Introduction to Proof Complexity

Olaf Beyersdorff

Institute of Computer Science
Friedrich Schiller University Jena

ICTS 2026

Outline

Resolution

A General Framework for Proof Systems

Cutting Planes

Tree-like Resolution and Prover-Delayer Games

A Lower Bound for the Pigeonhole Principle

Separation of Tree-like and General Resolution

General Resolution and the Size-Width Relation

Application to Tseitin Formulas

Application to the Pigeonhole Principle

Resolution

Normal forms – definition

- Literals are negated or unnegated variables.
- A clause is a finite set of literals.
- A formula φ is in conjunctive normal form (CNF) if

$$\varphi = \bigwedge_{i=1}^{n} \bigvee_{j=1}^{m_i} L_{ij}$$

  with literals L_{ij}.

Example

$\varphi = (x \vee \neg y) \wedge (\neg x \vee y \vee \neg z) \wedge z$ is in CNF,
with clauses $x \vee \neg y$, $\neg x \vee y \vee \neg z$, z.

Conjunctive normal form – definition

- Alternatively, we view φ as a set $\{C_1, \ldots, C_n\}$ of clauses with
$$C_i = \{L_{ij} \mid j = 1, \ldots, m_i\} \ .$$

Example

$\varphi = (x \vee \neg y) \wedge (\neg x \vee y \vee \neg z) \wedge z$ as a set of clauses:

$$\varphi \ = \ \big\{ \{x, \neg y\},\ \{\neg x, y, \neg z\},\ \{z\} \big\}$$

Satisfiability of clauses – definition

- A clause C is satisfied by an assignment α if α satisfies at least one literal of C.
  Notation: $\alpha \models C$
- A set of clauses $\Gamma = \{C_1, \ldots, C_n\}$ is satisfied by α if $\alpha \models C_i$ for $i = 1, \ldots, n$.
  Notation: $\alpha \models \Gamma$
- The empty clause is denoted by $\Box$ and is unsatisfiable.

Resolution

- Introduced by Blake 1937 as well as Davis and Putnam 1960 and Robinson 1965
- Resolution proofs operate with clauses.
- Resolution proofs are refutation proofs.

Definition

Let C and D be clauses with $p \in C$ and $\neg p \in D$.
The resolution rule applied to C and D yields the clause $(C \setminus \{p\}) \cup (D \setminus \{\neg p\})$.

Notation: $$\frac{C \qquad\qquad D}{(C \setminus \{p\}) \cup (D \setminus \{\neg p\})}$$

Resolution derivations

Definition

Let Γ be a set of clauses. A resolution derivation of a clause C from Γ is a sequence

$$C_1, \ldots, C_k = C$$

of clauses such that for all $i = 1, \ldots, k$

1. $C_i \in \Gamma$ or
2. there exist $1 \leq j_1 \leq j_2 < i$ with

$$\frac{C_{j_1} \quad C_{j_2}}{C_i} \quad .$$

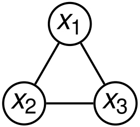
Resolution refutations

Definition

- A resolution refutation of Γ is a resolution derivation of $\Box$ from Γ.
- Notation: $\Gamma \vdash_{Res} C$ resp. $\Gamma \vdash_{Res} \Box$.

Example of a resolution refutation

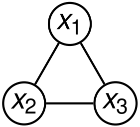
- Can the triangle be coloured with two colours?
- $x_i = 1$: node i is red; $x_i = 0$: node i is blue
- Edge $\{i, j\}$: different colours, i.e. clauses $\{x_i, x_j\}$ (not both blue) and $\{\neg x_i, \neg x_j\}$ (not both red)



$$\Gamma = \{ \{x_1, x_2\}, \{\neg x_1, \neg x_2\}, \{x_2, x_3\}, \{\neg x_2, \neg x_3\}, \{x_1, x_3\}, \{\neg x_1, \neg x_3\} \}$$

Example of a resolution refutation

- Can the triangle be coloured with two colours?
- $x_i = 1$: node i is red; $x_i = 0$: node i is blue
- Edge $\{i, j\}$: different colours, i.e. clauses $\{x_i, x_j\}$ (not both blue) and $\{\neg x_i, \neg x_j\}$ (not both red)



$$\Gamma = \{\, \{x_1, x_2\}, \{\neg x_1, \neg x_2\}, \{x_2, x_3\}, \{\neg x_2, \neg x_3\}, \{x_1, x_3\}, \{\neg x_1, \neg x_3\} \,\}$$

A tree-like resolution refutation of Γ:

$$\dfrac{\dfrac{\dfrac{\{x_1, x_2\} \quad \{\neg x_2, \neg x_3\}}{\{x_1, \neg x_3\}} \quad \{x_1, x_3\}}{\{x_1\}} \qquad \dfrac{\dfrac{\{\neg x_1, \neg x_2\} \quad \{x_2, x_3\}}{\{\neg x_1, x_3\}} \quad \{\neg x_1, \neg x_3\}}{\{\neg x_1\}}}{\square}$$

Completeness of Resolution

- Resolution is a refutation calculus, i.e. we show $\varphi \in \mathit{UNSAT}$ via $\varphi \vdash_{Res} \Box$.
- The resolution calculus is not complete.
- E.g. $p \models p \vee q$, but $p \nvdash_{Res} p \vee q$.
- Resolution is, however, refutationally complete.

Completeness of Resolution

- Resolution is a refutation calculus, i.e. we show $\varphi \in \mathit{UNSAT}$ via $\varphi \vdash_{Res} \Box$.
- The resolution calculus is not complete.
- E.g. $p \models p \vee q$, but $p \nvdash_{Res} p \vee q$.
- Resolution is, however, refutationally complete.

Theorem (Refutational completeness for Resolution)

Let Γ be a finite set of clauses. Then Γ is unsatisfiable if and only if there is a resolution refutation of Γ.
Symbolically: $\Gamma \models \Box \iff \Gamma \vdash_{Res} \Box$

General framework

for proof systems

Proof Systems

Definition (Cook, Reckhow 1979)

A proof system for a language L is a polynomial-time computable function f with $rng(f) = L$.
If $f(w) = x$, then w is called an f-proof of $x \in L$.

- correctness: $rng(f) \subseteq L$
- completeness: $L \subseteq rng(f)$
- efficiency: proofs should be easy to check,
  i.e. f should be easy to compute.

- Most research in proof complexity has studied propositional proof systems where $L = \text{TAUT}$.

Reductions between Proof Systems

Definition (Cook, Reckhow 1979, Krajíček, Pudlák 1989)

Let f and g be proof systems for L.

- f simulates g, if for any g-proof w there is an f-proof w' of length $|w'| = |w|^{O(1)}$ s.t. $f(w') = g(w)$.
- If w' is computable from w in polynomial time, then f p-simulates g.
- f and g are (p-)equivalent if they (p-)simulate each other.

Reductions between Proof Systems

Definition (Cook, Reckhow 1979, Krajíček, Pudlák 1989)

Let f and g be proof systems for L.

- f simulates g, if for any g-proof w there is an f-proof w' of length $|w'| = |w|^{O(1)}$ s.t. $f(w') = g(w)$.
- If w' is computable from w in polynomial time, then f p-simulates g.
- f and g are (p-)equivalent if they (p-)simulate each other.

Definition (Krajíček, Pudlák 1989)

A proof system f for L is (p)-optimal if f (p-)simulates every proof system for L.

Examples

Truth tables as a proof system

$$TT(\alpha, \varphi) = \begin{cases} \varphi & \text{if } \alpha \text{ is a truth table for } \varphi \text{ with all entries 1} \\ p \vee \neg p & \text{otherwise.} \end{cases}$$

Proof systems we have seen so far

- Resolution and its subsystems

Aims

- conceptually different proof systems
- compare efficiency of proof systems
- proof systems for other logics

Polynomially Bounded Proof Systems

Polynomial bounds on proofs

A proof system f for L is polynomially bounded if there exists a polynomial p such that every $x \in L$ has an f-proof of size $\leq p(|x|)$.

Polynomially Bounded Proof Systems

Polynomial bounds on proofs

A proof system f for L is polynomially bounded if there exists a polynomial p such that every $x \in L$ has an f-proof of size $\leq p(|x|)$.

Examples

- The standard proof system for SAT is polynomially bounded:

$$sat(\alpha, \varphi) = \begin{cases} \varphi & \text{if } \alpha \text{ is a satisfying assignment for } \varphi \\ p & \text{otherwise.} \end{cases}$$

- The truth-table system is not a polynomially bounded proof system for TAUT.
- Resolution is not polynomially bounded.

The Cook-Reckhow Theorem

Question

Is there a polynomially bounded proof system for TAUT?

The Cook-Reckhow Theorem

Question

Is there a polynomially bounded proof system for TAUT?

Theorem (Cook, Reckhow 1979)

A language L has a polynomially bounded proof system if and only if $L \in \mathrm{NP}$.

The Cook-Reckhow Theorem

Theorem (Cook, Reckhow 1979)

A language L has a polynomially bounded proof system if and only if $L \in \mathrm{NP}$.

Proof. $\Rightarrow$

Let P be a polynomially bounded proof system with bounding polynomial p. Consider the following algorithm:

```
Input:  a string x
guess π ∈ Σ^{≤p(|x|)}
IF P(π) = x THEN accept ELSE reject
```

The Cook-Reckhow Theorem

Theorem (Cook, Reckhow 1979)

A language L has a polynomially bounded proof system if and only if $L \in \mathrm{NP}$.

Proof. $\Leftarrow$

Let $L \in \mathrm{NP}$ and let M be a nondeterministic polynomial time Turing machine M that accepts L. Let the polynomial p bound the running time of M. Then

$$P(\pi) = \begin{cases} x & \text{if } \pi \text{ codes an accepting computation of } M(x) \\ x_0 & \text{otherwise} \end{cases}$$

with fixed $x_0 \in L$ is a proof system for L which is polynomially bounded by p. $\square$

The Cook-Reckhow Theorem

Question

Is there a polynomially bounded proof system for TAUT?

