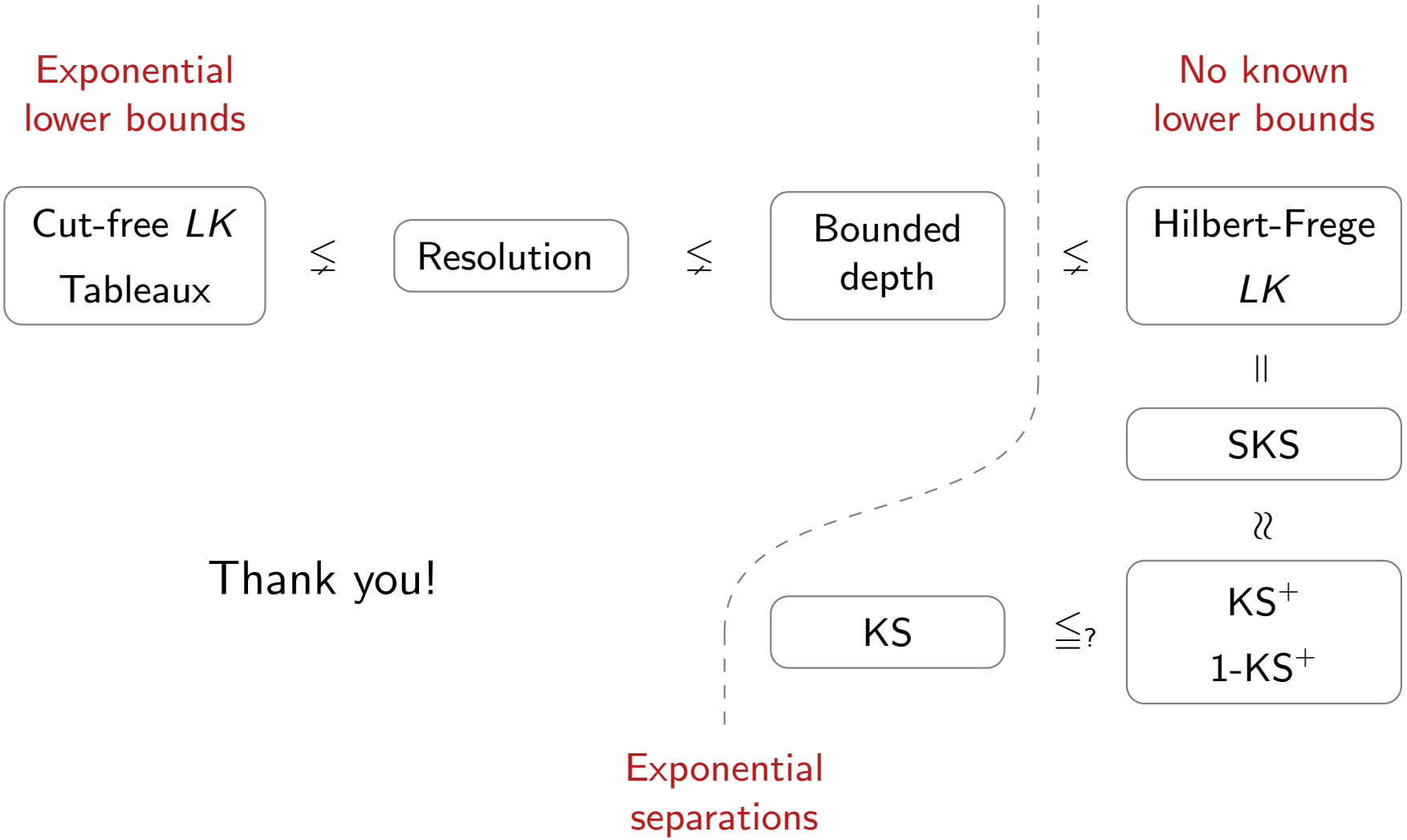
The current situation



The current situation



The current situation



Thank you!