Proof Complexity Lower Bounds from Graph Expansion and Combinatorial Games

Jakob Nordström

KTH Royal Institute of Technology
Stockholm, Sweden

IT University of Copenhagen
October 30, 2017

Based on joint work with Massimo Lauria and Mladen Mikša

The Satisfiability Problem (SAT)

$$(x \lor y) \land (x \lor \overline{y} \lor z) \land (\overline{x} \lor z) \land (\overline{y} \lor \overline{z}) \land (\overline{x} \lor \overline{z})$$

The Satisfiability Problem (SAT)

$$(x \lor y) \land (x \lor \overline{y} \lor z) \land (\overline{x} \lor z) \land (\overline{y} \lor \overline{z}) \land (\overline{x} \lor \overline{z})$$

- Variables should be set to true or false

The Satisfiability Problem (SAT)

$$(x \lor y) \land (x \lor \overline{y} \lor z) \land (\overline{x} \lor z) \land (\overline{y} \lor \overline{z}) \land (\overline{x} \lor \overline{z})$$

- Variables should be set to true or false
- Constraint $(x \lor \overline{y} \lor z)$: means x or z should be true or y false

The Satisfiability Problem (SAT)

$$(x \lor y) \land (x \lor \overline{y} \lor z) \land (\overline{x} \lor z) \land (\overline{y} \lor \overline{z}) \land (\overline{x} \lor \overline{z})$$

- Variables should be set to true or false
- Constraint $(x \lor \overline{y} \lor z)$: means x or z should be true or y false
- $\land$ means all constraints should hold simultaneously

The Satisfiability Problem (SAT)

$$(x \lor y) \land (x \lor \overline{y} \lor z) \land (\overline{x} \lor z) \land (\overline{y} \lor \overline{z}) \land (\overline{x} \lor \overline{z})$$

- Variables should be set to true or false
- Constraint $(x \lor \overline{y} \lor z)$: means x or z should be true or y false
- $\land$ means all constraints should hold simultaneously

Is there a truth value assignment satisfying all these conditions?
Or is it always the case that some constraint must fail to hold?

The Satisfiability Problem (SAT)

$$(x \lor y) \land (x \lor \overline{y} \lor z) \land (\overline{x} \lor z) \land (\overline{y} \lor \overline{z}) \land (\overline{x} \lor \overline{z})$$

- Variables should be set to true or false
- Constraint $(x \lor \overline{y} \lor z)$: means x or z should be true or y false
- $\land$ means all constraints should hold simultaneously

Is there a truth value assignment satisfying all these conditions?
Or is it always the case that some constraint must fail to hold?

1. Can this problem be solved efficiently?

The Satisfiability Problem (SAT)

$$(x \vee y) \wedge (x \vee \overline{y} \vee z) \wedge (\overline{x} \vee z) \wedge (\overline{y} \vee \overline{z}) \wedge (\overline{x} \vee \overline{z})$$

- Variables should be set to true or false
- Constraint $(x \vee \overline{y} \vee z)$: means x or z should be true or y false
- $\wedge$ means all constraints should hold simultaneously

Is there a truth value assignment satisfying all these conditions?
Or is it always the case that some constraint must fail to hold?

1. Can this problem be solved efficiently?
2. Is there an efficiently verifiable certificate for correct answer?

SAT and Proof Complexity

SAT, NP, and coNP

- SAT NP-complete [Coo71, Lev73], hence unlikely to be solvable efficiently worst-case
- Satisfiable formulas have small certificates (assignment)
- Unsatisfiable formulas don't, unless NP = coNP
  Starting point for proof complexity [CR79]

SAT and Proof Complexity

SAT, NP, and coNP

- SAT NP-complete [Coo71, Lev73], hence unlikely to be solvable efficiently worst-case
- Satisfiable formulas have small certificates (assignment)
- Unsatisfiable formulas don't, unless NP $=$ coNP
  Starting point for proof complexity [CR79]

Proof complexity

- Prove lower bounds on certificate size for increasingly stronger formal methods of reasoning ($\approx$"separation NP $\neq$ coNP in weak computational models")
- Analyze algorithms used in practice for SAT solving
- Quantify hardness/depth of different mathematical theorems

Proof Complexity and Expansion

- **General goal:** Prove that concrete proof systems cannot efficiently certify unsatisfiability of concrete CNF formulas
- **General theme:**

CNF formula $\mathcal{F}$ "expanding"

$\Downarrow$

Large proofs needed to refute $\mathcal{F}$

- Paradigm implemented for
  - resolution: well-developed machinery
  - polynomial calculus: very much less so

  (Will define these proof systems shortly)
- What "expanding" means is usually a formula-specific hack

A General Expansion Criterion for Hardness

Given CNF formula $\mathcal{F}$ over variables $\mathcal{V}$, build bipartite graph

- Left vertex set partition of clauses into $\mathcal{F} = \bigcup_{i=1}^{m} F_i$
- Right vertex set division of variables $\mathcal{V} = \bigcup_{j=1}^{n} V_j$
- Edge (F_i, V_j) if $Vars(F_i) \cap V_j \neq \emptyset$

A General Expansion Criterion for Hardness

Given CNF formula $\mathcal{F}$ over variables $\mathcal{V}$, build bipartite graph

- Left vertex set partition of clauses into $\mathcal{F} = \bigcup_{i=1}^{m} F_i$
- Right vertex set division of variables $\mathcal{V} = \bigcup_{j=1}^{n} V_j$
- Edge (F_i, V_j) if $Vars(F_i) \cap V_j \neq \emptyset$

Lower bound on proof size if

1. Bipartite graph is an expander (very well-connected)
2. We can win the edge game on every edge (F_i, V_j)

A General Expansion Criterion for Hardness

Given CNF formula $\mathcal{F}$ over variables $\mathcal{V}$, build bipartite graph

- Left vertex set partition of clauses into $\mathcal{F} = \bigcup_{i=1}^{m} F_i$
- Right vertex set division of variables $\mathcal{V} = \bigcup_{j=1}^{n} V_j$
- Edge (F_i, V_j) if $Vars(F_i) \cap V_j \neq \emptyset$

Lower bound on proof size if

1. Bipartite graph is an expander (very well-connected)
2. We can win the edge game on every edge (F_i, V_j)

Edge game on (F_i, V_j)

- Adversary assigns all variables $\mathcal{V} \setminus V_j$
- We assign V_j
- We win if F_i true

Main Message

Edge game on (F_i, V_j)

- Adversary assigns all variables $\mathcal{V} \setminus V_j$
- We assign V_j
- We win if F_i true

Main Message

Edge game on (F_i, V_j)

- Adversary assigns all variables $\mathcal{V} \setminus V_j$
- We assign V_j
- We win if F_i true

Who goes first?

- Adversary has to start $\Rightarrow$ resolution lower bound
- We have to start $\Rightarrow$ polynomial calculus lower bound

Main Message

Edge game on (F_i, V_j)

- Adversary assigns all variables $\mathcal{V} \setminus V_j$
- We assign V_j
- We win if F_i true

Who goes first?

- Adversary has to start $\Rightarrow$ resolution lower bound
- We have to start $\Rightarrow$ polynomial calculus lower bound

Consequences

- Extends techniques in [BW01] and [AR03]
- Unifies many previous lower bounds
- And yields some new ones

Outline

Some Notation and Terminology

- Literal a: variable x or its negation $\overline{x}$
- Clause $C = a_1 \lor \cdots \lor a_k$: disjunction of literals
  (Consider as sets, so no repetitions and order irrelevant)
- CNF formula $\mathcal{F} = C_1 \land \cdots \land C_m$: conjunction of clauses
- k-CNF formula: CNF formula with clauses of size $\leq k$
  $k = \mathcal{O}(1)$ constant in this talk
- $\mathit{true} = 1$; $\mathit{false} = 0$
- M = size of formula = # literals (≈ # clauses for k-CNF)
- N = # variables $\leq M$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\bot$ derived

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\perp$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1. $x \lor y$
2. $x \lor \overline{y} \lor z$
3. $\overline{x} \lor z$
4. $\overline{y} \lor \overline{z}$
5. $\overline{x} \lor \overline{z}$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\bot$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2,4)$
7.	x	Res$(1,6)$
8.	$\overline{x}$	Res$(3,5)$
9.	$\bot$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\bot$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1.	$x \lor y$	Axiom
2.	$\boldsymbol{x \lor \overline{y} \lor z}$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\boldsymbol{\overline{y} \lor \overline{z}}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2, 4)$
7.	x	Res$(1, 6)$
8.	$\overline{x}$	Res$(3, 5)$
9.	$\bot$	Res$(7, 8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\perp$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1.	$x \lor y$	Axiom
2.	$\boldsymbol{x \lor \overline{y} \lor z}$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\boldsymbol{\overline{y} \lor \overline{z}}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$\boldsymbol{x \lor \overline{y}}$	Res$(2, 4)$
7.	x	Res$(1, 6)$
8.	$\overline{x}$	Res$(3, 5)$
9.	$\perp$	Res$(7, 8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \vee x \qquad D \vee \overline{x}}{C \vee D}$$

Refutation ends when empty clause $\bot$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1.	$x \vee y$	Axiom
2.	$x \vee \overline{y} \vee z$	Axiom
3.	$\overline{x} \vee z$	Axiom
4.	$\overline{y} \vee \overline{z}$	Axiom
5.	$\overline{x} \vee \overline{z}$	Axiom
6.	$\boldsymbol{x \vee \overline{y}}$	Res$(2,4)$
7.	x	Res$(1,6)$
8.	$\overline{x}$	Res$(3,5)$
9.	$\bot$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\bot$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1.	$\boldsymbol{x \lor y}$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$\boldsymbol{x \lor \overline{y}}$	Res$(2, 4)$
7.	x	Res$(1, 6)$
8.	$\overline{x}$	Res$(3, 5)$
9.	$\bot$	Res$(7, 8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \vee x \qquad D \vee \overline{x}}{C \vee D}$$

Refutation ends when empty clause $\bot$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1.	$\boldsymbol{x \vee y}$	Axiom
2.	$x \vee \overline{y} \vee z$	Axiom
3.	$\overline{x} \vee z$	Axiom
4.	$\overline{y} \vee \overline{z}$	Axiom
5.	$\overline{x} \vee \overline{z}$	Axiom
6.	$\boldsymbol{x \vee \overline{y}}$	Res$(2, 4)$
7.	$\boldsymbol{x}$	Res$(1, 6)$
8.	$\overline{x}$	Res$(3, 5)$
9.	$\bot$	Res$(7, 8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\perp$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2,4)$
7.	$\boldsymbol{x}$	Res$(1,6)$
8.	$\overline{x}$	Res$(3,5)$
9.	$\perp$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \vee x \qquad D \vee \overline{x}}{C \vee D}$$

Refutation ends when empty clause $\perp$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1.	$x \vee y$	Axiom
2.	$x \vee \overline{y} \vee z$	Axiom
3.	$\overline{x} \vee z$	Axiom
4.	$\overline{y} \vee \overline{z}$	Axiom
5.	$\overline{x} \vee \overline{z}$	Axiom
6.	$x \vee \overline{y}$	Res$(2, 4)$
7.	x	Res$(1, 6)$
8.	$\overline{x}$	Res$(3, 5)$
9.	$\perp$	Res$(7, 8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \vee x \qquad D \vee \overline{x}}{C \vee D}$$

Refutation ends when empty clause $\perp$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1.	$x \vee y$	Axiom
2.	$x \vee \overline{y} \vee z$	Axiom
3.	$\overline{x} \vee z$	Axiom
4.	$\overline{y} \vee \overline{z}$	Axiom
5.	$\overline{x} \vee \overline{z}$	Axiom
6.	$x \vee \overline{y}$	Res$(2,4)$
7.	x	Res$(1,6)$
8.	$\overline{x}$	Res$(3,5)$
9.	$\perp$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \vee x \qquad D \vee \overline{x}}{C \vee D}$$

Refutation ends when empty clause $\bot$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1.	$x \vee y$	Axiom
2.	$x \vee \overline{y} \vee z$	Axiom
3.	$\overline{x} \vee z$	Axiom
4.	$\overline{y} \vee \overline{z}$	Axiom
5.	$\overline{x} \vee \overline{z}$	Axiom
6.	$x \vee \overline{y}$	Res$(2,4)$
7.	x	Res$(1,6)$
8.	$\overline{x}$	Res$(3,5)$
9.	$\bot$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\bot$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2,4)$
7.	$\boldsymbol{x}$	Res$(1,6)$
8.	$\boldsymbol{\overline{x}}$	Res$(3,5)$
9.	$\bot$	Res$(7,8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\perp$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2, 4)$
7.	$\boldsymbol{x}$	Res$(1, 6)$
8.	$\boldsymbol{\overline{x}}$	Res$(3, 5)$
9.	$\perp$	Res$(7, 8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\bot$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