Theorem (Cook, Reckhow 1979)

A language L has a polynomially bounded proof system if and only if $L \in \mathrm{NP}$.

The Cook-Reckhow Theorem

Question

Is there a polynomially bounded proof system for TAUT?

Theorem (Cook, Reckhow 1979)

A language L has a polynomially bounded proof system if and only if $L \in \mathrm{NP}$.

For propositional proof systems

TAUT has a polynomially bounded proof system if and only if $\mathrm{NP} = \mathrm{coNP}$.

Cook's Programme

Separate NP from coNP (and hence P and NP) by showing super-polynomial lower bounds to the size of proofs in all propositional proof systems.

Cook's Programme

Separate NP from coNP (and hence P and NP) by showing super-polynomial lower bounds to the size of proofs in all propositional proof systems.

Showing lower bounds for a system P means

finding an infinite family θ_n of propositional tautologies s.t.

- $|\theta_n| = n^{O(1)}$;
- θ_n requires super-polynomial size proofs in P.

Cook's Programme

Separate NP from coNP (and hence P and NP) by showing super-polynomial lower bounds to the size of proofs in all propositional proof systems.

Showing lower bounds for a system P means

finding an infinite family θ_n of propositional tautologies s.t.

- $|\theta_n| = n^{O(1)}$;
- θ_n requires super-polynomial size proofs in P.
  - Better: ... exponential size proofs.

Cook's Programme

Separate NP from coNP (and hence P and NP) by showing super-polynomial lower bounds to the size of proofs in all propositional proof systems.

Showing lower bounds for a system P means

finding an infinite family θ_n of propositional tautologies s.t.

- $|\theta_n| = n^{O(1)}$;
- θ_n requires super-polynomial size proofs in P.
  - Better: … exponential size proofs.

Even better

- Find a sequence of polynomially constructible formulas which require long proofs.
- This is usually the case: take θ_n as the propositional formalization of some combinatorial principle.
- Find a large set of formulas (e.g. random 3-CNF) which require long proofs.

Cook's Programme

Separate NP from coNP (and hence P and NP) by showing super-polynomial lower bounds to the size of proofs in all propositional proof systems.

Cook's Programme

Separate NP from coNP (and hence P and NP) by showing super-polynomial lower bounds to the size of proofs in all propositional proof systems.

Progress in this programme

- Haken (1985): exponential lower bound to the proof size in Resolution for the pigeonhole principle
- Ajtai (1988): Super-polynomial lower bound for bounded-depth Frege systems (Improved by Beame, Impagliazzo, Krajíček, Pitassi, Pudlák, Woods)
- Lower bounds for algebraic and geometric proof systems:
  - Cutting Planes
  - Polynomial Calculus
  - Nullstellensatz

Cutting Planes

Cutting Planes (CP)

- Cutting Planes uses the idea of linear programming.
- As in Resolution, CP is a refutation system that works with clauses.
- Clauses are translated into linear inequalities.
- CP introduced as a decision procedure for integer linear programming in [Gomory 1953], [Chvátal 1973]
- defined as a proof system by [Cook, Coullard, Turán 1987]

The Translation

- Clauses are translated into linear inequalities

$$a_1 p_1 + \cdots + a_n p_n \geq b \tag{1}$$

  with integer coefficients $a_1, \ldots, a_n$ and b.
- Propositional variables p are identically represented by integer variables p.
- $\neg p$ is translated to $1 - p$.
- A clause

$$C = \{l_1, \ldots, l_n\}$$

  with literals $l_i = p_i$ or $l_i = \neg p_i$ is translated into

$$f_1 + \cdots + f_n \geq 1$$

  with

$$f_i = \begin{cases} p_i & \text{if } l_i = p_i \\ 1 - p_i & \text{if } l_i = \neg p_i \end{cases}$$

  for $i = 1, \ldots, n$.
- To get an inequality of the form (1), constants are moved to the right hand side.

Axioms of CP

1. Let $\Gamma = \{C_1, \ldots, C_k\}$ be a set of clauses in variables $p_1, \ldots, p_n$.
2. As axioms in CP we use the translations of clauses $C_1, \ldots, C_k$ together with

$$p_i \geq 0, \quad -p_i \geq -1 \qquad i = 1, \ldots, n \ .$$

Rules of CP

1. Addition:

$$\frac{a_1 p_1 + \cdots + a_n p_n \geq b \qquad a'_1 p_1 + \cdots + a'_n p_n \geq b'}{(a_1 + a'_1) p_1 + \cdots + (a_n + a'_n) p_n \geq b + b'}$$

2. Multiplication:

$$\frac{a_1 p_1 + \cdots + a_n p_n \geq b}{c a_1 p_1 + \cdots + c a_n p_n \geq cb}$$

with an arbitrary integer $c > 0$.

3. Division:

$$\frac{c a_1 p_1 + \cdots + c a_n p_n \geq b}{a_1 p_1 + \cdots + a_n p_n \geq \left\lceil \frac{b}{c} \right\rceil}$$

with an arbitrary integer $c > 0$.

CP Refutations

- A CP refutation of a set of clauses Γ is a CP derivation of
$$0 \geq 1$$
from the axioms corresponding to Γ.
- We can also consider CP refutations of sets of inequalities, which are not necessarily translations of clauses.

Remarks

- If a set of linear inequalities has a CP refutation, then it does not have any integer solutions.
- It might, however, have fractional solutions.
- Such solutions are eliminated by applications of the division rule.

Complexity of CP

Proposition [Cook, Coullard, Turán 1987]

Cutting Planes p-simulates Resolution.

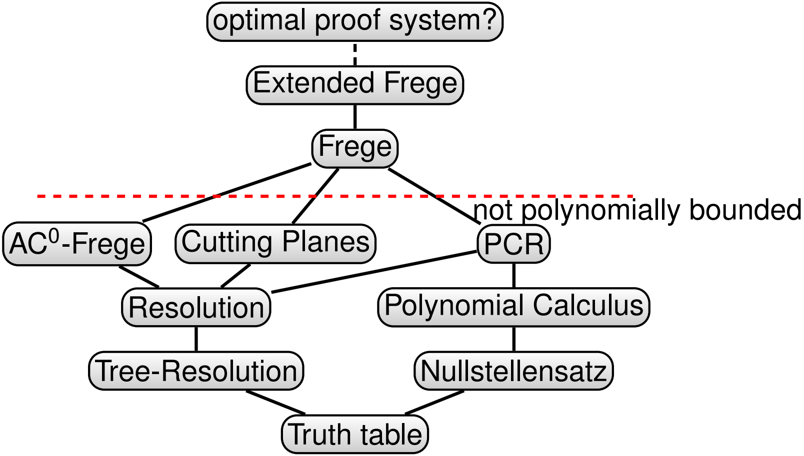
Proposition [Cook, Coullard, Turán 1987]

Cutting Planes has polynomial-size proofs of PHP_n^{n+1}.

Corollary

Resolution does not simulate Cutting Planes.

Simulations between important propositional proof systems



Tree-like Resolution

Tree-like Resolution

Refutation calculus for unsatisfiable CNFs

- Resolution rule $\dfrac{C \cup \{x\} \quad D \cup \{\neg x\}}{C \cup D}$
- tree-like refutations: each derived clause is used at most once

$$\dfrac{\dfrac{\{x_1, x_2\} \quad \{\neg x_1, x_2\}}{\{x_2\}} \qquad \dfrac{\{\neg x_1, \neg x_2\} \quad \{x_1, \neg x_2\}}{\{\neg x_2\}}}{\square}$$

Proof size

- number of clauses in the proof, i.e. number of nodes of the tree
- Goal: lower bounds for the size of tree-like resolution proofs

Tree-like Resolution

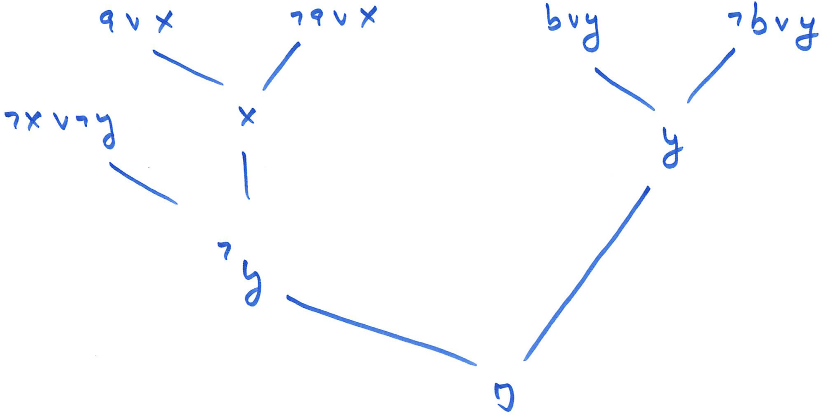
- A resolution derivation can be represented as a directed graph, where nodes are labelled with the clauses of the derivation. For a resolution step

$$\frac{C \quad D}{E}$$

  we insert edges (C, E) and (D, E).
- As the graph is acyclic, resolution is also referred to as dag-like Resolution.
- If the graph is a tree, the proof is called tree-like. If we only allow tree-like proofs, we speak of the tree-like resolution system.
- In tree-like resolution each derived clause can be used at most once as a parent clause in a resolution step.