1.	$x \lor y$	Axiom
2.	$x \lor \overline{y} \lor z$	Axiom
3.	$\overline{x} \lor z$	Axiom
4.	$\overline{y} \lor \overline{z}$	Axiom
5.	$\overline{x} \lor \overline{z}$	Axiom
6.	$x \lor \overline{y}$	Res$(2, 4)$
7.	x	Res$(1, 6)$
8.	$\overline{x}$	Res$(3, 5)$
9.	$\bot$	Res$(7, 8)$

The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

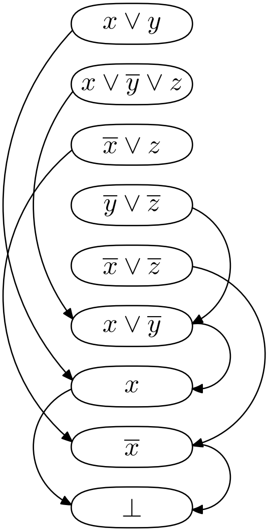
Derive new clauses by resolution rule

$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\bot$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph



The Resolution Proof System

Goal: refute **unsatisfiable** CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

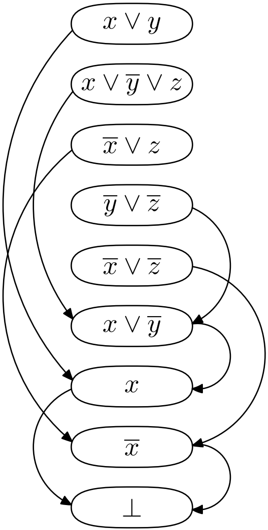
$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\bot$ derived

Can represent refutation as

- annotated list or
- directed acyclic graph

Tree-like resolution if DAG is tree



Resolution Size/Length

Size/length $=$ $\#$ clauses in refutation [9 in our example]

Most fundamental measure in proof complexity

Never worse than $\exp(\mathcal{O}(N))$

Matching $\exp(\Omega(M))$ lower bounds known

(Recall $N = \#$ variables $\leq$ formula size $= M$)

Examples of Hard Formulas w.r.t Resolution Size (1/3)

Pigeonhole principle (PHP) [Hak85]
"$n+1$ pigeons don't fit into n holes"

Variables $p_{i,j}$ = "pigeon i goes into hole j"

$p_{i,1} \lor p_{i,2} \lor \cdots \lor p_{i,n}$	every pigeon i gets a hole
$\overline{p}_{i,j} \lor \overline{p}_{i',j}$	no hole j gets two pigeons $i \neq i'$

Can also add "functionality" and "onto" axioms

$\overline{p}_{i,j} \lor \overline{p}_{i,j'}$	no pigeon i gets two holes $j \neq j'$
$p_{1,j} \lor p_{2,j} \lor \cdots \lor p_{n+1,j}$	every hole j gets a pigeon

Examples of Hard Formulas w.r.t Resolution Size (1/3)

Pigeonhole principle (PHP) [Hak85]
"$n+1$ pigeons don't fit into n holes"

Variables $p_{i,j}$ = "pigeon i goes into hole j"

$p_{i,1} \lor p_{i,2} \lor \cdots \lor p_{i,n}$	every pigeon i gets a hole
$\overline{p}_{i,j} \lor \overline{p}_{i',j}$	no hole j gets two pigeons $i \neq i'$

Can also add "functionality" and "onto" axioms

$\overline{p}_{i,j} \lor \overline{p}_{i,j'}$	no pigeon i gets two holes $j \neq j'$
$p_{1,j} \lor p_{2,j} \lor \cdots \lor p_{n+1,j}$	every hole j gets a pigeon

Even onto functional PHP formulas are hard for resolution
"Resolution cannot count"

Examples of Hard Formulas w.r.t Resolution Size (1/3)

Pigeonhole principle (PHP) [Hak85]
"$n+1$ pigeons don't fit into n holes"

Variables $p_{i,j}$ = "pigeon i goes into hole j"

$p_{i,1} \lor p_{i,2} \lor \cdots \lor p_{i,n}$	every pigeon i gets a hole
$\overline{p}_{i,j} \lor \overline{p}_{i',j}$	no hole j gets two pigeons $i \neq i'$

Can also add "functionality" and "onto" axioms

$\overline{p}_{i,j} \lor \overline{p}_{i,j'}$	no pigeon i gets two holes $j \neq j'$
$p_{1,j} \lor p_{2,j} \lor \cdots \lor p_{n+1,j}$	every hole j gets a pigeon

Even onto functional PHP formulas are hard for resolution
"Resolution cannot count"

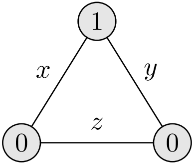
But only lower bound $\exp\left(\Omega\left(\sqrt[3]{M}\right)\right)$ in terms of formula size

Examples of Hard Formulas w.r.t Resolution Size (2/3)

Tseitin formulas [Urq87]
"Sum of degrees of vertices in graph is even"

Variables = edges (in undirected graph of bounded degree)

- Label every vertex 0/1 so that sum of labels odd
- Write CNF requiring parity of # true incident edges = label



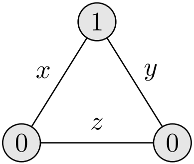
$$
\begin{array}{ll}
(x \lor y) & \land\ (\overline{x} \lor z) \\
\land\ (\overline{x} \lor \overline{y}) & \land\ (y \lor \overline{z}) \\
\land\ (x \lor \overline{z}) & \land\ (\overline{y} \lor z)
\end{array}
$$

Examples of Hard Formulas w.r.t Resolution Size (2/3)

Tseitin formulas [Urq87]
"Sum of degrees of vertices in graph is even"

Variables = edges (in undirected graph of bounded degree)

- Label every vertex 0/1 so that sum of labels odd
- Write CNF requiring parity of # true incident edges = label



$$
\begin{array}{ll}
(x \lor y) & \land (\overline{x} \lor z) \\
\land (\overline{x} \lor \overline{y}) & \land (y \lor \overline{z}) \\
\land (x \lor \overline{z}) & \land (\overline{y} \lor z)
\end{array}
$$

Requires size $\exp(\Omega(M))$ on bounded-degree edge expanders
"Resolution cannot count $\bmod 2$"

Examples of Hard Formulas w.r.t Resolution Size (3/3)

Random k-CNF formulas [CS88, BKPS02]
Δn randomly sampled k-clauses over n variables

($\Delta \gtrsim 4.5$ sufficient to get unsatisfiable 3-CNF almost surely)

Again lower bound $\exp(\Omega(M))$

Examples of Hard Formulas w.r.t Resolution Size (3/3)

Random k-CNF formulas [CS88, BKPS02]
Δn randomly sampled k-clauses over n variables

($\Delta \gtrsim 4.5$ sufficient to get unsatisfiable 3-CNF almost surely)

Again lower bound $\exp(\Omega(M))$

And more...

- k-colourability [BCMM05]
- Independent sets and vertex covers [BIS07]
- Subset cardinality formulas [Spe10, VS10, MN14]
- Et cetera...

Resolution Width

Width = size of largest clause in refutation (always $\leq N$)

Resolution Width

Width = size of largest clause in refutation (always $\leq N$)

Width upper bound $\Rightarrow$ size upper bound

Proof: at most $(2N)^{\text{width}}$ distinct clauses
(And this counting argument is essentially tight [ALN16])

Resolution Width

Width $=$ size of largest clause in refutation (always $\leq N$)

Width upper bound $\Rightarrow$ size upper bound

Proof: at most $(2N)^{\text{width}}$ distinct clauses
(And this counting argument is essentially tight [ALN16])

Width lower bound $\Rightarrow$ size lower bound

Much less obvious. . .

Width Lower Bounds Imply Size Lower Bounds

Theorem ([BW01])

For k-CNF formula over N variables

$$\textit{proof size} \geq \exp\left(\Omega\left(\frac{(\textit{proof width})^2}{N}\right)\right)$$

Width Lower Bounds Imply Size Lower Bounds

Theorem ([BW01])

For k-CNF formula over N variables

$$\textit{proof size} \geq \exp\left(\Omega\left(\frac{(\textit{proof width})^2}{N}\right)\right)$$

Yields superpolynomial size bounds for width $\omega\left(\sqrt{N \log N}\right)$

Almost all known lower bounds on size derivable via width

Width Lower Bounds Imply Size Lower Bounds

Theorem ([BW01])

For k-CNF formula over N variables

$$\textit{proof size} \geq \exp\left(\Omega\left(\frac{(\textit{proof width})^2}{N}\right)\right)$$

Yields superpolynomial size bounds for width $\omega\left(\sqrt{N \log N}\right)$
Almost all known lower bounds on size derivable via width

For tree-like resolution have proof size $\geq 2^{\text{width}}$ [BW01]

General resolution: width up to $\mathcal{O}\left(\sqrt{N \log N}\right)$ implies no size lower bounds — possible to tighten analysis? **No!**

Optimality of the Size-Width Lower Bound

Ordering principles [Stå96, BG01]
"Every (partially) ordered set $\{e_1, \ldots, e_n\}$ has minimal element"

Variables $x_{i,j} =$ "$e_i < e_j$"

$\overline{x}_{i,j} \lor \overline{x}_{j,i}$	anti-symmetry; not both $e_i < e_j$ and $e_j < e_i$
$\overline{x}_{i,j} \lor \overline{x}_{j,k} \lor x_{i,k}$	transitivity; $e_i < e_j$ and $e_j < e_k$ implies $e_i < e_k$
$\bigvee_{1 \leq i \leq n,\, i \neq j} x_{i,j}$	e_j is not a minimal element

Optimality of the Size-Width Lower Bound

Ordering principles [Stå96, BG01]
"Every (partially) ordered set $\{e_1, \ldots, e_n\}$ has minimal element"

Variables $x_{i,j} =$ "$e_i < e_j$"

$\overline{x}_{i,j} \lor \overline{x}_{j,i}$	anti-symmetry; not both $e_i < e_j$ and $e_j < e_i$
$\overline{x}_{i,j} \lor \overline{x}_{j,k} \lor x_{i,k}$	transitivity; $e_i < e_j$ and $e_j < e_k$ implies $e_i < e_k$
$\bigvee_{1 \leq i \leq n,\, i \neq j} x_{i,j}$	e_j is not a minimal element

Refutable in resolution in size $\mathcal{O}\left(N^{3/2}\right) = \mathcal{O}(M)$
Requires resolution width $\Omega\left(\sqrt{N}\right)$

Optimality of the Size-Width Lower Bound

Ordering principles [Stå96, BG01]
"Every (partially) ordered set $\{e_1, \ldots, e_n\}$ has minimal element"

Variables $x_{i,j} =$ "$e_i < e_j$"

$\overline{x}_{i,j} \vee \overline{x}_{j,i}$	anti-symmetry; not both $e_i < e_j$ and $e_j < e_i$
$\overline{x}_{i,j} \vee \overline{x}_{j,k} \vee x_{i,k}$	transitivity; $e_i < e_j$ and $e_j < e_k$ implies $e_i < e_k$
$\bigvee_{1 \leq i \leq n,\, i \neq j} x_{i,j}$	e_j is not a minimal element

Refutable in resolution in size $\mathcal{O}\big(N^{3/2}\big) = \mathcal{O}(M)$
Requires resolution width $\Omega\big(\sqrt{N}\big)$

But initial clauses have width $\Omega(n) = \Omega\big(\sqrt{N}\big)$ — a bit more work needed to make the width lower bound meaningful. . .