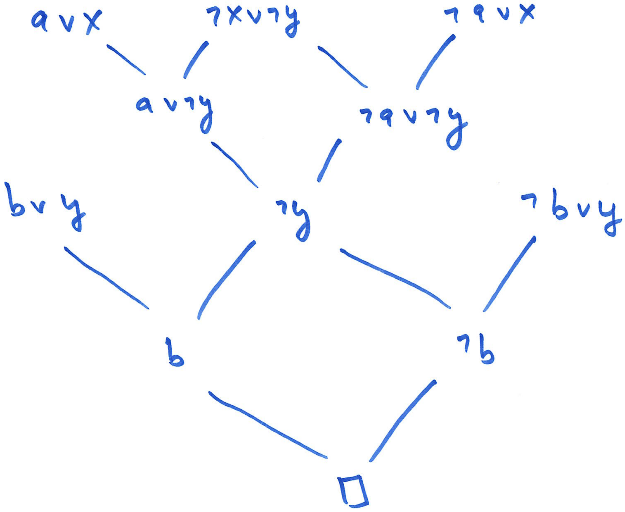
A tree-like refutation

$$F = (a \vee x) \wedge (\neg a \vee x) \wedge (b \vee y) \wedge (\neg b \vee y) \wedge (\neg x \vee \neg y)$$



A non-tree-like refutation (dag-like)

$$F = (a \vee x) \wedge (\neg a \vee x) \wedge (b \vee y) \wedge (\neg b \vee y) \wedge (\neg x \vee \neg y)$$



A game for tree-like Resolution

Prover-Delayer game [Pudlák & Impagliazzo 2000]

- Let F be a set of clauses in n variables $x_1, \ldots, x_n$.
- Prover and Delayer construct a (partial) assignment to $x_1, \ldots, x_n$.
- The game ends as soon as the assignment falsifies a clause of F.
- In each round, Prover chooses a variable x_i. Delayer either chooses a value 0/1 for x_i or leaves the choice to Prover.
- If Prover chooses the value, Delayer scores 1 point.
- Prover can always win on unsatisfiable formulas, but how many points can Delayer earn?

Points and proof length

Intuition

Good Delayer strategies on unsatisfiable CNFs F yield lower bounds for tree-like resolution refutations of F.

Theorem (Pudlák & Impagliazzo 2000)

Let F be an unsatisfiable set of clauses.
If F has tree-like resolution refutations of size at most S, then Delayer can earn at most $\log S$ points in every Prover-Delayer game on F.

Corollary

If Delayer has a strategy that secures him p points in every game on F, then tree-like resolution refutations of F have size $2^{\Omega(p)}$.

The proof

- Let F be an unsatisfiable set of clauses in variables $x_1, \ldots, x_n$ and let π be a tree-like resolution refutation of F.
- Prover and Delayer play a game on F, successively constructing an assignment α.
- Let α_i be the partial assignment constructed after i rounds of the game.
- Let p_i denote Delayer's score after i rounds.
- Let π_{α_i} be the subtree of π whose root is reached by the path specified by α_i.

Invariant during the game

$$|\pi_{\alpha_i}| \leq \frac{|\pi|}{2^{p_i}} \quad \text{for each round } i.$$

Invariant during the game

Invariant

$$|\pi_{\alpha_i}| \leq \frac{|\pi|}{2^{p_i}} \text{ for each round } i.$$

The invariant implies the theorem

- At the end of the game a contradiction is reached and the size of π_α is 1.
- By the invariant we have

$$1 \leq \frac{|\pi|}{2^{p_\alpha}} \quad ,$$

hence $p_\alpha \leq \log |\pi|$.

Invariant during the game

Invariant

$$|\pi_{\alpha_i}| \leq \frac{|\pi|}{2^{p_i}} \text{ for each round } i.$$

Base case

At the start of the game $\pi_{\alpha_0} = \pi$ and Delayer has 0 points. Thus the invariant holds.

Induction step

If Delayer chooses the value, then $p_{i+1} = p_i$ and therefore

$$|\pi_{\alpha_{i+1}}| \leq |\pi_{\alpha_i}| \leq \frac{|\pi|}{2^{p_i}} = \frac{|\pi|}{2^{p_{i+1}}} \quad .$$

Invariant during the game

Induction step

- If Delayer leaves the choice to Prover, then Prover chooses the value of x that leads to the smaller subtree, i.e. Prover sets $x = 0$ if
$$|\pi_{\alpha_i \cup \{x=0\}}| \leq \frac{|\pi_{\alpha_i}|}{2},$$
and $x = 1$ otherwise.
- If Prover sets $x = j$ with $j \in \{0, 1\}$, then
$$|\pi_{\alpha_{i+1}}| = |\pi_{\alpha_i \cup \{x=j\}}| \leq \frac{|\pi_{\alpha_i}|}{2} \leq \frac{|\pi|}{2 \cdot 2^{p_i}} = \frac{|\pi|}{2^{p_i+1}} = \frac{|\pi|}{2^{p_{i+1}}} \quad .$$

A lower bound for the pigeonhole principle

Hardness for (tree-like) Resolution

The first historic lower bound

- Pigeonhole principle: $n + 1$ pigeons do not fit into n holes

Hardness for (tree-like) Resolution

The first historic lower bound

- Pigeonhole principle: $n+1$ pigeons do not fit into n holes

Hardness for (tree-like) Resolution

The first historic lower bound

- Pigeonhole principle: $n+1$ pigeons do not fit into n holes

Dirichlet's box principle, 1834:
If one has a certain number of boxes and puts more objects into the boxes than there are boxes, then some box receives at least two of these objects.

Peter Gustav Lejeune Dirichlet (1805-1859)

Hardness for (tree-like) Resolution

The first historic lower bound

- Pigeonhole principle: $n+1$ pigeons do not fit into n holes

Hardness for (tree-like) Resolution

The first historic lower bound

- Pigeonhole principle: $n+1$ pigeons do not fit into n holes
- CNF formalization PHP_n^{n+1}

$$\bigvee_{j \in [n]} x_{i,j} \qquad \text{for all pigeons } i \in [n+1]$$

$$\neg x_{i_1,j} \vee \neg x_{i_2,j} \qquad \text{for distinct } i_1, i_2 \in [n+1] \text{ and } j \in [n]$$

- PHP_n^{n+1} requires resolution refutations of size $2^{\Omega(n)}$. [Haken 1985]

Hardness for (tree-like) Resolution

The first historic lower bound

- Pigeonhole principle: $n+1$ pigeons do not fit into n holes
- CNF formalization PHP_n^{n+1}

$$\bigvee_{j\in[n]} x_{i,j} \qquad \text{for all pigeons } i \in [n+1]$$

$$\neg x_{i_1,j} \vee \neg x_{i_2,j} \qquad \text{for distinct } i_1, i_2 \in [n+1] \text{ and } j \in [n]$$

- PHP_n^{n+1} requires resolution refutations of size $2^{\Omega(n)}$. [Haken 1985]

Many strong lower bounds

Random 3-CNFs are hard for Resolution.
[Beame et al. 1998]

The pigeonhole principle

- PHP^m_n with $m > n$ uses variables $x_{i,j}$ with $i \in [m]$ and $j \in [n]$,
- $x_{i,j}$ indicates that pigeon i goes into hole j.
- PHP^m_n consists of the clauses

$$\bigvee_{j \in [n]} x_{i,j} \quad \text{for all pigeons } i \in [m]$$

and

$$\neg x_{i_1,j} \vee \neg x_{i_2,j}$$

for all pairwise distinct pigeons $i_1, i_2 \in [m]$ and holes $j \in [n]$.

Tree-like resolution refutations of PHP

- We show that PHP_n^m for $m > n$ is hard for tree-like Resolution.
- A lower bound via the Prover-Delayer game requires a suitable strategy for Delayer.

Theorem

Every tree-like resolution refutation of PHP_n^m for $m > n$ has size $2^{\Omega(n)}$.

Delayer's strategy

We call a hole j occupied if there is an $i \in [m]$ such that $x_{i,j}$ has been set to 1 during the game so far.

Delayer's strategy

We call a hole j occupied if there is an $i \in [m]$ such that $x_{i,j}$ has been set to 1 during the game so far.

Delayer's strategy

If Prover asks a variable $x_{i,j}$, then Delayer answers 0 if hole j is already occupied, otherwise he leaves the choice to Prover.

Delayer's strategy

We call a hole j occupied if there is an $i \in [m]$ such that $x_{i,j}$ has been set to 1 during the game so far.

Delayer's strategy

If Prover asks a variable $x_{i,j}$, then Delayer answers 0 if hole j is already occupied, otherwise he leaves the choice to Prover.

Observation

- The game never ends by falsifying a clause $\neg x_{i_1,j} \vee \neg x_{i_2,j}$.
- Thus the game ends on a big clause $\bigvee_{j \in [n]} x_{i,j}$, i.e. for some $i \in [m]$ all variables $x_{i,j}$ with $j \in [n]$ have been set to 0, either by Prover or by Delayer.

Delayer's score

- For some $i \in [m]$ all variables $x_{i,j}$ with $j \in [n]$ have been set to 0, either by Prover or by Delayer.
- Claim: Delayer earns at least n points in the game.
- If $x_{i,j}$ was set to 0 by Prover, then Delayer earns 1 point.
- If $x_{i,j}$ was set to 0 by Delayer, then by Delayer's strategy there is another pigeon $i' \neq i$ sitting in hole j, i.e. $x_{i',j}$ was set to 1.
- This decision was made by Prover, since Delayer never sets a variable to 1.
- In total, Delayer thus receives one point for each variable $x_{i,j}$ with $j \in [n]$.
- The lower bound follows by the previous theorem.

Complexity of the pigeonhole principle

Theorem

Every tree-like resolution refutation of PHP^m_n with $m > n$ has size $2^{\Omega(n)}$.

Complexity of the pigeonhole principle

Theorem

Every tree-like resolution refutation of PHP_n^m with $m > n$ has size $2^{\Omega(n)}$.

This bound is not optimal.

With the game, a lower bound of size s can only be shown if the graph of every tree-like resolution refutation contains a complete binary tree of depth $\log s$ as a minor.

Theorem (Iwama & Miyazaki 1999)

Every tree-like resolution refutation of PHP_n^m with $m > n$ has size $2^{\Omega(n \log n)}$.