Conversion to k-CNF "Graph Versions" of Formulas

- Need bounded-width CNFs to use lower bound in [BW01]
- But PHP and ordering principle formulas have wide clauses
- **Solution:** Restrict formulas to bounded-degree graphs

Conversion to k-CNF "Graph Versions" of Formulas

- Need bounded-width CNFs to use lower bound in [BW01]
- But PHP and ordering principle formulas have wide clauses
- **Solution:** Restrict formulas to bounded-degree graphs

For (onto functional) PHP, pigeons can fly only to neighbour holes:

$\bigvee_{j \in \mathcal{N}(i)} p_{i,j}$	pigeon i goes into hole in $\mathcal{N}(i)$
$\bigvee_{i \in \mathcal{N}(j)} p_{i,j}$	hole j gets pigeon from $\mathcal{N}(j)$

For ordering principle, non-minimality only witnessed by neighbours:

$\bigvee_{i \in \mathcal{N}(j)} x_{i,j}$	some e_i for $i \in \mathcal{N}(j)$ shows e_j not minimal

Conversion to k-CNF "Graph Versions" of Formulas

- Need bounded-width CNFs to use lower bound in [BW01]
- But PHP and ordering principle formulas have wide clauses
- **Solution:** Restrict formulas to bounded-degree graphs

For (onto functional) PHP, pigeons can fly only to neighbour holes:

$\bigvee_{j \in \mathcal{N}(i)} p_{i,j}$	pigeon i goes into hole in $\mathcal{N}(i)$
$\bigvee_{i \in \mathcal{N}(j)} p_{i,j}$	hole j gets pigeon from $\mathcal{N}(j)$

For ordering principle, non-minimality only witnessed by neighbours:

$\bigvee_{i \in \mathcal{N}(j)} x_{i,j}$	some e_i for $i \in \mathcal{N}(j)$ shows e_j not minimal

- Now strong width lower bounds $\Rightarrow$ strong size lower bounds
- And size lower bounds hold for original, unrestricted formulas

Polynomial Calculus (PC)

From [CEI96]; with adjustment in [ABRW02]

Clauses interpreted as polynomial equations over field $\mathbb{F}$

Example: $x \vee y \vee \overline{z}$ gets translated to $\overline{x}\,\overline{y}z = 0$

Polynomial Calculus (PC)

From [CEI96]; with adjustment in [ABRW02]

Clauses interpreted as polynomial equations over field $\mathbb{F}$

Example: $x \vee y \vee \overline{z}$ gets translated to $\overline{x}\,\overline{y}z = 0$

Derivation rules

Boolean axioms $\dfrac{}{x^2 - x = 0}$ Negation $\dfrac{}{x + \overline{x} = 1}$

Linear combination $\dfrac{p = 0 \quad q = 0}{\alpha p + \beta q = 0}$ Multiplication $\dfrac{p = 0}{xp = 0}$

Goal: Derive $1 = 0 \Leftrightarrow$ no common root $\Leftrightarrow$ formula unsatisfiable

Formalizes Gröbner basis computation

Polynomial Calculus Size and Degree

Clauses turn into monomials

Write out all polynomials as sums of monomials

W.l.o.g. all polynomials multilinear (because of Boolean axioms)

Polynomial Calculus Size and Degree

Clauses turn into monomials
Write out all polynomials as sums of monomials
W.l.o.g. all polynomials multilinear (because of Boolean axioms)

Size — analogue of resolution length/size
total # monomials in refutation counted with repetitions

Degree — analogue of resolution width
largest degree of monomial in refutation

Polynomial Calculus Strictly Stronger than Resolution

Polynomial calculus simulates resolution efficiently

- Can mimic resolution refutation step by step
- Essentially no increase in length/size or width/degree
- Hence worst-case upper bounds for resolution carry over

Polynomial Calculus Strictly Stronger than Resolution

Polynomial calculus simulates resolution efficiently

- Can mimic resolution refutation step by step
- Essentially no increase in length/size or width/degree
- Hence worst-case upper bounds for resolution carry over

Polynomial calculus strictly stronger w.r.t. size and degree

- Tseitin formulas (over $\mathrm{GF}(2)$ can do Gaussian elimination)
- Onto functional pigeonhole principle (over any field) [Rii93]
- Also other examples

Size vs. Degree

- Degree upper bound ⇒ size upper bound [CEI96]
  Similar to resolution bound; argument a bit more involved
  Again essentially tight by [ALN16]

Size vs. Degree

- Degree upper bound ⇒ size upper bound [CEI96]
  Similar to resolution bound; argument a bit more involved
  Again essentially tight by [ALN16]
- Degree lower bound ⇒ size lower bound [IPS99]
  Precursor of [BW01] — can do same proof to get exactly same bound

Size vs. Degree

- Degree upper bound ⇒ size upper bound [CEI96]
  Similar to resolution bound; argument a bit more involved
  Again essentially tight by [ALN16]
- Degree lower bound ⇒ size lower bound [IPS99]
  Precursor of [BW01] — can do same proof to get exactly same bound
- Size-degree bound essentially optimal [GL10]
  Example: same ordering principle formulas

Size vs. Degree

- Degree upper bound ⇒ size upper bound [CEI96]
  Similar to resolution bound; argument a bit more involved
  Again essentially tight by [ALN16]
- Degree lower bound ⇒ size lower bound [IPS99]
  Precursor of [BW01] — can do same proof to get exactly same bound
- Size-degree bound essentially optimal [GL10]
  Example: same ordering principle formulas
- Most size lower bounds for polynomial calculus derived via degree lower bounds, but machinery much less developed

Size vs. Degree

- Degree upper bound ⇒ size upper bound [CEI96]
  Similar to resolution bound; argument a bit more involved
  Again essentially tight by [ALN16]
- Degree lower bound ⇒ size lower bound [IPS99]
  Precursor of [BW01] — can do same proof to get exactly same bound
- Size-degree bound essentially optimal [GL10]
  Example: same ordering principle formulas
- Most size lower bounds for polynomial calculus derived via degree lower bounds, but machinery much less developed
- **Examples of open problems:**
  - Hardness of functional PHP and onto PHP formulas?
  - Hardness of k-colouring formulas?

Lower Bounds via Graph Expansion

Standard approach:

Lower bounds from expansion

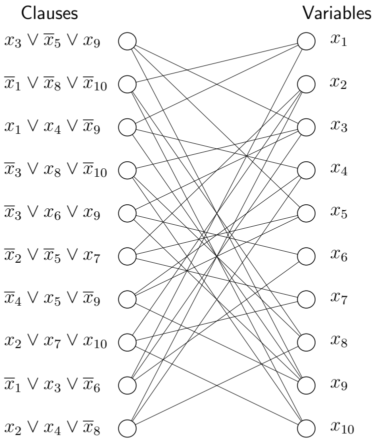
Simplest example is the clause-variable incidence graph (CVIG)

Lower Bounds via Graph Expansion

Standard approach:

Lower bounds from expansion

Simplest example is the clause-variable incidence graph (CVIG)

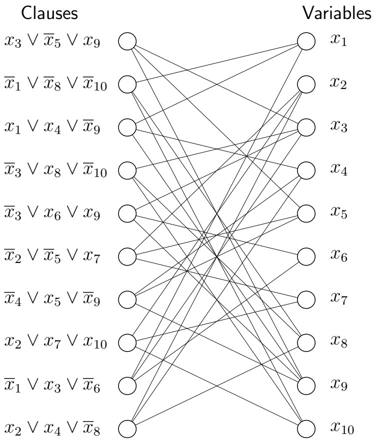


Lower Bounds via Graph Expansion

Standard approach:
Lower bounds from expansion

Simplest example is the clause-variable incidence graph (CVIG)

Boundary expansion:
Subsets of left vertices have many unique right neighbours

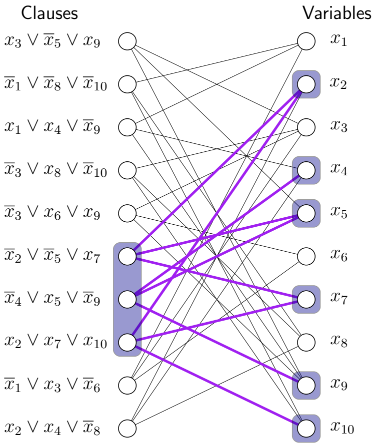


Lower Bounds via Graph Expansion

Standard approach:
Lower bounds from expansion

Simplest example is the clause-variable incidence graph (CVIG)

Boundary expansion:
Subsets of left vertices have many unique right neighbours



Lower Bounds via Graph Expansion

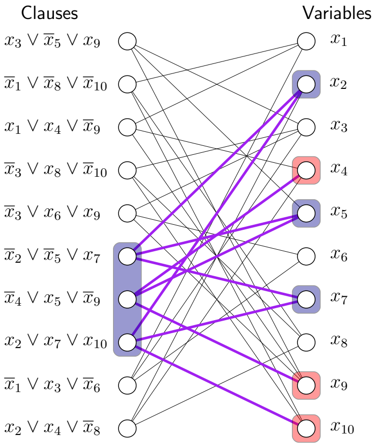
Standard approach:

Lower bounds from expansion

Simplest example is the clause-variable incidence graph (CVIG)

Boundary expansion:

Subsets of left vertices have many unique right neighbours



Lower Bounds via Graph Expansion

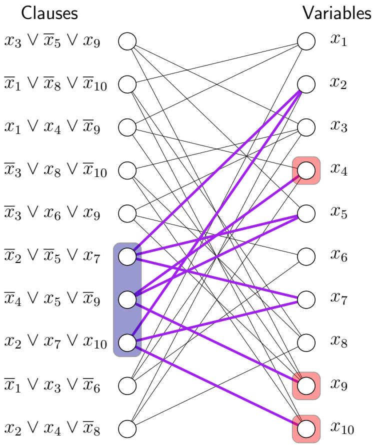
Standard approach:

Lower bounds from expansion

Simplest example is the clause-variable incidence graph (CVIG)

Boundary expansion:

Subsets of left vertices have many unique right neighbours



Lower Bounds via Graph Expansion

Standard approach:

Lower bounds from expansion

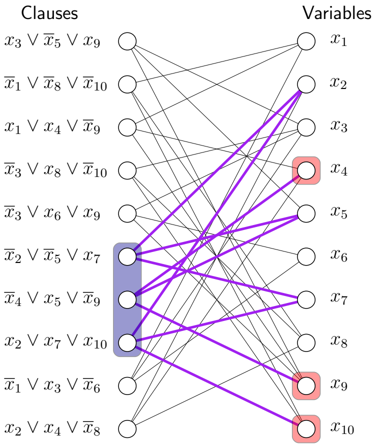
Simplest example is the clause-variable incidence graph (CVIG)

Boundary expansion:

Subsets of left vertices have many unique right neighbours

Problem:

CVIG often loses expansion of combinatorial problem



Lower Bounds via Graph Expansion

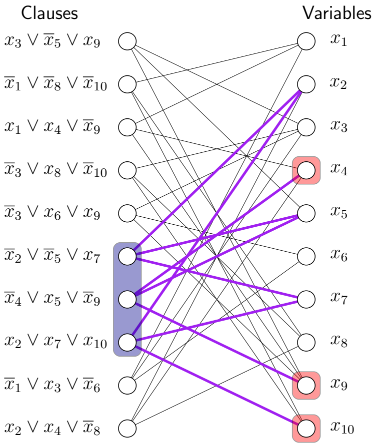
Standard approach:
Lower bounds from expansion

Simplest example is the clause-variable incidence graph (CVIG)

Boundary expansion:
Subsets of left vertices have many unique right neighbours

Problem:
CVIG often loses expansion of combinatorial problem

Need graph capturing combinatorial structure!



Generalized Incidence Graphs for CNF Formulas

Given CNF formula $\mathcal{F}$ over variables $\mathcal{V}$

- Partition clauses into $\mathcal{F} = E \cup \bigcup_{i=1}^{m} F_i$ (for E satisifiable)
- Divide variables into $\mathcal{V} = \bigcup_{j=1}^{n} V_j$ — **not** always partition
- Overlap ℓ: Any x appears in $\leq \ell$ different V_j

Generalized Incidence Graphs for CNF Formulas

Given CNF formula $\mathcal{F}$ over variables $\mathcal{V}$

- Partition clauses into $\mathcal{F} = E \cup \bigcup_{i=1}^{m} F_i$ (for E satisifiable)
- Divide variables into $\mathcal{V} = \bigcup_{j=1}^{n} V_j$ — **not** always partition
- Overlap ℓ: Any x appears in $\leq \ell$ different V_j

Build bipartite $(\mathcal{U}, \mathcal{V})_E$-graph $\mathcal{G}$

- Left vertices $\mathcal{U} = \{F_1, \ldots, F_m\}$
- Right vertices $\mathcal{V} = \{V_1, \ldots, V_n\}$
- Edge (F_i, V_j) if $\mathit{Vars}(F_i) \cap V_j \neq \emptyset$

The Resolution Edge Game

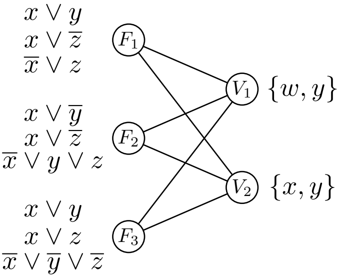
Resolution edge game on (F_i, V_j) w.r.t. "filtering set" E

- Adversary choses any total assignment α such that $\alpha(E) = 1$
- We can modify α on V_j to get α'
- We win if $\alpha'(F_i \wedge E) = 1$

The Resolution Edge Game

Resolution edge game on (F_i, V_j) w.r.t. "filtering set" E

- Adversary choses any total assignment α such that $\alpha(E) = 1$
- We can modify α on V_j to get α'
- We win if $\alpha'(F_i \wedge E) = 1$

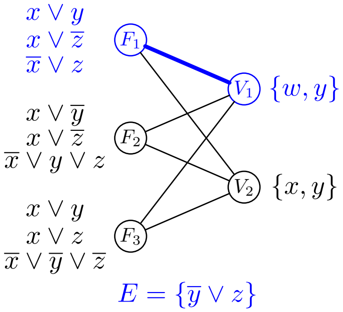


$$E = \{\overline{y} \vee z\}$$

The Resolution Edge Game

Resolution edge game on (F_i, V_j) w.r.t. "filtering set" E

- Adversary choses any total assignment α such that $\alpha(E) = 1$
- We can modify α on V_j to get α'
- We win if $\alpha'(F_i \wedge E) = 1$