Tree-like versus general Resolution

Tree-like vs. general proofs

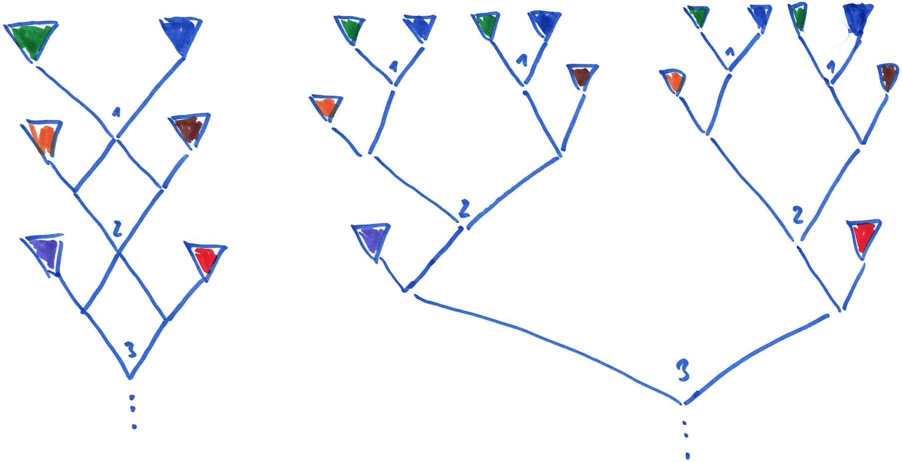
A general question

Are tree-like proofs longer than general proofs (dag-like)?

The answer depends on the proof system.

- For Resolution: general Resolution is strictly stronger than tree-like Resolution (exponential separation).

Transformation from dag-like to tree-like



Tree-like vs. general Resolution

The result

There is a family of unsatisfiable clause sets
that have polynomial-size resolution proofs, but require
exponential-size tree-like resolution proofs.

History

- Goerdt 1992: first separation, with polynomial-size resolution proofs, but only quasi-polynomial tree-like proofs (a version of PHP).
- Bonet, Esteban, Galesi, Johannsen 1998: first exponential separation
- Ben-Sasson, Impagliazzo, Wigderson 2004: simplified separation via games

Separation of tree-like and dag-like Resolution

- separation via pebbling formulas
- uses the pebbling game
- proof method: Prover-Delayer games
- we follow Ben-Sasson, Impagliazzo, Wigderson 2004

Pebbling games

- Pebbling games are played on directed acyclic graphs (dags).
- Source nodes: in-degree 0
- Target nodes: out-degree 0
- Game: place pebbles on nodes according to the rules below
- Goal: to place a pebble on a target node

Rules

1. Source nodes can be pebbled.
2. Any other node can be pebbled if all its parents are pebbled.
3. Pebbles can be removed at any time.

Pebbling number

Complexity measure

Maximal number of pebbles that are placed on the graph simultaneously.

Pebbling number of a pebbling strategy

- Let S be a pebbling strategy for the graph G.
- $P(G, S) =$ max # of pebbles that are on the graph simultaneously when using strategy S

Pebbling number of G

$P(G) = \min\{ P(G, S) \mid S \text{ is a pebbling strategy for } G \}$

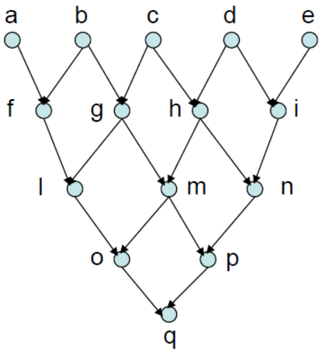
Graphs with high pebbling number

Theorem (Paul, Tarjan, Celoni 1977)

There are graphs G with n nodes and

$$P(G) = \Omega\left(\frac{n}{\log n}\right).$$

- The proof is constructive.
- Example: pyramidal graphs



Pebbling formulas

Directed graph $G = (V, E)$

Propositional variables

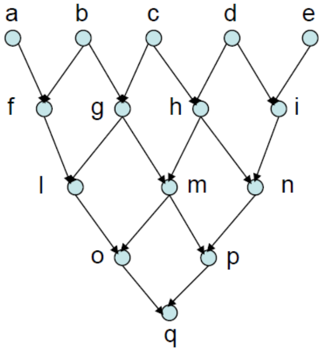
- x_v for all $v \in V$
- Meaning: $x_v = 1$ if v is pebbled

Clauses in $Peb^0(G)$

x_v	for every source node v
$(\bigwedge_{u \in N^-(v)} x_u) \rightarrow x_v$	for all nodes v where $N^-(v)$ are the predecessors of v
$\neg x_v$	for all target nodes v

Complexity of Peb^0

- $Peb^0(G)$ is unsatisfiable.
- But: the formulas have polynomial-size tree-like resolution refutations.
- Idea: start with the target node and use breadth-first search.



Boosting the complexity of $Peb^0(G)$

New idea

- Use pebbles with two different colours: black and white
- Nodes can be pebbled with black and white pebbles.

The new principle

- Source nodes can always be pebbled black or white.
- An inner node can be pebbled black or white if all its predecessors are pebbled black or white.
- No target node is pebbled black or white.

The new formulas

Directed graph $G = (V, E)$ with in-degree ≤ 2

Propositional variables

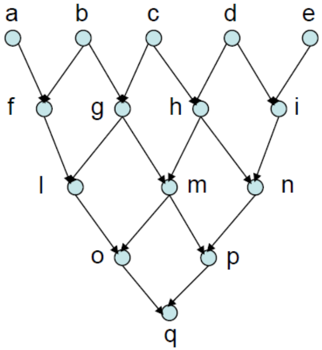
- $x_{v,c}$ for all $v \in V$ and $c \in \{B, W\}$
- Meaning: $x_{v,B} = 1$ if v is pebbled black
  $x_{v,W} = 1$ if v is pebbled white

Clauses in $Peb(G)$

$x_{v,B} \vee x_{v,W}$	for all source nodes v
$x_{u,a} \wedge x_{w,b} \rightarrow x_{v,B} \vee x_{v,W}$	for $v \in V$, $a, b \in \{B, W\}$, where u, w are v's predecessors
$\neg x_{v,B}, \neg x_{v,W}$	for all target nodes v

Complexity of *Peb*(G)

- *Peb*(G) is unsatisfiable.
- The earlier proof strategy for Peb^0 does not work any more.
- But: the formulas have polynomial-size general resolution refutations.
- Goal: lower bound for tree-like Resolution



The pebbling formulas in tree-like Resolution

Theorem

Let G be a directed graph with in-degree ≤ 2. Then Delayer has a strategy that earns him $P(G) - 3$ points in every Prover-Delayer game on $Peb(G)$.

Theorem (Paul, Tarjan, Celoni 1977)

There are graphs G with n nodes and

$$P(G) = \Omega\left(\frac{n}{\log n}\right).$$

Corollary

There are graphs G with n nodes such that $Peb(G)$ *requires tree-like resolution refutations of size* $2^{\Omega\left(\frac{n}{\log n}\right)}$.

Proof of the theorem

Let G be a directed graph with source nodes S and target nodes T.

Delayer strategy

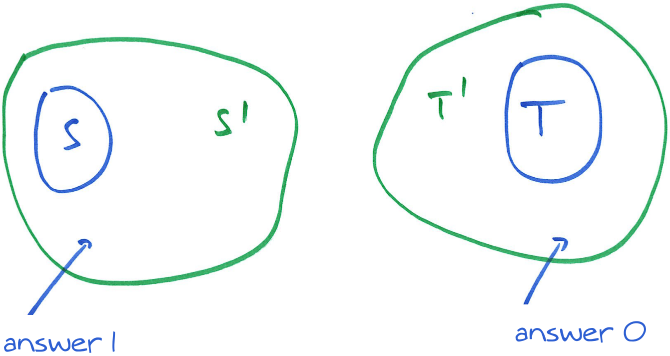
- Use two sets S' and T'.
- Initially set $S' = S$ and $T' = T$.
- Let $P(G, S', T')$ be the pebbling number of G with source nodes S' and target nodes T'.
- If Prover asks the variable $x_{v,\cdot}$ for the node v, then Delayer reacts as follows:
  1. If $v \in S'$, answer 1.
  2. If $v \in T'$, answer 0.
  3. If $v \notin S' \cup T'$ and $P(G, S', T' \cup \{v\}) = P(G, S', T')$, answer 0 and set $T' = T' \cup \{v\}$.
  4. If $v \notin S' \cup T'$ and $P(G, S', T' \cup \{v\}) < P(G, S', T')$, leave the decision to Prover and set $S' = S' \cup \{v\}$.

Intuition for the strategy

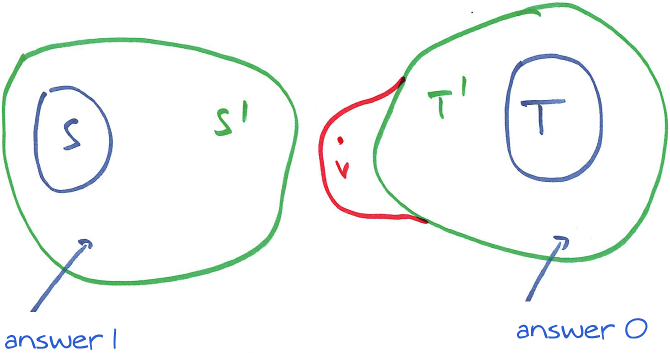
If Prover asks the variable $x_{v,\cdot}$ for the node v, then Delayer reacts as follows:

1. If $v \in S'$, answer 1.
   Source nodes are always pebbled.
2. If $v \in T'$, answer 0.
   Target nodes are never pebbled.
3. If $v \notin S' \cup T'$ and $P(G, S', T' \cup \{v\}) = P(G, S', T')$, answer 0 and set $T' = T' \cup \{v\}$.
   If the pebbling number stays the same, add v to T' and do not pebble it.
4. If $v \notin S' \cup T'$ and $P(G, S', T' \cup \{v\}) < P(G, S', T')$, leave the decision to Prover and set $S' = S' \cup \{v\}$.
   If the pebbling number drops, add v to S'.
   Prover pays, but can only choose the colour of v.

Delayer's strategy



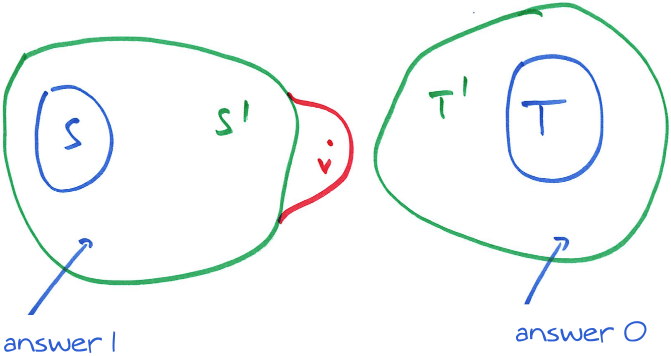
Delayer's strategy



Case 3: $P(G, S', T' \cup \{v\}) = P(G, S', T')$

Delayer decides

Delayer's strategy



Case 4: $P(G, S', T' \cup \{v\}) < P(G, S', T')$

Prover decides, but still $v \in S'$

How many points does Delayer earn?

Intuition

- When the pebbling number drops, Delayer earns a point.
- Consequently, Delayer earns proportionally to the pebbling number of G.

Lemma 1

When the game ends, $P(G, S', T') \leq 3$.

Lemma 2

For all nodes v and sets S, T

$$P(G, S, T) \leq \max\{P(G, S, T \cup \{v\}), P(G, S \cup \{v\}, T) + 1\}.$$

How many points does Delayer earn?

Lemma 1
When the game ends, $P(G, S', T') \leq 3$.

Lemma 2
For all nodes v and sets S, T

$$P(G, S, T) \leq \max\{P(G, S, T \cup \{v\}), P(G, S \cup \{v\}, T) + 1\}.$$

Lemma 3
After every round: if Delayer has p points, then $P(G, S', T') \geq P(G, S, T) - p$.

Corollary
Delayer earns at least $P(G, S, T) - 3$ points.

The result

Theorem

There is an infinite family of explicitly constructible formulas θ_n such that

1. $|\theta_n| = O(n)$;
2. θ_n *requires tree-like resolution refutations of size* $2^{\Omega\left(\frac{n}{\log n}\right)}$.

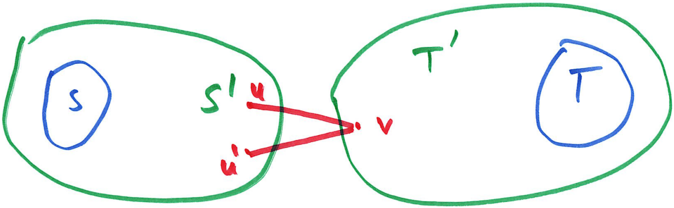
How many points does Delayer earn?

Lemma 1

When the game ends, $P(G, S', T') \leq 3$.

Proof

- During the game no source or target axioms are violated.
- The game therefore ends on a pebbling axiom with nodes u, u', v that is falsified.
- Then $u, u' \in S'$ and $v \in T'$.
- Hence 3 pebbles are needed.



How many points does Delayer earn?

Lemma 2

For all nodes v and sets S, T

$$P(G, S, T) \leq \max\{P(G, S, T \cup \{v\}), P(G, S \cup \{v\}, T) + 1\}.$$

Proof

We describe a pebbling strategy for (G, S, T).

- First use the pebbling strategy for $(G, S, T \cup \{v\})$.
- This needs $P(G, S, T \cup \{v\})$ pebbles.
- If a pebble lies on a node of T, we are done.
- Otherwise a pebble lies on v.
- Remove all pebbles except this pebble on v.
- Now use the pebbling strategy for $(G, S \cup \{v\}, T)$.
- This requires $P(G, S \cup \{v\}, T) + 1$ pebbles.

Lemma 3

After every round: if Delayer has p points, then $P(G, S', T') \geq P(G, S, T) - p$.

Proof

by induction on the number of rounds of the game

Base case: $p = 0$ and $S' = S$, $T' = T$

Induction step: Delayer plays according to cases 1-3 $\Rightarrow$ claim holds.

Delayer plays according to case 4 $\Rightarrow$ at the start of the round we have

$$P(G, S', T' \cup \{v\}) < P(G, S', T') \tag{2}$$

Lemma 2 $\Rightarrow$

$$P(G, S', T') \leq \max\{P(G, S', T' \cup \{v\}), P(G, S' \cup \{v\}, T') + 1\} \tag{3}$$

(2) & (3) $\Rightarrow P(G, S', T') \leq P(G, S' \cup \{v\}, T') + 1$

$\Rightarrow P(G, S' \cup \{v\}, T') \geq P(G, S', T') - 1 \underbrace{\geq}_{IH} P(G, S, T) - (p + 1)$

Delayer earns 1 point $\Rightarrow$ claim holds.

The result

Theorem

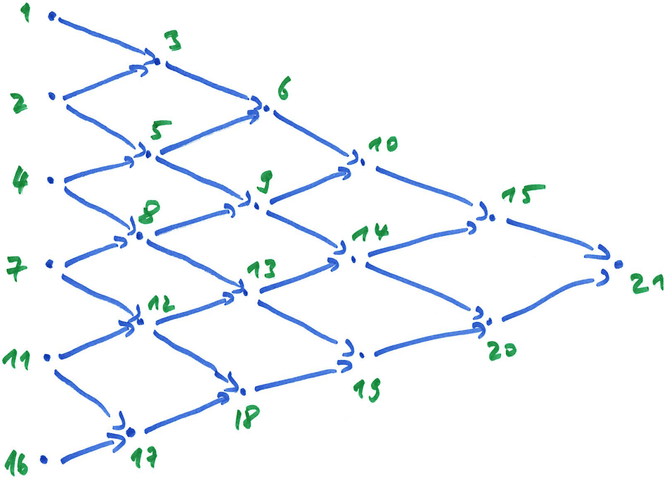
There is an infinite family of explicitly constructible formulas θ_n such that

1. $|\theta_n| = O(n)$;
2. θ_n *requires tree-like resolution refutations of size* $2^{\Omega\left(\frac{n}{\log n}\right)}$;
3. θ_n *has resolution refutations of size* $O(n)$.

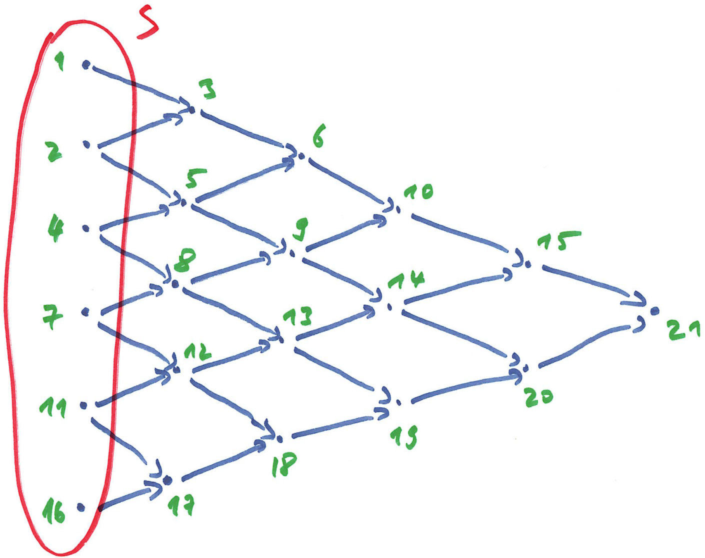
Linear-size resolution proofs of the pebbling formulas

- Fix a topological order of G.
- Inductively derive the clauses $x_{v,B} \vee x_{v,W}$ following this order.
- If v has no predecessors, then $v \in S$ and $x_{v,B} \vee x_{v,W}$ is an axiom.
- If v has two predecessors u, w, then by induction we have already derived $x_{u,B} \vee x_{u,W}$ and $x_{w,B} \vee x_{w,W}$.
- Together with the four pebbling axioms for v this implies $x_{v,B} \vee x_{v,W}$.
- By the completeness of resolution there is a resolution derivation of $x_{v,B} \vee x_{v,W}$ from these clauses.
- As only 6 variables occur, this derivation has constant size.
- Thus we derive $x_{t,B} \vee x_{t,W}$ for a target node $t \in T$.
- With the target node axioms we obtain a contradiction.

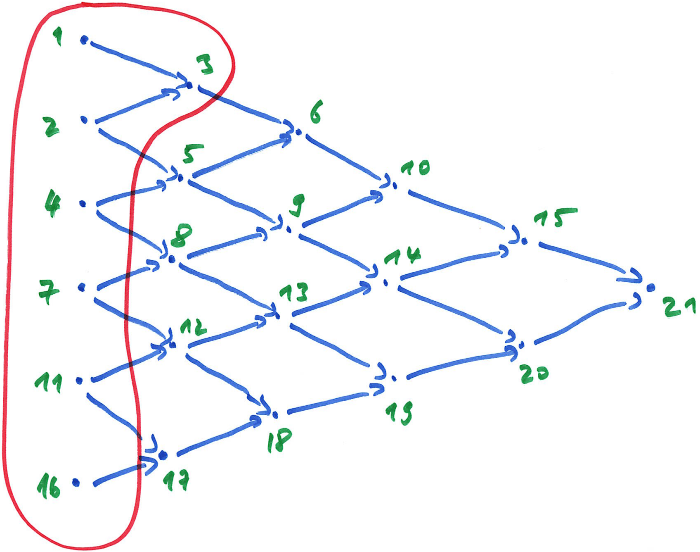
Topological sorting



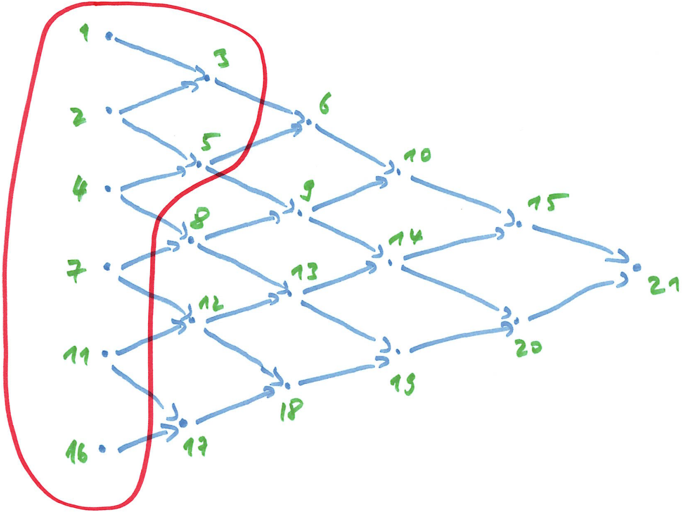
Linear-size resolution proofs of the pebbling formulas



Linear-size resolution proofs of the pebbling formulas



Linear-size resolution proofs of the pebbling formulas



Lower bounds for general Resolution

Resolution with weakening

Resolution

In the following we consider resolution with the weakening rules

$$\frac{C}{C \cup \{x\}} \qquad\qquad \frac{C}{C \cup \{\neg x\}}$$

for arbitrary clauses C and variables x.