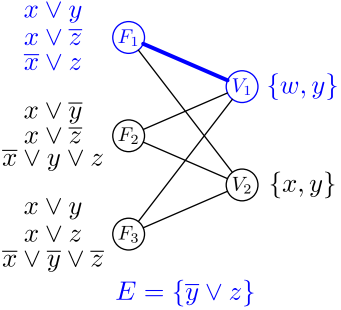


Edge game on (F_1, V_1) w.r.t. E

The Resolution Edge Game

Resolution edge game on (F_i, V_j) w.r.t. "filtering set" E

- Adversary choses any total assignment α such that $\alpha(E) = 1$
- We can modify α on V_j to get α'
- We win if $\alpha'(F_i \wedge E) = 1$



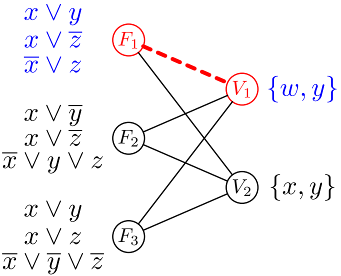
Edge game on (F_1, V_1) w.r.t. E

Take $\alpha_1 = \{x \mapsto 1, y \mapsto 0, z \mapsto 0\}$

The Resolution Edge Game

Resolution edge game on (F_i, V_j) w.r.t. "filtering set" E

- Adversary choses any total assignment α such that $\alpha(E) = 1$
- We can modify α on V_j to get α'
- We win if $\alpha'(F_i \wedge E) = 1$



$E = \{\overline{y} \vee z\}$

Edge game on (F_1, V_1) w.r.t. E

Take $\alpha_1 = \{x \mapsto 1, y \mapsto 0, z \mapsto 0\}$

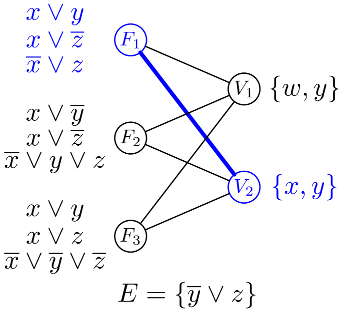
Can't win, since

- $\alpha_1(\overline{x} \vee z) = 0$
- can't flip x or z (not in V_1)

The Resolution Edge Game

Resolution edge game on (F_i, V_j) w.r.t. "filtering set" E

- Adversary choses any total assignment α such that $\alpha(E) = 1$
- We can modify α on V_j to get α'
- We win if $\alpha'(F_i \wedge E) = 1$

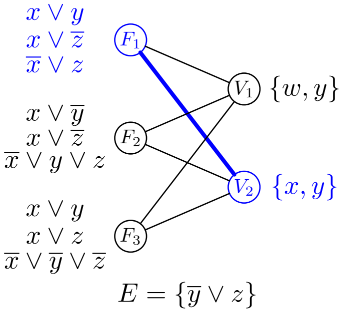


Edge game on (F_1, V_2) w.r.t. E

The Resolution Edge Game

Resolution edge game on (F_i, V_j) w.r.t. "filtering set" E

- Adversary choses any total assignment α such that $\alpha(E) = 1$
- We can modify α on V_j to get α'
- We win if $\alpha'(F_i \wedge E) = 1$



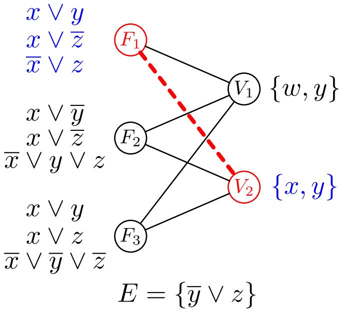
Edge game on (F_1, V_2) w.r.t. E

Take (partial) $\alpha_2 = \{y \mapsto 0, z \mapsto 0\}$

The Resolution Edge Game

Resolution edge game on (F_i, V_j) w.r.t. "filtering set" E

- Adversary choses any total assignment α such that $\alpha(E) = 1$
- We can modify α on V_j to get α'
- We win if $\alpha'(F_i \wedge E) = 1$



Edge game on (F_1, V_2) w.r.t. E

Take (partial) $\alpha_2 = \{y \mapsto 0, z \mapsto 0\}$

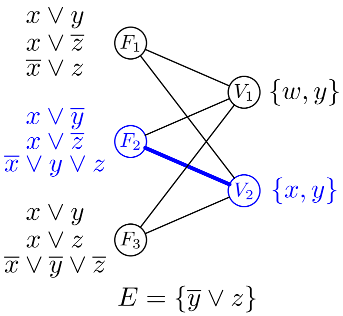
Again can't win, since

- can't flip z (not in V_2)
- flipping $y \in V_2$ falsifies E
- $F_1{\restriction}_{\alpha_2} = \{x, \overline{x}\}$

The Resolution Edge Game

Resolution edge game on (F_i, V_j) w.r.t. "filtering set" E

- Adversary choses any total assignment α such that $\alpha(E) = 1$
- We can modify α on V_j to get α'
- We win if $\alpha'(F_i \wedge E) = 1$

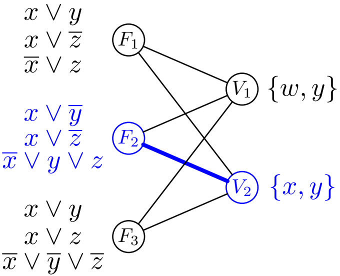


Edge game on (F_2, V_2) w.r.t. E

The Resolution Edge Game

Resolution edge game on (F_i, V_j) w.r.t. "filtering set" E

- Adversary choses any total assignment α such that $\alpha(E) = 1$
- We can modify α on V_j to get α'
- We win if $\alpha'(F_i \wedge E) = 1$



$E = \{\overline{y} \vee z\}$

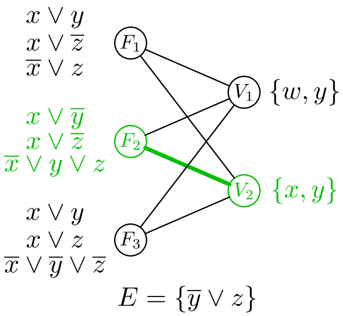
Edge game on (F_2, V_2) w.r.t. E

Now we can win!

The Resolution Edge Game

Resolution edge game on (F_i, V_j) w.r.t. "filtering set" E

- Adversary choses any total assignment α such that $\alpha(E) = 1$
- We can modify α on V_j to get α'
- We win if $\alpha'(F_i \wedge E) = 1$



Edge game on (F_2, V_2) w.r.t. E

Now we can win!

Given any α_3 s.t. $\alpha_3(E) = 1$:

- assign $x \mapsto \alpha_3(y \vee z)$
- E still OK — didn't touch y, z
- F_2 OK — encodes $x \leftrightarrow (y \vee z)$

Edge Game, Expansion, and Width Lower Bounds

Recall boundary $\partial(\mathcal{U}') = \{V \in \mathcal{N}(\mathcal{U}') \mid \mathcal{N}(V) \cap \mathcal{U}' = \{F\} \text{ unique}\}$

Edge Game, Expansion, and Width Lower Bounds

Recall boundary $\partial(\mathcal{U}') = \{V \in \mathcal{N}(\mathcal{U}') \mid \mathcal{N}(V) \cap \mathcal{U}' = \{F\} \text{ unique}\}$

Resolution expander

Say that an $(\mathcal{U}, \mathcal{V})_E$-graph is an (s, δ, E)-resolution expander if

- For all $\mathcal{U}' \subseteq \mathcal{U}$, $|\mathcal{U}'| \leq s$ it holds that $|\partial(\mathcal{U}')| \geq \delta|\mathcal{U}'|$
- For all edges (F_i, V_j) we can win the resolution edge game with respect to E

Edge Game, Expansion, and Width Lower Bounds

Recall boundary $\partial(\mathcal{U}') = \{V \in \mathcal{N}(\mathcal{U}') \mid \mathcal{N}(V) \cap \mathcal{U}' = \{F\} \text{ unique}\}$

Resolution expander

Say that an $(\mathcal{U}, \mathcal{V})_E$-graph is an (s, δ, E)-resolution expander if

- For all $\mathcal{U}' \subseteq \mathcal{U}$, $|\mathcal{U}'| \leq s$ it holds that $|\partial(\mathcal{U}')| \geq \delta|\mathcal{U}'|$
- For all edges (F_i, V_j) we can win the resolution edge game with respect to E

Theorem (essentially [BW01])

If the CNF formula $\mathcal{F}$ admits an (s, δ, E)-resolution expander with overlap ℓ, then

$$\textit{resolution proof width} > \frac{\delta s}{2\ell}$$

Progress Measure Approach (1/4)

Theorem (essentially [BW01])

If the CNF formula $\mathcal{F}$ admits an (s, δ, E)-resolution expander with overlap ℓ, then

$$\textit{resolution proof width} > \frac{\delta s}{2\ell}$$

Proof overview: Define "progress measure" $\mu : \{\text{clauses}\} \to \mathbb{N}$ such that

1. $\mu(\text{axiom clause}) = \mathcal{O}(1)$
2. $\mu(C \vee D) \leq \mu(C \vee x) + \mu(D \vee \overline{x})$
3. $\mu(\bot) > s$

Progress Measure Approach (1/4)

Theorem (essentially [BW01])

If the CNF formula $\mathcal{F}$ admits an (s, δ, E)-resolution expander with overlap ℓ, then

$$\textit{resolution proof width} > \frac{\delta s}{2\ell}$$

Proof overview: Define “progress measure” $\mu : \{\text{clauses}\} \to \mathbb{N}$ such that

1. $\mu(\text{axiom clause}) = \mathcal{O}(1)$
2. $\mu(C \vee D) \leq \mu(C \vee x) + \mu(D \vee \overline{x})$
3. $\mu(\bot) > s$

$\Rightarrow$ in any resolution proof $\exists\, C$ with $\mu(C) = \sigma$ for $s/2 < \sigma \leq s$

Progress Measure Approach (1/4)

Theorem (essentially [BW01])

If the CNF formula $\mathcal{F}$ admits an (s, δ, E)-resolution expander with overlap ℓ, then

$$\textit{resolution proof width} > \frac{\delta s}{2\ell}$$

Proof overview: Define "progress measure" $\mu : \{\text{clauses}\} \to \mathbb{N}$ such that

1. $\mu(\text{axiom clause}) = \mathcal{O}(1)$
2. $\mu(C \vee D) \leq \mu(C \vee x) + \mu(D \vee \overline{x})$
3. $\mu(\bot) > s$

$\Rightarrow$ in any resolution proof $\exists\, C$ with $\mu(C) = \sigma$ for $s/2 < \sigma \leq s$

$\Rightarrow$ such clause C has width $\geq \delta\sigma/\ell$ □

Progress Measure Approach (2/4)

Given (s, δ, E)-resolution expander $(\mathcal{U}, \mathcal{V})_E$ for $\mathcal{F}$, define

$$\mu(C) := \min\left\{ \left|\mathcal{U}'\right| ; \bigwedge_{F \in \mathcal{U}'} F \wedge E \vDash C \right\}$$

Progress Measure Approach (2/4)

Given (s, δ, E)-resolution expander $(\mathcal{U}, \mathcal{V})_E$ for $\mathcal{F}$, define

$$\mu(C) := \min\{|\mathcal{U}'| \,;\, \bigwedge_{F \in \mathcal{U}'} F \wedge E \vDash C\}$$

1. $\mu(A) = \mathcal{O}(1)$ for axioms $A \in \mathcal{F} = \bigcup_{i=1}^{m} F_i \cup E$
   - $A \in E$: $\mu(A) = 0$ since $E \vDash A$
   - $A \in F_i$: $\mu(A) = 1$ since $F_i \wedge E \vDash A$

Progress Measure Approach (2/4)

Given (s, δ, E)-resolution expander $(\mathcal{U}, \mathcal{V})_E$ for $\mathcal{F}$, define

$$\mu(C) := \min\left\{\left|\mathcal{U}'\right| ; \bigwedge_{F \in \mathcal{U}'} F \wedge E \vDash C\right\}$$

1. $\mu(A) = \mathcal{O}(1)$ for axioms $A \in \mathcal{F} = \bigcup_{i=1}^{m} F_i \cup E$
   - $A \in E$: $\mu(A) = 0$ since $E \vDash A$
   - $A \in F_i$: $\mu(A) = 1$ since $F_i \wedge E \vDash A$
2. $\mu(C \vee D) \leq \mu(C \vee x) + \mu(D \vee \overline{x})$
   - Fix minimal $\mathcal{U}_1$ s.t. $\bigwedge_{F \in \mathcal{U}_1} F \wedge E \vDash C \vee x$
   - Fix minimal $\mathcal{U}_2$ s.t. $\bigwedge_{F \in \mathcal{U}_2} F \wedge E \vDash D \vee \overline{x}$
   - Then it holds that

$$\bigwedge_{F \in \mathcal{U}_1 \cup \mathcal{U}_2} F \wedge E \vDash C \vee D ,$$

so $\mu(C \vee D) \leq \left|\mathcal{U}_1 \cup \mathcal{U}_2\right| \leq \left|\mathcal{U}_1\right| + \left|\mathcal{U}_2\right| = \mu(C \vee x) + \mu(D \vee \overline{x})$