Remarks

- Resolution with weakening is complete, i.e. for all CNFs Γ and clauses C we have
  $\Gamma \models C$ *if and only if* $\Gamma \vdash_{Res} C$.
- Using the weakening rule does not shorten proofs.

Measures for Resolution

Let Γ be a set of clauses and A a clause.

- We write $\pi : \Gamma \vdash A$ if π is a resolution proof of A from Γ.
- For a resolution proof $\pi = (C_1, \ldots, C_k)$, $|\pi| = k$ is the number of clauses in π.
- minimal size

$$S(\Gamma \vdash A) = \min\{|\pi| \mid \pi : \Gamma \vdash A\}.$$

- minimal tree size

$$S_T(\Gamma \vdash A) = \min\{|\pi| \mid \pi : \Gamma \vdash A \text{ and } \pi \text{ is tree-like}\}.$$

Measures for Resolution

- The width of a clause C is

$$w(C) = |C| \ .$$

- The width of a set of clauses Γ is

$$w(\Gamma) = \max\{w(C) \mid C \in \Gamma\}$$

- For a resolution proof $\pi = (C_1, \ldots, C_k)$, $w(\pi) = \max\{w(C_i) \mid i = 1, \ldots, k\}$.
- The minimal width of a resolution derivation of A from Γ is

$$w(\Gamma \vdash A) = \min\{w(\pi) \mid \pi : \Gamma \vdash A\} \ .$$

- We write $\Gamma \vdash_k A$ as an abbreviation for $w(\Gamma \vdash A) \leq k$.

Restrictions and width

Definition

Let C be a clause and $a \in \{0, 1\}$. The restriction of C by $x = a$ is defined as

$$C|_{x=a} = \begin{cases} C & \text{if } x \notin C \text{ and } \neg x \notin C \\ 1 & \text{if } x^a \in C \\ C \setminus \{x^{1-a}\} & \text{if } x^{1-a} \in C. \end{cases}$$

Lemma 1

Let Γ be a set of clauses and $a \in \{0, 1\}$. Then

$$\Gamma|_{x=a} \vdash_w A \quad \Longrightarrow \quad \Gamma \vdash_{w+1} A \cup \{x^{1-a}\}.$$

Two important lemmas

Lemma 1

Let Γ be a set of clauses and $a \in \{0,1\}$. Then

$$\Gamma|_{x=a} \vdash_w A \quad \Longrightarrow \quad \Gamma \vdash_{w+1} A \cup \{x^{1-a}\}.$$

Lemma 2

Let Γ be a set of clauses, $a \in \{0,1\}$ and $\Gamma_{x^a} = \{C \in \Gamma \mid x^a \in C\}$. Then

$$\Gamma|_{x=a} \vdash_{k-1} \Box, \;\; \Gamma|_{x=1-a} \vdash_k \Box \quad \Longrightarrow \quad w(\Gamma \vdash \Box) \leq \max\{k, w(\Gamma_{x^a})\}.$$

Proof of Lemma 1

Lemma 1

$\Gamma|_{x=a} \vdash_w A \quad \Longrightarrow \quad \Gamma \vdash_{w+1} A \cup \{x^{1-a}\}$

Proof

- Let π be a resolution proof of A from $\Gamma|_{x=a}$ with $w(\pi) \leq w$.
- Add the literal x^{1-a} to every clause of π. We check that this yields a proof π' of $A \cup \{x^{1-a}\}$ from Γ. Let C be a clause of π:
- If C is an axiom of $\Gamma|_{x=a}$ that arose from some $C_0 \in \Gamma$ by removing x^{1-a}, then $C \cup \{x^{1-a}\} = C_0 \in \Gamma$.
- If C is an axiom with $C \in \Gamma$ and $x \notin var(C)$, then derive $C \cup \{x^{1-a}\}$ from C by one weakening step.
- Every resolution step of π remains a valid resolution step (the pivot variable is never x); every weakening step remains a weakening step.
- Each clause grows by at most one literal, so $w(\pi') \leq w + 1$. $\square$

Proof of Lemma 2

Lemma 2

$\Gamma|_{x=a} \vdash_{k-1} \Box, \;\; \Gamma|_{x=1-a} \vdash_k \Box \;\; \Longrightarrow \;\; w(\Gamma \vdash \Box) \leq \max\{k, w(\Gamma_{x^a})\}$

Proof (derive $\Box$ from Γ in three stages)

1. Applying Lemma 1 to $\Gamma|_{x=a} \vdash_{k-1} \Box$ gives a derivation of the unit clause $\{x^{1-a}\}$ from Γ of width $\leq k$:

$$\Gamma \vdash_k \{x^{1-a}\} \quad .$$

2. Resolving $\{x^{1-a}\}$ with each clause $C \cup \{x^a\} \in \Gamma_{x^a}$ (where $x^a \notin C$) yields C. Together with the clauses of Γ not containing x, this derives every clause of $\Gamma|_{x=1-a}$; these steps have width $\leq w(\Gamma_{x^a})$.
3. Now run the given width-k refutation of $\Gamma|_{x=1-a}$.

In total: $w(\Gamma \vdash \Box) \leq \max\{k, w(\Gamma_{x^a})\}$. $\Box$

Width and size for tree-like Resolution

Theorem [Ben-Sasson & Wigderson 2001]

For every unsatisfiable set of clauses Γ we have

$$w(\Gamma \vdash \Box) \leq w(\Gamma) + \log S_T(\Gamma \vdash \Box) \quad .$$

Corollary

For every unsatisfiable set of clauses Γ we have

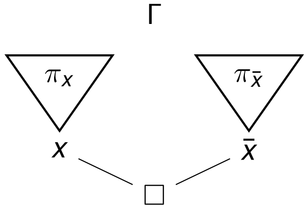
$$S_T(\Gamma \vdash \Box) \geq 2^{w(\Gamma \vdash \Box) - w(\Gamma)} \quad .$$

Proof of the size-width relation (tree-like case)

Claim (induction on n and b)

If $S_T(\Gamma \vdash \Box) \leq 2^b$, then $w(\Gamma \vdash \Box) \leq w(\Gamma) + b$.

- The claim gives the theorem for $b = \log S_T(\Gamma \vdash \Box)$ (rounded up).
- Base case $b = 0$: then $S_T(\Gamma \vdash \Box) \leq 1$, so $\Box \in \Gamma$.
- Induction step: let $\pi : \Gamma \vdash \Box$ be tree-like, $|\pi| \leq 2^b$. The last step of π resolves some variable x from the unit clauses $\{x\}$ and $\{\bar{x}\}$, derived by subtrees π_x and $\pi_{\bar{x}}$:



- $|\pi_x| + |\pi_{\bar{x}}| + 1 = |\pi| \leq 2^b$, so w.l.o.g. $|\pi_x| \leq 2^{b-1}$.

Proof of the size-width relation (tree-like case)

- Restricting every clause of π_x by $x = 0$ turns the derived unit clause $\{x\}$ into $\Box$:
  this yields a tree-like refutation of $\Gamma|_{x=0}$ of size $\leq |\pi_x| \leq 2^{b-1}$.
- Likewise, $\pi_{\bar{x}}$ restricted by $x = 1$ is a tree-like refutation of $\Gamma|_{x=1}$ of size $\leq 2^b$.

Proof of the size-width relation (tree-like case)

- Restricting every clause of π_x by $x = 0$ turns the derived unit clause $\{x\}$ into $\Box$:
  this yields a tree-like refutation of $\Gamma|_{x=0}$ of size $\leq |\pi_x| \leq 2^{b-1}$.
- Likewise, $\pi_{\bar{x}}$ restricted by $x = 1$ is a tree-like refutation of $\Gamma|_{x=1}$ of size $\leq 2^b$.
- Induction on b:

$$w(\Gamma|_{x=0} \vdash \Box) \;\leq\; w(\Gamma|_{x=0}) + b - 1 \;\leq\; w(\Gamma) + b - 1 \quad .$$

- Induction on n ($\Gamma|_{x=1}$ has one variable less):

$$w(\Gamma|_{x=1} \vdash \Box) \;\leq\; w(\Gamma|_{x=1}) + b \;\leq\; w(\Gamma) + b \quad .$$

Proof of the size-width relation (tree-like case)

- Restricting every clause of π_x by $x = 0$ turns the derived unit clause $\{x\}$ into $\Box$:
  this yields a tree-like refutation of $\Gamma|_{x=0}$ of size $\leq |\pi_x| \leq 2^{b-1}$.
- Likewise, $\pi_{\bar{x}}$ restricted by $x = 1$ is a tree-like refutation of $\Gamma|_{x=1}$ of size $\leq 2^b$.
- Induction on b:

$$w(\Gamma|_{x=0} \vdash \Box) \;\leq\; w(\Gamma|_{x=0}) + b - 1 \;\leq\; w(\Gamma) + b - 1 \;.$$