Progress Measure Approach (3/4)

3. $\mu(\bot) > s$ for empty clause $\bot$

Progress Measure Approach (3/4)

3. $\mu(\bot) > s$ for empty clause $\bot$
   - Consider any $\mathcal{U}' \subseteq \mathcal{U}$, $\left|\mathcal{U}'\right| = s$, $\mathcal{U}' = \{F_1, \ldots, F_s\}$

Progress Measure Approach (3/4)

3. $\mu(\bot) > s$ for empty clause $\bot$
   - Consider any $\mathcal{U}' \subseteq \mathcal{U}$, $\left|\mathcal{U}'\right| = s$, $\mathcal{U}' = \{F_1, \ldots, F_s\}$
   - By expansion $\left|\partial(\mathcal{U}')\right| \geq \delta\left|\mathcal{U}'\right| > 0$

Progress Measure Approach (3/4)

3. $\mu(\perp) > s$ for empty clause $\perp$
   - Consider any $\mathcal{U}' \subseteq \mathcal{U}$, $\left|\mathcal{U}'\right| = s$, $\mathcal{U}' = \{F_1, \ldots, F_s\}$
   - By expansion $\left|\partial(\mathcal{U}')\right| \geq \delta\left|\mathcal{U}'\right| > 0$
   - By "peeling argument" $\exists$ matching $F_1 \leftrightarrow V_1, \ldots, F_s \leftrightarrow V_s$ such that $V_i \in \mathcal{N}(F_i) \setminus \mathcal{N}\left(\bigcup_{j=1}^{i-1} F_j\right)$ (i.e., V_i is not a neighbour of any F_j, $j < i$)

Progress Measure Approach (3/4)

3. $\mu(\bot) > s$ for empty clause $\bot$
   - Consider any $\mathcal{U}' \subseteq \mathcal{U}$, $|\mathcal{U}'| = s$, $\mathcal{U}' = \{F_1, \ldots, F_s\}$
   - By expansion $|\partial(\mathcal{U}')| \geq \delta|\mathcal{U}'| > 0$
   - By "peeling argument" $\exists$ matching $F_1 \leftrightarrow V_1, \ldots, F_s \leftrightarrow V_s$ such that $V_i \in \mathcal{N}(F_i) \setminus \mathcal{N}\left(\bigcup_{j=1}^{i-1} F_j\right)$ (i.e., V_i is not a neighbour of any F_j, $j < i$)
   - Given any α such that $\alpha(E) = 1$, for $i = 1, 2, \ldots, s$ we can modify α on V_i to satisfy F_i without falsifying $\bigwedge_{j<i} F_j \wedge E$ (since we can win the resolution edge game)

Progress Measure Approach (3/4)

3. $\mu(\bot) > s$ for empty clause $\bot$
   - Consider any $\mathcal{U}' \subseteq \mathcal{U}$, $|\mathcal{U}'| = s$, $\mathcal{U}' = \{F_1, \ldots, F_s\}$
   - By expansion $|\partial(\mathcal{U}')| \geq \delta|\mathcal{U}'| > 0$
   - By "peeling argument" $\exists$ matching $F_1 \leftrightarrow V_1, \ldots, F_s \leftrightarrow V_s$ such that $V_i \in \mathcal{N}(F_i) \setminus \mathcal{N}\left(\bigcup_{j=1}^{i-1} F_j\right)$ (i.e., V_i is not a neighbour of any F_j, $j < i$)
   - Given any α such that $\alpha(E) = 1$, for $i = 1, 2, \ldots, s$ we can modify α on V_i to satisfy F_i without falsifying $\bigwedge_{j<i} F_j \wedge E$ (since we can win the resolution edge game)
   - Yields α' such that $\alpha'\left(\bigwedge_{F_i \in \mathcal{U}'} F_i \wedge E\right) = 1$

Progress Measure Approach (3/4)

3. $\mu(\bot) > s$ for empty clause $\bot$
   - Consider any $\mathcal{U}' \subseteq \mathcal{U}$, $\lvert\mathcal{U}'\rvert = s$, $\mathcal{U}' = \{F_1, \ldots, F_s\}$
   - By expansion $\lvert\partial(\mathcal{U}')\rvert \geq \delta\lvert\mathcal{U}'\rvert > 0$
   - By "peeling argument" $\exists$ matching $F_1 \leftrightarrow V_1, \ldots, F_s \leftrightarrow V_s$ such that $V_i \in \mathcal{N}(F_i) \setminus \mathcal{N}\left(\bigcup_{j=1}^{i-1} F_j\right)$ (i.e., V_i is not a neighbour of any F_j, $j < i$)
   - Given any α such that $\alpha(E) = 1$, for $i = 1, 2, \ldots, s$ we can modify α on V_i to satisfy F_i without falsifying $\bigwedge_{j<i} F_j \wedge E$ (since we can win the resolution edge game)
   - Yields α' such that $\alpha'\left(\bigwedge_{F_i \in \mathcal{U}'} F_i \wedge E\right) = 1$
   - So $\bigwedge_{F_i \in \mathcal{U}'} F_i \wedge E \nvDash \bot$ for any $\lvert\mathcal{U}'\rvert \leq s$ and hence $\mu(\bot) > s$

Progress Measure Approach (4/4)

Given (s, δ, E)-resolution expander $(\mathcal{U}, \mathcal{V})_E$ with overlap ℓ

Progress Measure Approach (4/4)

Given (s, δ, E)-resolution expander $(\mathcal{U}, \mathcal{V})_E$ with overlap ℓ

Already showed: In any proof $\exists\, C$ with $\mu(C) = \sigma \in (s/2, s]$

Progress Measure Approach (4/4)

Given (s, δ, E)-resolution expander $(\mathcal{U}, \mathcal{V})_E$ with overlap ℓ

Already showed: In any proof $\exists\, C$ with $\mu(C) = \sigma \in (s/2, s]$

Want to show: $\mu(C) = \sigma \leq s$ implies C has width $\geq \delta\sigma/\ell$
Fix minimal $\mathcal{U}_C$ of size $\left|\mathcal{U}_C\right| = \sigma$ s.t. $\bigwedge_{F \in \mathcal{U}_C} F \wedge E \vDash C$

Progress Measure Approach (4/4)

Given (s, δ, E)-resolution expander $(\mathcal{U}, \mathcal{V})_E$ with overlap ℓ

Already showed: In any proof $\exists\, C$ with $\mu(C) = \sigma \in (s/2, s]$

Want to show: $\mu(C) = \sigma \leq s$ implies C has width $\geq \delta\sigma/\ell$
Fix minimal $\mathcal{U}_C$ of size $|\mathcal{U}_C| = \sigma$ s.t. $\bigwedge_{F \in \mathcal{U}_C} F \wedge E \vDash C$

Claim

If $V \in \partial(\mathcal{U}_C)$, then $V \cap \mathit{Vars}(C) \neq \emptyset$

Progress Measure Approach (4/4)

Given (s, δ, E)-resolution expander $(\mathcal{U}, \mathcal{V})_E$ with overlap ℓ

Already showed: In any proof $\exists\, C$ with $\mu(C) = \sigma \in (s/2, s]$

Want to show: $\mu(C) = \sigma \leq s$ implies C has width $\geq \delta\sigma/\ell$
Fix minimal $\mathcal{U}_C$ of size $\left|\mathcal{U}_C\right| = \sigma$ s.t. $\bigwedge_{F \in \mathcal{U}_C} F \wedge E \vDash C$

Claim

If $V \in \partial(\mathcal{U}_C)$, then $V \cap \mathit{Vars}(C) \neq \emptyset$

Since every variable occurs in $\leq \ell$ sets V, the clause C then must have width $\geq \left|\partial(\mathcal{U}_C)\right|/\ell \geq \delta\left|\mathcal{U}_C\right|/\ell = \delta\sigma/\ell$ □

Progress Measure Approach (4/4)

Given (s, δ, E)-resolution expander $(\mathcal{U}, \mathcal{V})_E$ with overlap ℓ

Already showed: In any proof $\exists\, C$ with $\mu(C) = \sigma \in (s/2, s]$

Want to show: $\mu(C) = \sigma \leq s$ implies C has width $\geq \delta\sigma/\ell$
Fix minimal $\mathcal{U}_C$ of size $|\mathcal{U}_C| = \sigma$ s.t. $\bigwedge_{F \in \mathcal{U}_C} F \wedge E \vDash C$

Claim

If $V \in \partial(\mathcal{U}_C)$, then $V \cap \mathit{Vars}(C) \neq \emptyset$

Since every variable occurs in $\leq \ell$ sets V, the clause C then must have width $\geq |\partial(\mathcal{U}_C)|/\ell \geq \delta|\mathcal{U}_C|/\ell = \delta\sigma/\ell$ □

Proof of claim: Another flipping argument using the resolution edge game:

- Fix $V \in \partial(\mathcal{U}_C)$ and unique neighbour $F_V \in \mathcal{U}_C$ of V
- By minimality, $\exists\, \alpha$ s.t. $\alpha\left(\bigwedge_{F \in \mathcal{U}_C \setminus \{F_V\}} F \wedge E\right) = 1$ but $\alpha(C) = 0$
- If $V \cap \mathit{Vars}(C) = \emptyset$, then flip α on V to satisfy $F_V \wedge E$ ↯

Applications: Tseitin and Onto-FPHP

Tseitin formulas

- F_i = clauses encoding parity constraint for ith vertex
- V_j = singleton set with jth edge (so overlap $\ell = 1$)
- $E = \emptyset$
- If underlying graph edge expander, then $(\mathcal{U}, \mathcal{V})_E$-graph is resolution expander with same parameters

Applications: Tseitin and Onto-FPHP

Tseitin formulas

- F_i = clauses encoding parity constraint for ith vertex
- V_j = singleton set with jth edge (so overlap $\ell = 1$)
- $E = \emptyset$
- If underlying graph edge expander, then $(\mathcal{U}, \mathcal{V})_E$-graph is resolution expander with same parameters

Onto functional PHP formulas

- F_i = singleton set with pigeon axiom for pigeon i
- V_j = all variables $p_{i,j}$ mentioning hole j (again overlap $\ell = 1$)
- E = all hole, functional, and onto axioms
- If onto FPHP restricted to bipartite graph, then $(\mathcal{U}, \mathcal{V})_E$-graph is resolution expander with same parameters

From Resolution to Polynomial Calculus

So far: Obtain resolution width lower bounds from expander graphs where we can win following game on all edges

Resolution edge game on (F, V) with respect to E

1. Adversary provides total assignment α such that $\alpha(E) = 1$
2. Choose $\alpha_V : V \rightarrow \{0, 1\}$ so that $\alpha[\alpha_V/V](F \wedge E) = 1$

From Resolution to Polynomial Calculus

So far: Obtain resolution width lower bounds from expander graphs where we can win following game on all edges

Resolution edge game on (F, V) with respect to E

1. Adversary provides total assignment α such that $\alpha(E) = 1$
2. Choose $\alpha_V : V \to \{0, 1\}$ so that $\alpha[\alpha_V/V](F \wedge E) = 1$

But Tseitin and onto FPHP both easy for polynomial calculus!

From Resolution to Polynomial Calculus

So far: Obtain resolution width lower bounds from expander graphs where we can win following game on all edges

Resolution edge game on (F, V) with respect to E

1. Adversary provides total assignment α such that $\alpha(E) = 1$
2. Choose $\alpha_V : V \to \{0, 1\}$ so that $\alpha[\alpha_V/V](F \wedge E) = 1$

But Tseitin and onto FPHP both easy for polynomial calculus!

Polynomial calculus degree lower bounds require harder game

Polynomial calculus edge game on (F, V) with respect to E

1. Commit to partial assignment $\alpha_V : V \to \{0, 1\}$
2. Adversary provides total assignment α such that $\alpha(E) = 1$
3. Substituting α_V for V should yield $\alpha[\alpha_V/V](F \wedge E) = 1$

The Polynomial Calculus Edge Game

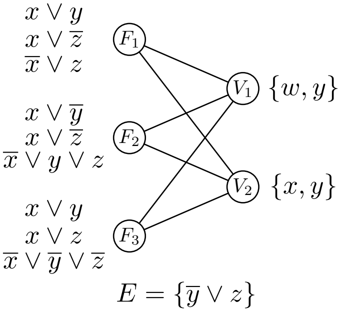
To win PC edge game on (F, V), need to find $\alpha_V : V \rightarrow \{0, 1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$

The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \to \{0, 1\}$ s.t.