- Induction on n ($\Gamma|_{x=1}$ has one variable less):

$$w(\Gamma|_{x=1} \vdash \Box) \;\leq\; w(\Gamma|_{x=1}) + b \;\leq\; w(\Gamma) + b \;.$$

- Lemma 2 (with $a = 0$ and $k = w(\Gamma) + b$), using $w(\Gamma_{x^0}) \leq w(\Gamma)$:

$$w(\Gamma \vdash \Box) \;\leq\; \max\{\, w(\Gamma){+}b,\; w(\Gamma_{x^0}) \,\} \;=\; w(\Gamma){+}b \;. \qquad \Box$$

Width and size for general Resolution

Theorem [Ben-Sasson & Wigderson 2001]

For every unsatisfiable set of clauses Γ in n variables we have

$$w(\Gamma \vdash \Box) \leq w(\Gamma) + O(\sqrt{n \log S(\Gamma \vdash \Box)}) \ .$$

Corollary

For every unsatisfiable set of clauses Γ in n variables we have

$$S(\Gamma \vdash \Box) = \exp\left(\Omega\left(\frac{(w(\Gamma \vdash \Box) - w(\Gamma))^2}{n}\right)\right) \ .$$

Proof of the size-width relation (general case)

Setup: fat clauses [Ben-Sasson & Wigderson 2001]

- Let $\pi : \Gamma \vdash \Box$ be a refutation of minimal size $|\pi| = S := S(\Gamma \vdash \Box)$,
  and let n be the number of variables of Γ.
- Fix the width threshold and an auxiliary base

$$d := \sqrt{2n \ln S} \qquad \text{and} \qquad a := \left(1 - \frac{d}{2n}\right)^{-1}.$$

- A clause C is fat if $w(C) > d$. $\quad \pi^* := \{C \in \pi \mid w(C) > d\}$

Proof of the size-width relation (general case)

Setup: fat clauses [Ben-Sasson & Wigderson 2001]

- Let $\pi : \Gamma \vdash \Box$ be a refutation of minimal size
  $|\pi| = S := S(\Gamma \vdash \Box)$,
  and let n be the number of variables of Γ.
- Fix the width threshold and an auxiliary base

$$d := \sqrt{2n \ln S} \qquad \text{and} \qquad a := \left(1 - \frac{d}{2n}\right)^{-1}.$$

- A clause C is fat if $w(C) > d$. $\quad \pi^* := \{C \in \pi \mid w(C) > d\}$

Claim (induction on n and b)

If Γ has a refutation π with $|\pi^*| < a^b$ fat clauses, then

$$w(\Gamma \vdash \Box) \ \leq \ w(\Gamma) + d + b \ .$$

Proof of the size-width relation (general case)

Setup: fat clauses [Ben-Sasson & Wigderson 2001]

- Let $\pi : \Gamma \vdash \Box$ be a refutation of minimal size $|\pi| = S := S(\Gamma \vdash \Box)$, and let n be the number of variables of Γ.
- Fix the width threshold and an auxiliary base

$$d := \sqrt{2n \ln S} \qquad \text{and} \qquad a := \left(1 - \frac{d}{2n}\right)^{-1} .$$

- A clause C is fat if $w(C) > d$. $\quad \pi^* := \{C \in \pi \mid w(C) > d\}$

Claim (induction on n and b)

If Γ has a refutation π with $|\pi^*| < a^b$ fat clauses, then

$$w(\Gamma \vdash \Box) \ \leq \ w(\Gamma) + d + b \ .$$

Base case $b = 0$: $\pi^* = \emptyset$, so $w(\Gamma \vdash \Box) \leq d$.

Proof of the size-width relation (general case)

Induction step

- The fat clauses contain $> d \cdot |\pi^*|$ literal occurrences, but there are only $2n$ literals.
- Averaging (a pigeonhole argument!): some literal $x^{a'}$ occurs in at least $\frac{d}{2n} \cdot |\pi^*|$ fat clauses.
- The restriction $x = a'$ satisfies all these clauses: $\pi|_{x=a'}$ is a refutation of $\Gamma|_{x=a'}$ with fewer than $\left(1 - \frac{d}{2n}\right) a^b = a^{b-1}$ fat clauses.
- Induction on b: $\quad w(\Gamma|_{x=a'} \vdash \Box) \leq w(\Gamma) + d + b - 1$.
- Induction on n: $\quad w(\Gamma|_{x=1-a'} \vdash \Box) \leq w(\Gamma) + d + b$.
- Lemma 2, using $w(\Gamma_{x^{a'}}) \leq w(\Gamma)$:

$$w(\Gamma \vdash \Box) \;\leq\; \max\{\, w(\Gamma)+d+b,\; w(\Gamma_{x^{a'}})\,\} \;=\; w(\Gamma)+d+b \;.$$

Proof of the size-width relation (general case)

Choosing the parameters

- For $b := d$ we get, using $(1-z)^{-1} \geq e^z$ for $0 \leq z < 1$,

$$a^d = \left(1 - \frac{d}{2n}\right)^{-d} \geq e^{\frac{d^2}{2n}} = e^{\ln S} = S \ .$$

- Since $\Box \in \pi$ is not fat: $|\pi^*| < |\pi| = S \leq a^d$.
- So the Claim applies with $b = d$ and yields

$$\begin{aligned} w(\Gamma \vdash \Box) &\leq w(\Gamma) + 2d = w(\Gamma) + 2\sqrt{2n \ln S} \\ &= w(\Gamma) + O\left(\sqrt{n \log S}\right) . \end{aligned}$$

$\Box$

Proof of the size-width relation (general case)

Choosing the parameters

- For $b := d$ we get, using $(1-z)^{-1} \geq e^z$ for $0 \leq z < 1$,

$$a^d \;=\; \left(1 - \frac{d}{2n}\right)^{-d} \;\geq\; e^{\frac{d^2}{2n}} \;=\; e^{\ln S} \;=\; S\;.$$

- Since $\Box \in \pi$ is not fat: $|\pi^*| < |\pi| = S \leq a^d$.
- So the Claim applies with $b = d$ and yields

$$\begin{aligned} w(\Gamma \vdash \Box) \;&\leq\; w(\Gamma) + 2d \;=\; w(\Gamma) + 2\sqrt{2n \ln S} \\ &=\; w(\Gamma) + O\Big(\sqrt{n \log S}\Big)\,. \qquad \Box \end{aligned}$$

Remark

The same two lemmas drive both proofs: refute both restrictions $\Gamma|_{x=0}$ and $\Gamma|_{x=1}$ in small width, then recombine via Lemma 2.

Lower bounds

For lower bounds for resolution we need an infinite sequence Γ_n of unsatisfiable clause sets with the following properties:

1. Γ_n uses n propositional variables.
2. The width of Γ_n is constant: $w(\Gamma_n) = O(1)$.
3. The refutation width of Γ_n is linear: $w(\Gamma_n \vdash \Box) = \Omega(n)$.

Application to

Tseitin formulas

Tseitin formulas

- Let $G = (V, E)$ be a graph and $f : V \to \{0, 1\}$ a function.
- We define the weight of f as

$$\sum_{v \in V} f(v) \ .$$

- For each edge $e \in E$ we introduce a propositional variable x_e.
- The variable x_e plays the role of an edge weight, which can take the value 0 or 1 for each edge.
- For a node $v \in V$ let

$$Parity_v : \quad \sum_{u \in V, \{u,v\} \in E} x_{\{u,v\}} \equiv f(v) \pmod 2 \ .$$

- The Tseitin formula for G and f is

$$\tau(G, f) = \bigwedge_{v \in V} Parity_v .$$

Tseitin formulas as CNFs

Tseitin formulas

- For a node $v \in V$ let

$$\mathit{Parity}_v : \quad \sum_{u \in V,\, \{u,v\} \in E} x_{\{u,v\}} \equiv f(v) \pmod 2 .$$

- The Tseitin formula for G and f is

$$\tau(G,f) = \bigwedge_{v \in V} \mathit{Parity}_v .$$

Remark

If d is the degree of G, then $\tau(G,f)$ can be represented as a d-CNF with $\leq n2^{d-1}$ clauses and $\leq \frac{nd}{2}$ variables (exercise).

Tseitin formulas as contradictions

Tseitin formulas

- For a node $v \in V$ let

$$Parity_v : \sum_{u \in V,\, \{u,v\} \in E} x_{\{u,v\}} \equiv f(v) \pmod 2 .$$

- The Tseitin formula for G and f is

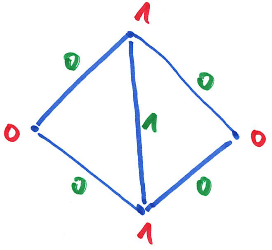
$$\tau(G, f) = \bigwedge_{v \in V} Parity_v .$$

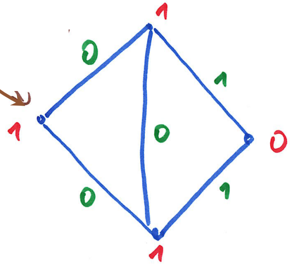
Lemma

For a connected graph $G = (V, E)$, $\tau(G, f)$ is unsatisfiable if and only if f has odd weight.

Graph for Tseitin formulas – example

parity condition violated





node weights

$f(G) = 2$

$f(G) = 3$

edge weights: formula $\tau(G, f)$ satisfiable

formula $\tau(G, f)$ unsatisfiable

Expander graphs

Definition

- Let $G = (V, E)$ be a graph.
- The expansion of a graph is defined as

  $$e(G) = \min\{|E(V', V \setminus V')| : V' \subseteq V,\ \tfrac{1}{3}|V| \leq |V'| \leq \tfrac{2}{3}|V|\}$$

  with

  $$E(V', V'') = \{\{u, v\} \in E \mid u \in V', v \in V''\}$$

  for $V', V'' \subseteq V$ and $V' \cap V'' = \emptyset$.
- G is called an expander if $e(G) = \Omega(n)$ with $n = |V|$.