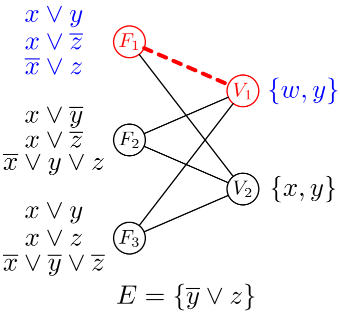
- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$



The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \to \{0,1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$



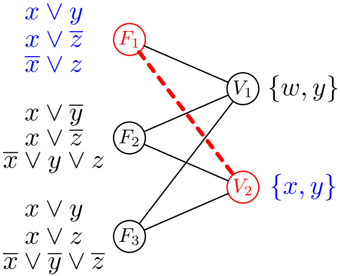
Recall that for resolution edge game we:

- Lose on (F_1, V_1)

The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \to \{0, 1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$



$E = \{\overline{y} \vee z\}$

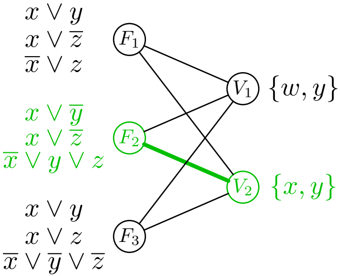
Recall that for resolution edge game we:

- Lose on (F_1, V_1)
- Lose on (F_1, V_2)

The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \to \{0, 1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$



$E = \{\overline{y} \vee z\}$

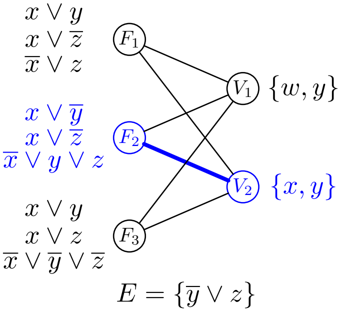
Recall that for resolution edge game we:

- Lose on (F_1, V_1)
- Lose on (F_1, V_2)
- Win on (F_2, V_2)

The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \to \{0,1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$

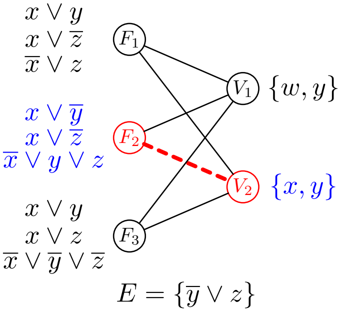


PC edge game on (F_2, V_2) w.r.t. E

The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \to \{0,1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$



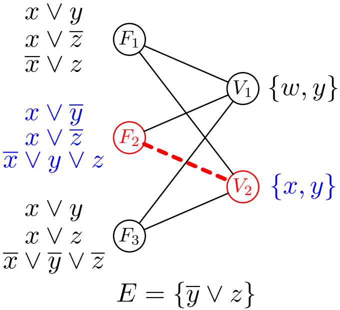
PC edge game on (F_2, V_2) w.r.t. E

Now we can't win

The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \rightarrow \{0,1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$



PC edge game on (F_2, V_2) w.r.t. E

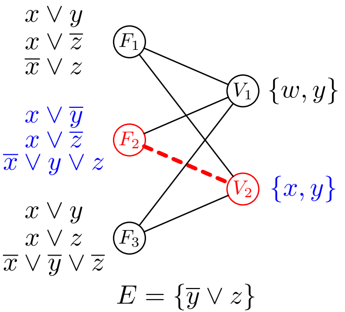
Now we can't win

- $E = \{\overline{y} \vee z\}$ needs $y \mapsto 0$

The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \to \{0,1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$



PC edge game on (F_2, V_2) w.r.t. E

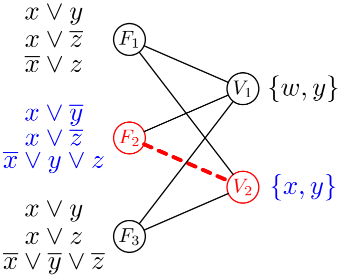
Now we can't win

- $E = \{\overline{y} \lor z\}$ needs $y \mapsto 0$
- But $F_2\restriction_{\{y \mapsto 0\}} = \{x \lor \overline{z},\ \overline{x} \lor z\}$

The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \rightarrow \{0,1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$



$E = \{\overline{y} \vee z\}$

PC edge game on (F_2, V_2) w.r.t. E

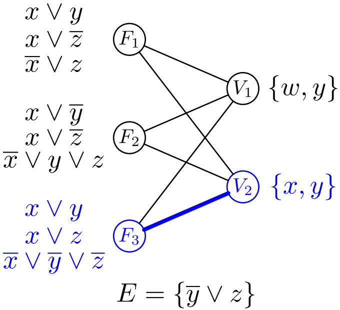
Now we can't win

- $E = \{\overline{y} \vee z\}$ needs $y \mapsto 0$
- But $F_2\restriction_{\{y \mapsto 0\}} = \{x \vee \overline{z},\ \overline{x} \vee z\}$
- Adversary sets $z \mapsto 1 - \alpha_V(x)$

The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \rightarrow \{0,1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$

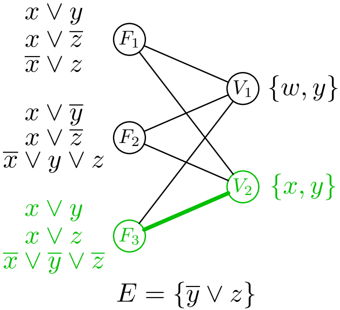


PC edge game on (F_3, V_2) w.r.t. E

The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \to \{0,1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$



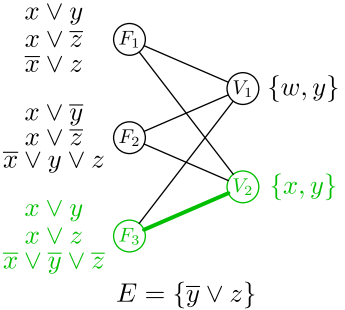
PC edge game on (F_3, V_2) w.r.t. E

On this edge we can win!

The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \rightarrow \{0,1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$



PC edge game on (F_3, V_2) w.r.t. E

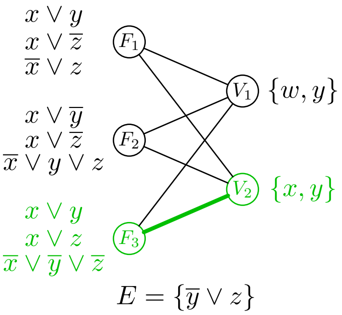
On this edge we can win!

- Choose $\alpha_V = \{x \mapsto 1, y \mapsto 0\}$

The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \to \{0,1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$



PC edge game on (F_3, V_2) w.r.t. E

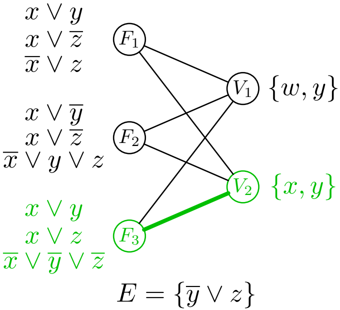
On this edge we can win!

- Choose $\alpha_V = \{x \mapsto 1, y \mapsto 0\}$
- $\alpha_V(F_3) = 1$

The Polynomial Calculus Edge Game

To win PC edge game on (F, V), need to find $\alpha_V : V \to \{0,1\}$ s.t.

- $\alpha_V(F) = 1$
- $\alpha_V(C) = 1$ for all clauses $C \in E$ with $V \cap Vars(C) \neq \emptyset$



PC edge game on (F_3, V_2) w.r.t. E

On this edge we can win!

- Choose $\alpha_V = \{x \mapsto 1, y \mapsto 0\}$
- $\alpha_V(F_3) = 1$
- $\alpha_V(E) = 1$

A Generalized Method for PC Degree Lower Bounds

Polynomial calculus expander

Say that an $(\mathcal{U}, \mathcal{V})_E$-graph is an (s, δ, E)-PC expander if

- For all $\mathcal{U}' \subseteq \mathcal{U}$, $|\mathcal{U}'| \leq s$ it holds that $|\partial(\mathcal{U}')| \geq \delta|\mathcal{U}'|$
- For all edges (F_i, V_j) we can win the PC edge game with respect to E

A Generalized Method for PC Degree Lower Bounds

Polynomial calculus expander

Say that an $(\mathcal{U}, \mathcal{V})_E$-graph is an (s, δ, E)-PC expander if

- For all $\mathcal{U}' \subseteq \mathcal{U}$, $|\mathcal{U}'| \leq s$ it holds that $|\partial(\mathcal{U}')| \geq \delta |\mathcal{U}'|$
- For all edges (F_i, V_j) we can win the PC edge game with respect to E

Theorem ([MN15] building on [AR03])

If $\mathcal{F}$ admits an (s, δ, E)-PC expander with overlap ℓ, then

$$\textit{PC proof degree} > \frac{\delta s}{2\ell}$$

A Generalized Method for PC Degree Lower Bounds

Polynomial calculus expander

Say that an $(\mathcal{U},\mathcal{V})_E$-graph is an (s,δ,E)-PC expander if

- For all $\mathcal{U}' \subseteq \mathcal{U}$, $|\mathcal{U}'| \leq s$ it holds that $|\partial(\mathcal{U}')| \geq \delta|\mathcal{U}'|$
- For all edges (F_i, V_j) we can win the PC edge game with respect to E

Theorem ([MN15] building on [AR03])

If $\mathcal{F}$ admits an (s,δ,E)-PC expander with overlap ℓ, then

$$\text{PC proof degree} > \frac{\delta s}{2\ell}$$

Also holds for sets of polynomials not obtained from CNFs
Proof by carefully adapting [AR03] (fairly involved — can't say much)

Consequences

Common framework for previous lower bounds

- Random k-CNF formulas [BI10, AR03]
- CNF formulas with expanding CVIGs [AR03]
- "Vanilla" PHP formulas [AR03]
- Ordering principle formulas [GL10]
- Subset cardinality formulas [MN14]

Consequences

Common framework for previous lower bounds

- Random k-CNF formulas [BI10, AR03]
- CNF formulas with expanding CVIGs [AR03]
- "Vanilla" PHP formulas [AR03]
- Ordering principle formulas [GL10]
- Subset cardinality formulas [MN14]

New lower bounds

- Functional pigeonhole principle [MN15]
- Graph colouring [LN17]

Hardness of Different Flavours of PHP

Variant	Resolution	Polynomial calculus
PHP		
FPHP		
Onto-PHP		
Onto-FPHP		

Hardness of Different Flavours of PHP

Variant	Resolution	Polynomial calculus
PHP	hard [Hak85]	
FPHP		
Onto-PHP		
Onto-FPHP		

Hardness of Different Flavours of PHP

Variant	Resolution	Polynomial calculus
PHP	hard [Hak85]	
FPHP	hard [Hak85]	
Onto-PHP	hard [Hak85]	
Onto-FPHP	hard [Hak85]	

Hardness of Different Flavours of PHP

Variant	Resolution	Polynomial calculus
PHP	hard [Hak85]	hard [AR03]
FPHP	hard [Hak85]	
Onto-PHP	hard [Hak85]	
Onto-FPHP	hard [Hak85]	

Hardness of Different Flavours of PHP

Variant	Resolution	Polynomial calculus
PHP	hard [Hak85]	hard [AR03]
FPHP	hard [Hak85]	
Onto-PHP	hard [Hak85]	
Onto-FPHP	hard [Hak85]	easy! [Rii93]

Hardness of Different Flavours of PHP

Variant	Resolution	Polynomial calculus
PHP	hard [Hak85]	hard [AR03]
FPHP	hard [Hak85]	?
Onto-PHP	hard [Hak85]	?
Onto-FPHP	hard [Hak85]	easy! [Rii93]

Hardness of Different Flavours of PHP

Variant	Resolution	Polynomial calculus
PHP	hard [Hak85]	hard [AR03]
FPHP	hard [Hak85]	?
Onto-PHP	hard [Hak85]	hard [AR03]
Onto-FPHP	hard [Hak85]	easy! [Rii93]

Joint work with Mladen Mikša [MN15]:

- Observe that [AR03] proves hardness of Onto-PHP

Hardness of Different Flavours of PHP

Variant	Resolution	Polynomial calculus
PHP	hard [Hak85]	hard [AR03]
FPHP	hard [Hak85]	hard [MN15]
Onto-PHP	hard [Hak85]	hard [AR03]
Onto-FPHP	hard [Hak85]	easy! [Rii93]

Joint work with Mladen Mikša [MN15]:

- Observe that [AR03] proves hardness of Onto-PHP
- Prove that functional PHP is hard for polynomial calculus (answering open question in [Raz02, Raz14])

Degree Lower Bound for Functional PHP

Theorem ([MN15])

If G is a (standard) bipartite (s, δ)-boundary expander with left degree $\leq d$, then $FPHP_G$ requires PC degree $> \delta s/(2d)$

Degree Lower Bound for Functional PHP

Theorem ([MN15])

If G is a (standard) bipartite (s, δ)-boundary expander with left degree $\leq d$, then $FPHP_G$ requires PC degree $> \delta s/(2d)$

Proof: Just need to build expanding $(\mathcal{U}, \mathcal{V})_E$-graph

Degree Lower Bound for Functional PHP

Theorem ([MN15])

If G is a (standard) bipartite (s, δ)-boundary expander with left degree $\leq d$, then $FPHP_G$ requires PC degree $> \delta s/(2d)$

Proof: Just need to build expanding $(\mathcal{U}, \mathcal{V})_E$-graph

- F_i = pigeon axiom for pigeon i

Degree Lower Bound for Functional PHP

Theorem ([MN15])

If G is a (standard) bipartite (s, δ)-boundary expander with left degree $\leq d$, then $FPHP_G$ requires PC degree $> \delta s/(2d)$

Proof: Just need to build expanding $(\mathcal{U}, \mathcal{V})_E$-graph

- F_i = pigeon axiom for pigeon i
- E = all hole and functional axioms

Degree Lower Bound for Functional PHP

Theorem ([MN15])

If G is a (standard) bipartite (s, δ)-boundary expander with left degree $\leq d$, then $FPHP_G$ requires PC degree $> \delta s/(2d)$

Proof: Just need to build expanding $(\mathcal{U}, \mathcal{V})_E$-graph

- F_i = pigeon axiom for pigeon i
- E = all hole and functional axioms
- $V_j = \left\{ p_{i',j'} \,\middle|\, i' \in \mathcal{N}(j) \text{ and } j' \in \mathcal{N}(i') \right\}$
  "All holes pigeons incident to hole j can go to"

Degree Lower Bound for Functional PHP

Theorem ([MN15])

If G is a (standard) bipartite (s, δ)-boundary expander with left degree $\leq d$, then $FPHP_G$ requires PC degree $> \delta s/(2d)$

Proof: Just need to build expanding $(\mathcal{U}, \mathcal{V})_E$-graph

- F_i = pigeon axiom for pigeon i
- E = all hole and functional axioms
- $V_j = \left\{ p_{i',j'} \,\middle|\, i' \in \mathcal{N}(j) \text{ and } j' \in \mathcal{N}(i') \right\}$
  "All holes pigeons incident to hole j can go to"
- Can prove (straightforward exercise):
  - Overlap ℓ satisfies $1 < \ell \leq d$
  - Can win PC edge game on all edges (F_i, V_j)
  - Original graph G and $(\mathcal{U}, \mathcal{V})_E$ are isomorphic

Degree Lower Bound for Functional PHP

Theorem ([MN15])

If G is a (standard) bipartite (s, δ)-boundary expander with left degree $\leq d$, then $FPHP_G$ requires PC degree $> \delta s/(2d)$

Proof: Just need to build expanding $(\mathcal{U}, \mathcal{V})_E$-graph

- F_i = pigeon axiom for pigeon i
- E = all hole and functional axioms
- $V_j = \left\{ p_{i',j'} \,\middle|\, i' \in \mathcal{N}(j) \text{ and } j' \in \mathcal{N}(i') \right\}$
  "All holes pigeons incident to hole j can go to"
- Can prove (straightforward exercise):
  - Overlap ℓ satisfies $1 < \ell \leq d$
  - Can win PC edge game on all edges (F_i, V_j)
  - Original graph G and $(\mathcal{U}, \mathcal{V})_E$ are isomorphic
- So get same expansion parameters, and theorem follows □

Graph Colouring

Graph k-colouring formulas
"$G = (V, E)$ is k-colourable"

Variables $x_{v,c}$ = "vertex v gets colour c"

$x_{v,1} \lor x_{v,2} \lor \cdots \lor x_{v,k}$	every vertex v gets a colour
$\overline{x}_{v,c} \lor \overline{x}_{v,c'}$	every vertex v is uniquely coloured
$\overline{x}_{u,c} \lor \overline{x}_{v,c}$	neighbours $(u, v) \in E$ get different colours

Graph Colouring

Graph k-colouring formulas
"$G = (V, E)$ is k-colourable"

Variables $x_{v,c}$ = "vertex v gets colour c"

$x_{v,1} \lor x_{v,2} \lor \cdots \lor x_{v,k}$	every vertex v gets a colour
$\overline{x}_{v,c} \lor \overline{x}_{v,c'}$	every vertex v is uniquely coloured
$\overline{x}_{u,c} \lor \overline{x}_{v,c}$	neighbours $(u, v) \in E$ get different colours

Average-case exponential lower bounds for resolution [BCMM05]

Graph Colouring

Graph k-colouring formulas
"$G = (V, E)$ is k-colourable"

Variables $x_{v,c}$ = "vertex v gets colour c"

$x_{v,1} \vee x_{v,2} \vee \cdots \vee x_{v,k}$	every vertex v gets a colour
$\overline{x}_{v,c} \vee \overline{x}_{v,c'}$	every vertex v is uniquely coloured
$\overline{x}_{u,c} \vee \overline{x}_{v,c}$	neighbours $(u, v) \in E$ get different colours

Average-case exponential lower bounds for resolution [BCMM05]

No lower bounds for polynomial calculus

Graph Colouring

Graph k-colouring formulas
"$G = (V, E)$ is k-colourable"

Variables $x_{v,c}$ = "vertex v gets colour c"

$x_{v,1} \vee x_{v,2} \vee \cdots \vee x_{v,k}$	every vertex v gets a colour
$\overline{x}_{v,c} \vee \overline{x}_{v,c'}$	every vertex v is uniquely coloured
$\overline{x}_{u,c} \vee \overline{x}_{v,c}$	neighbours $(u, v) \in E$ get different colours

Average-case exponential lower bounds for resolution [BCMM05]

No lower bounds for polynomial calculus

On the contrary, [DLMM08, DLMO09, DLMM11, DMP^{+}15] claim very efficient algorithms based on Nullstellensatz ("static PC") for slightly different encoding using primitive kth roots of unity

Polynomial Calculus Lower Bound for Colouring

Joint work with Massimo Lauria [LN17]:

Theorem ([LN17])

For any $k \geq 3$ $\exists$ constant-degree graphs which require linear PC degree, and hence exponential size, to be proven non-k-colourable

Polynomial Calculus Lower Bound for Colouring

Joint work with Massimo Lauria [LN17]:

Theorem ([LN17])

For any $k \geq 3$ $\exists$ constant-degree graphs which require linear PC degree, and hence exponential size, to be proven non-k-colourable

Proof idea:

- Reduce functional PHP instance to graph colouring instance
- Show that polynomial calculus "can compute this reduction"
- Hence these graph colouring instances must be hard

Polynomial Calculus Lower Bound for Colouring

Joint work with Massimo Lauria [LN17]:

Theorem ([LN17])

For any $k \geq 3$ $\exists$ constant-degree graphs which require linear PC degree, and hence exponential size, to be proven non-k-colourable

Proof idea:

- Reduce functional PHP instance to graph colouring instance
- Show that polynomial calculus "can compute this reduction"
- Hence these graph colouring instances must be hard

Lower bound applies also to kth-root-of-unity encoding

Answers open question raised in [DLMO09, LLO16]

Sketch of Reduction

- Given FPHP instance for bipartite graph of left degree k

Sketch of Reduction

- Given FPHP instance for bipartite graph of left degree k
- Order available holes $\mathcal{N}(i) = \{j_{i,1}, \ldots, j_{i,c}\}$ for every pigeon i

Sketch of Reduction

- Given FPHP instance for bipartite graph of left degree k
- Order available holes $\mathcal{N}(i) = \{j_{i,1}, \ldots, j_{i,c}\}$ for every pigeon i
- Vertex i coloured with colour c $\Leftrightarrow$ pigeon i flies to hole $j_{i,c}$

Sketch of Reduction

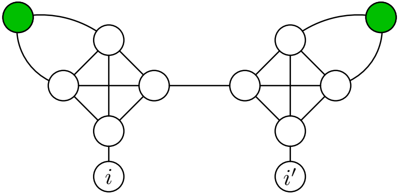
- Given FPHP instance for bipartite graph of left degree k
- Order available holes $\mathcal{N}(i) = \{j_{i,1}, \ldots, j_{i,c}\}$ for every pigeon i
- Vertex i coloured with colour c $\Leftrightarrow$ pigeon i flies to hole $j_{i,c}$
- $j_{i,c} = j_{i',c'}$ $\Rightarrow$ can't colour i by c and i' by c' simultaneously

Sketch of Reduction

- Given FPHP instance for bipartite graph of left degree k
- Order available holes $\mathcal{N}(i) = \{j_{i,1}, \ldots, j_{i,c}\}$ for every pigeon i
- Vertex i coloured with colour c $\Leftrightarrow$ pigeon i flies to hole $j_{i,c}$
- $j_{i,c} = j_{i',c'}$ $\Rightarrow$ can't colour i by c and i' by c' simultaneously
- *Almost* colouring, except forbidding specific colour pair (c, c') instead of arbitrary but same colour — fix with gadgets!

Sketch of Reduction

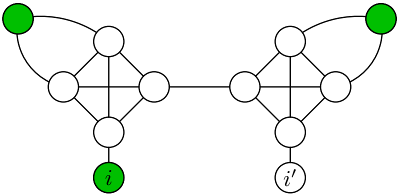
- Given FPHP instance for bipartite graph of left degree k
- Order available holes $\mathcal{N}(i) = \{j_{i,1}, \ldots, j_{i,c}\}$ for every pigeon i
- Vertex i coloured with colour c $\Leftrightarrow$ pigeon i flies to hole $j_{i,c}$
- $j_{i,c} = j_{i',c'}$ $\Rightarrow$ can't colour i by c and i' by c' simultaneously
- *Almost* colouring, except forbidding specific colour pair (c, c') instead of arbitrary but same colour — fix with gadgets!



not i and i' both green

Sketch of Reduction

- Given FPHP instance for bipartite graph of left degree k
- Order available holes $\mathcal{N}(i) = \{j_{i,1}, \ldots, j_{i,c}\}$ for every pigeon i
- Vertex i coloured with colour c $\Leftrightarrow$ pigeon i flies to hole $j_{i,c}$
- $j_{i,c} = j_{i',c'}$ $\Rightarrow$ can't colour i by c and i' by c' simultaneously
- *Almost* colouring, except forbidding specific colour pair (c, c') instead of arbitrary but same colour — fix with gadgets!

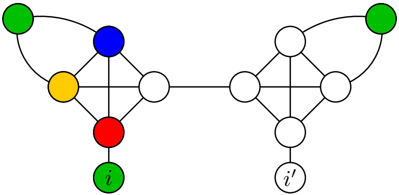


not i and i' both green

Colouring i green...

Sketch of Reduction

- Given FPHP instance for bipartite graph of left degree k
- Order available holes $\mathcal{N}(i) = \{j_{i,1}, \ldots, j_{i,c}\}$ for every pigeon i
- Vertex i coloured with colour c $\Leftrightarrow$ pigeon i flies to hole $j_{i,c}$
- $j_{i,c} = j_{i',c'}$ $\Rightarrow$ can't colour i by c and i' by c' simultaneously
- *Almost* colouring, except forbidding specific colour pair (c, c') instead of arbitrary but same colour — fix with gadgets!

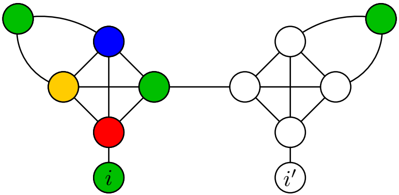


not i and i' both green

Colouring i green forces left 4-clique use all other colours

Sketch of Reduction

- Given FPHP instance for bipartite graph of left degree k
- Order available holes $\mathcal{N}(i) = \{j_{i,1}, \ldots, j_{i,c}\}$ for every pigeon i
- Vertex i coloured with colour c $\Leftrightarrow$ pigeon i flies to hole $j_{i,c}$
- $j_{i,c} = j_{i',c'}$ $\Rightarrow$ can't colour i by c and i' by c' simultaneously
- *Almost* colouring, except forbidding specific colour pair (c, c') instead of arbitrary but same colour — fix with gadgets!

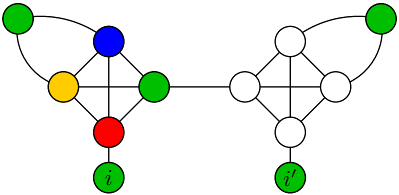


not i and i' both green

Colouring i green forces left 4-clique use all other colours making rightmost node green

Sketch of Reduction

- Given FPHP instance for bipartite graph of left degree k
- Order available holes $\mathcal{N}(i) = \{j_{i,1}, \ldots, j_{i,c}\}$ for every pigeon i
- Vertex i coloured with colour c $\Leftrightarrow$ pigeon i flies to hole $j_{i,c}$
- $j_{i,c} = j_{i',c'}$ $\Rightarrow$ can't colour i by c and i' by c' simultaneously
- *Almost* colouring, except forbidding specific colour pair (c, c') instead of arbitrary but same colour — fix with gadgets!



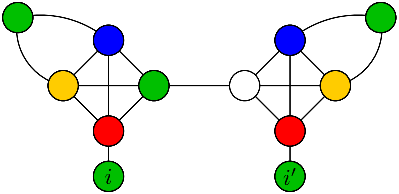
not i and i' both green

Colouring i green forces left 4-clique use all other colours making rightmost node green

Symmetric argument in right subgadget. . .

Sketch of Reduction

- Given FPHP instance for bipartite graph of left degree k
- Order available holes $\mathcal{N}(i) = \{j_{i,1}, \ldots, j_{i,c}\}$ for every pigeon i
- Vertex i coloured with colour c $\Leftrightarrow$ pigeon i flies to hole $j_{i,c}$
- $j_{i,c} = j_{i',c'}$ $\Rightarrow$ can't colour i by c and i' by c' simultaneously
- *Almost* colouring, except forbidding specific colour pair (c, c') instead of arbitrary but same colour — fix with gadgets!



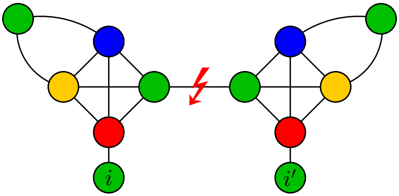
not i and i' both green

Colouring i green forces left 4-clique use all other colours making rightmost node green

Symmetric argument in right subgadget. . .

Sketch of Reduction

- Given FPHP instance for bipartite graph of left degree k
- Order available holes $\mathcal{N}(i) = \{j_{i,1}, \ldots, j_{i,c}\}$ for every pigeon i
- Vertex i coloured with colour c $\Leftrightarrow$ pigeon i flies to hole $j_{i,c}$
- $j_{i,c} = j_{i',c'}$ $\Rightarrow$ can't colour i by c and i' by c' simultaneously
- *Almost* colouring, except forbidding specific colour pair (c, c') instead of arbitrary but same colour — fix with gadgets!



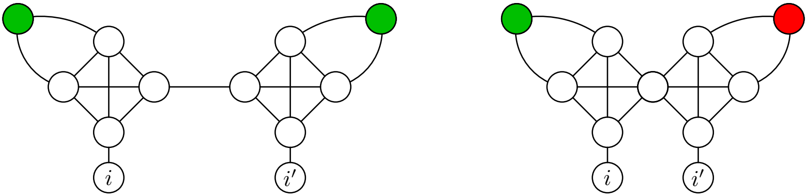
not i and i' both green

Colouring i green forces left 4-clique use all other colours making rightmost node green

Symmetric argument in right subgadget $\Rightarrow$ **contradiction**

Sketch of Reduction

- Given FPHP instance for bipartite graph of left degree k
- Order available holes $\mathcal{N}(i) = \{j_{i,1}, \ldots, j_{i,c}\}$ for every pigeon i
- Vertex i coloured with colour $c \Leftrightarrow$ pigeon i flies to hole $j_{i,c}$
- $j_{i,c} = j_{i',c'} \Rightarrow$ can't colour i by c and i' by c' simultaneously
- *Almost* colouring, except forbidding specific colour pair (c, c') instead of arbitrary but same colour — fix with gadgets!



not i and i' both green

not i green and i' red

Open Problems

- Prove polynomial calculus lower bounds for other formulas

Open Problems

- Prove polynomial calculus lower bounds for other formulas
  - independent set formulas
  - graph colouring formulas (average-case)

Open Problems

- Prove polynomial calculus lower bounds for other formulas
  - independent set formulas
  - graph colouring formulas (average-case)
- Prove size lower bounds via technique that doesn't use degree

Open Problems

- Prove polynomial calculus lower bounds for other formulas
  - independent set formulas
  - graph colouring formulas (average-case)
- Prove size lower bounds via technique that doesn't use degree
  - k-clique formulas
  - weak pigeonhole principle formulas ($\geq n^2$ pigeons)

Open Problems

- Prove polynomial calculus lower bounds for other formulas
  - independent set formulas
  - graph colouring formulas (average-case)
- Prove size lower bounds via technique that doesn't use degree
  - k-clique formulas
  - weak pigeonhole principle formulas ($\geq n^2$ pigeons)
- Find truly general framework capturing all degree bounds

Open Problems

- Prove polynomial calculus lower bounds for other formulas
  - independent set formulas
  - graph colouring formulas (average-case)
- Prove size lower bounds via technique that doesn't use degree
  - k-clique formulas
  - weak pigeonhole principle formulas ($\geq n^2$ pigeons)
- Find truly general framework capturing all degree bounds
  - We generalize only part of [AR03]
  - Cannot handle characteristic-dependent bounds à la [BGIP01]

Open Problems

- Prove polynomial calculus lower bounds for other formulas
  - independent set formulas
  - graph colouring formulas (average-case)
- Prove size lower bounds via technique that doesn't use degree
  - k-clique formulas
  - weak pigeonhole principle formulas ($\geq n^2$ pigeons)
- Find truly general framework capturing all degree bounds
  - We generalize only part of [AR03]
  - Cannot handle characteristic-dependent bounds à la [BGIP01]
- Go beyond polynomial calculus (e.g. to Positivstellensatz, a.k.a. Lasserre/sums-of-squares)

Take-away Message

Generalized method for width and degree lower bounds

- Unified framework for most previous lower bounds
- Highlights similarities and differences between resolution and polynomial calculus
- Exponential polynomial calculus size lower bound for
  - functional PHP
  - graph colouring

Future directions

- Extend techniques further to other tricky formulas
- Develop non-degree-based size lower bound techniques

Take-away Message

Generalized method for width and degree lower bounds

- Unified framework for most previous lower bounds
- Highlights similarities and differences between resolution and polynomial calculus
- Exponential polynomial calculus size lower bound for
  - functional PHP
  - graph colouring

Future directions

- Extend techniques further to other tricky formulas
- Develop non-degree-based size lower bound techniques

Thank you for your attention!

References I

[ABRW02] Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Space complexity in propositional calculus. *SIAM Journal on Computing*, 31(4):1184–1211, 2002. Preliminary version in *STOC '00*.

[ALN16] Albert Atserias, Massimo Lauria, and Jakob Nordström. Narrow proofs may be maximally long. *ACM Transactions on Computational Logic*, 17(3):19:1–19:30, May 2016. Preliminary version in *CCC '14*.

[AR03] Michael Alekhnovich and Alexander A. Razborov. Lower bounds for polynomial calculus: Non-binomial case. *Proceedings of the Steklov Institute of Mathematics*, 242:18–35, 2003. Available at `http://people.cs.uchicago.edu/~razborov/files/misha.pdf`. Preliminary version in *FOCS '01*.

[BCMM05] Paul Beame, Joseph C. Culberson, David G. Mitchell, and Cristopher Moore. The resolution complexity of random graph k-colorability. *Discrete Applied Mathematics*, 153(1-3):25–47, December 2005.

[BG01] María Luisa Bonet and Nicola Galesi. Optimality of size-width tradeoffs for resolution. *Computational Complexity*, 10(4):261–276, December 2001. Preliminary version in *FOCS '99*.

References II

[BGIP01] Samuel R. Buss, Dima Grigoriev, Russell Impagliazzo, and Toniann Pitassi. Linear gaps between degrees for the polynomial calculus modulo distinct primes. *Journal of Computer and System Sciences*, 62(2):267–289, March 2001. Preliminary version in *CCC '99*.

[BI10] Eli Ben-Sasson and Russell Impagliazzo. Random CNF's are hard for the polynomial calculus. *Computational Complexity*, 19(4):501–519, 2010. Preliminary version in *FOCS '99*.

[BIS07] Paul Beame, Russell Impagliazzo, and Ashish Sabharwal. The resolution complexity of independent sets and vertex covers in random graphs. *Computational Complexity*, 16(3):245–297, October 2007. Preliminary version in *CCC '01*.

[BKPS02] Paul Beame, Richard Karp, Toniann Pitassi, and Michael Saks. The efficiency of resolution and Davis-Putnam procedures. *SIAM Journal on Computing*, 31(4):1048–1075, 2002. Preliminary versions of these results appeared in *FOCS '96* and *STOC '98*.

References III

[BW01] Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow—resolution made simple. *Journal of the ACM*, 48(2):149–169, March 2001. Preliminary version in *STOC '99*.

[CEI96] Matthew Clegg, Jeffery Edmonds, and Russell Impagliazzo. Using the Groebner basis algorithm to find proofs of unsatisfiability. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC '96)*, pages 174–183, May 1996.

[Coo71] Stephen A. Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing (STOC '71)*, pages 151–158, 1971.

[CR79] Stephen A. Cook and Robert Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, March 1979.

[CS88] Vašek Chvátal and Endre Szemerédi. Many hard examples for resolution. *Journal of the ACM*, 35(4):759–768, October 1988.

References IV

[DLMM08] Jesús A. De Loera, Jon Lee, Peter N. Malkin, and Susan Margulies. Hilbert's Nullstellensatz and an algorithm for proving combinatorial infeasibility. In *Proceedings of the 21st International Symposium on Symbolic and Algebraic Computation (ISSAC '08)*, pages 197–206, July 2008.

[DLMM11] Jesús A. De Loera, Jon Lee, Peter N. Malkin, and Susan Margulies. Computing infeasibility certificates for combinatorial problems through Hilbert's Nullstellensatz. *Journal of Symbolic Computation*, 46(11):1260–1283, November 2011.

[DLMO09] Jesús A. De Loera, Jon Lee, Susan Margulies, and Shmuel Onn. Expressing combinatorial problems by systems of polynomial equations and Hilbert's Nullstellensatz. *Combinatorics, Probability and Computing*, 18(04):551–582, July 2009.

References V

[DMP+15] Jesús A. De Loera, Susan Margulies, Michael Pernpeintner, Eric Riedl, David Rolnick, Gwen Spencer, Despina Stasi, and Jon Swenson. Graph-coloring ideals: Nullstellensatz certificates, Gröbner bases for chordal graphs, and hardness of Gröbner bases. In *Proceedings of the 40th International Symposium on Symbolic and Algebraic Computation (ISSAC '15)*, pages 133–140, July 2015.

[GL10] Nicola Galesi and Massimo Lauria. Optimality of size-degree trade-offs for polynomial calculus. *ACM Transactions on Computational Logic*, 12(1):4:1–4:22, November 2010.

[Hak85] Armin Haken. The intractability of resolution. *Theoretical Computer Science*, 39(2-3):297–308, August 1985.

[IPS99] Russell Impagliazzo, Pavel Pudlák, and Jiří Sgall. Lower bounds for the polynomial calculus and the Gröbner basis algorithm. *Computational Complexity*, 8(2):127–144, 1999.

[Lev73] Leonid A. Levin. Universal sequential search problems. *Problemy peredachi informatsii*, 9(3):115–116, 1973. In Russian. Available at `http://mi.mathnet.ru/ppi914`.

References VI

[LLO16] Bo Li, Benjamin Lowenstein, and Mohamed Omar. Low degree Nullstellensatz certificates for 3-colorability. *The Electronic Journal of Combinatorics*, 23(1), January 2016.

[LN17] Massimo Lauria and Jakob Nordström. Graph colouring is hard for algorithms based on Hilbert's Nullstellensatz and Gröbner bases. In *Proceedings of the 32nd Annual Computational Complexity Conference (CCC '17)*, volume 79 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:20, July 2017.

[MN14] Mladen Mikša and Jakob Nordström. Long proofs of (seemingly) simple formulas. In *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT '14)*, volume 8561 of *Lecture Notes in Computer Science*, pages 121–137. Springer, July 2014.

[MN15] Mladen Mikša and Jakob Nordström. A generalized method for proving polynomial calculus degree lower bounds. In *Proceedings of the 30th Annual Computational Complexity Conference (CCC '15)*, volume 33 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 467–487, June 2015.

References VII

[Raz02] Alexander A. Razborov. Proof complexity of pigeonhole principles. In *5th International Conference on Developments in Language Theory, (DLT '01), Revised Papers*, volume 2295 of *Lecture Notes in Computer Science*, pages 100–116. Springer, July 2002.

[Raz14] Alexander A. Razborov. Possible research directions. List of open problems (in proof complexity and other areas) available at `http://people.cs.uchicago.edu/~razborov/teaching/`, 2014.

[Rii93] Søren Riis. *Independence in Bounded Arithmetic*. PhD thesis, University of Oxford, 1993.

[Spe10] Ivor Spence. sgen1: A generator of small but difficult satisfiability benchmarks. *Journal of Experimental Algorithmics*, 15:1.2:1–1.2:15, March 2010.

[Stå96] Gunnar Stålmarck. Short resolution proofs for a sequence of tricky formulas. *Acta Informatica*, 33(3):277–280, May 1996.

[Urq87] Alasdair Urquhart. Hard examples for resolution. *Journal of the ACM*, 34(1):209–219, January 1987.

References VIII

[VS10] Allen Van Gelder and Ivor Spence. Zero-one designs produce small hard SAT instances. In *Proceedings of the 13th International Conference on Theory and Applications of Satisfiability Testing (SAT '10)*, volume 6175 of *Lecture Notes in Computer Science*, pages 388–397. Springer, July 2010.