Theorem

For every $n \geq 1$ there are 3-regular connected expanders with n nodes.

Width and size of Tseitin formulas

Theorem [Ben-Sasson & Wigderson 2001]

For every connected graph $G = (V, E)$ and every function $f : V \to \{0, 1\}$ of odd weight we have

$$w(\tau(G, f) \vdash \Box) \geq e(G) \ .$$

Corollary [Urquhart 1987]

If $G_n = (V_n, E_n)$ is a sequence of 3-regular connected expanders with $|V_n| = n$ and $f_n : V_n \to \{0, 1\}$ a sequence of functions of odd weight, then

$$S(\tau(G_n, f_n) \vdash \Box) = 2^{\Omega(|\tau(G_n, f_n)|)} \ .$$

Proof strategy

For $\Gamma \vdash \Box$ define a measure

$$\mu : \text{clauses} \to \mathbb{N}$$

with:

1. $\mu(\text{axiom}) \leq 1 \quad$ (i.e. $\mu(C) \leq 1$ for $C \in \Gamma$)
2. $\mu(\Box)$ is large.
3. Show: every resolution refutation of Γ contains a clause C with $\mu(C)$ of medium size.
4. Show: $\mu(C)$ of medium size $\Rightarrow w(C)$ large

The measure μ for Tseitin formulas

Let $G = (V, E)$ be connected with $|V| = n$ and let f have odd weight.

For $V' \subseteq V$ let $\quad A_{V'} = \{\, \mathit{Parity}_v \mid v \in V' \,\}$.

Definition

For a clause C let $\quad \mu(C) = \min\{\, |V'| \;:\; V' \subseteq V \text{ and } A_{V'} \models C \,\}$.

The measure μ for Tseitin formulas

Let $G = (V, E)$ be connected with $|V| = n$ and let f have odd weight.

For $V' \subseteq V$ let $\quad A_{V'} = \{\, Parity_v \mid v \in V' \,\}$.

Definition

For a clause C let $\quad \mu(C) = \min\{\, |V'| \;:\; V' \subseteq V \text{ and } A_{V'} \models C \,\}$.

Properties

- $\mu(C) \leq 1$ for every axiom C of $\tau(G, f)$: each clause of $\tau(G, f)$ belongs to the CNF of a single $Parity_v$, so $A_{\{v\}} \models C$.
- μ is subadditive: for a resolution step $\dfrac{C \cup \{x\} \quad D \cup \{\bar{x}\}}{C \cup D}$ we have

$$\mu(C \cup D) \leq \mu(C \cup \{x\}) + \mu(D \cup \{\bar{x}\})$$

(take the union of the two witness sets); for weakening, $\mu(C \cup \{x\}) \leq \mu(C)$.

$\mu(\Box) = n$

Claim

$\mu(\Box) = n$, i.e. for every proper subset $V' \subsetneq V$ the set $A_{V'}$ is satisfiable.

Proof

- Let $V' \subsetneq V$ and choose a node $v \in V \setminus V'$.
- Define
$$f'(u) = \begin{cases} 1 - f(u) & u = v \\ f(u) & \text{otherwise.} \end{cases}$$
- f' has even weight, so $\tau(G, f')$ is satisfiable (by the lemma on Tseitin formulas as contradictions – here connectivity of G is used).
- $\tau(G, f)$ and $\tau(G, f')$ differ only in $Parity_v$, and $v \notin V'$: a satisfying assignment of $\tau(G, f')$ thus satisfies $A_{V'}$. $\Box$

A clause of medium complexity

Claim

Every resolution refutation π of $\tau(G, f)$ contains a clause C with

$$\frac{n}{3} \leq \mu(C) \leq \frac{2n}{3} .$$

Proof

- $\mu(\Box) = n \geq \frac{n}{3}$, so π contains clauses of μ-value $\geq \frac{n}{3}$.
- Let C be the first clause in the sequence π with $\mu(C) \geq \frac{n}{3}$.
- C is no axiom ($\mu \leq 1$) and no weakening (weakening does not increase μ), so C is a resolvent.
- Both parents of C appear earlier in π, hence have μ-value $< \frac{n}{3}$.
- Subadditivity: $\mu(C) < \frac{n}{3} + \frac{n}{3} = \frac{2n}{3}$. $\square$

Boundary variables must occur in C

For $V' \subseteq V$ let $\quad \partial A_{V'} = \{\, x_e \mid e \in E(V', V \setminus V') \,\}$
(the variables of the boundary edges of V').

Claim

Let C be a clause with $\frac{n}{3} \leq \mu(C) \leq \frac{2n}{3}$ and let $V' \subseteq V$ be a minimal set with $A_{V'} \models C$. Then every variable of $\partial A_{V'}$ occurs in C.

Boundary variables must occur in C

For $V' \subseteq V$ let $\quad \partial A_{V'} = \{\, x_e \mid e \in E(V', V \setminus V') \,\}$
(the variables of the boundary edges of V').

Claim

Let C be a clause with $\frac{n}{3} \leq \mu(C) \leq \frac{2n}{3}$ and let $V' \subseteq V$ be a minimal set with $A_{V'} \models C$. Then every variable of $\partial A_{V'}$ occurs in C.

The claim proves the theorem

$|V'| = \mu(C)$, so $\frac{1}{3}|V| \leq |V'| \leq \frac{2}{3}|V|$ and therefore

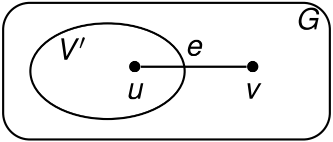
$$w(C) \;\geq\; |\partial A_{V'}| \;=\; |E(V', V \setminus V')| \;\geq\; e(G) \;.$$

Hence $w(\tau(G, f) \vdash \Box) \geq e(G)$. $\Box$

Proof of the claim

Suppose $x_e \in \partial A_{V'}$ with $e = \{u, v\}$, $u \in V'$, $v \notin V'$, but x_e does not occur in C.

- Minimality of V': $A_{V' \setminus \{u\}} \not\models C$, so there is an assignment α with $\alpha \models A_{V' \setminus \{u\}}$ and $\alpha \not\models C$; since $A_{V'} \models C$, necessarily $\mathit{Parity}_u(\alpha) = 0$.
- Flip α on x_e and obtain α'.
- x_e occurs in no $\mathit{Parity}_{v'}$ with $v' \in V' \setminus \{u\}$ (its endpoints are u and $v \notin V'$): still $\alpha' \models A_{V' \setminus \{u\}}$.
- $\mathit{Parity}_u(\alpha') = 1$, so $\alpha' \models A_{V'}$, and hence $C(\alpha') = 1$.
- But x_e does not occur in C, so $C(\alpha') = C(\alpha) = 0$ – a contradiction. □



From width to size: proof of the corollary

Let G_n be 3-regular connected expanders, i.e. $e(G_n) = \Omega(n)$, and let f_n have odd weight.

- The theorem gives $w(\tau(G_n, f_n) \vdash \Box) \geq e(G_n) = \Omega(n)$.
- $\tau(G_n, f_n)$ is a 3-CNF with $\leq 4n$ clauses over $\frac{3n}{2}$ variables: $w(\tau(G_n, f_n)) = 3$ and $|\tau(G_n, f_n)| \leq 12n$.
- The size-width relation (over $\frac{3n}{2}$ variables) yields

$$S(\tau(G_n, f_n) \vdash \Box) \;=\; \exp\left(\Omega\left(\frac{(\Omega(n) - 3)^2}{3n/2}\right)\right) \;=\; \exp(\Omega(n)) \;.$$

- Since $|\tau(G_n, f_n)| = \Theta(n)$, this is $2^{\Omega(|\tau(G_n, f_n)|)}$: exponential in the size of the formula. $\Box$

Application to

the pigeonhole principle

Recall: the pigeonhole principle

- PHP^m_n with $m > n$ uses variables $x_{i,j}$ with $i \in [m]$, $j \in [n]$.
- $x_{i,j}$ says that pigeon i sits in hole j.
- PHP^m_n contains the clauses

$$\bigvee_{j \in [n]} x_{i,j} \quad \text{for all pigeons } i \in [m]$$

and $\qquad \neg x_{i_1,j} \vee \neg x_{i_2,j}$

for all pairwise distinct pigeons $i_1, i_2 \in [m]$ and holes $j \in [n]$.

Problem

- The number of variables of PHP^m_n is $mn > n^2$.
- With $w(PHP^m_n \vdash \Box) = \Omega(n)$, the size-width relation gives no nontrivial lower bounds for $S(PHP^m_n \vdash \Box)$.

A modification of PHP with fewer variables

Graph pigeonhole principle

- Let $G = (V \dot\cup U, E)$ be a bipartite graph with $|V| = m$ and $|U| = n$.
- We use variables x_e for the edges $e \in E$.
- *G-PHP* uses the clauses

$$P_v = \bigvee_{v \in e} x_e \quad \text{for } v \in V$$

and

$$H^u_{v,v'} = \neg x_e \vee \neg x_{e'}$$

for $e = \{v, u\}$, $e' = \{v', u\}$, $v, v' \in V$, $u \in U$, $v \neq v'$.

The graph pigeonhole principle

G-PHP generalizes *PHP*

- Let $K_{m,n}$ be the complete bipartite graph with $V = [m]$ and $U = [n]$.
- Then $PHP^m_n = K_{m,n}\text{-}PHP$.

Lemma

Let G and G' be bipartite graphs with $E \subseteq E'$. Then

$$S(G\text{-}PHP \vdash \square) \leq S(G'\text{-}PHP \vdash \square).$$

Bipartite graphs for *G-PHP*

Definition

- For $u \in U$ the set of neighbours is

$$N(u) = \{v \in V \,:\, \{u, v\} \in E\}.$$

- For $V' \subseteq V$ the boundary is

$$\delta(V') = \{u \in U \,:\, |N(u) \cap V'| = 1\}$$

- Intuition: boundary(pigeons) $=$ $\{$holes : only one pigeon wants to go into this hole$\}$
- G is an (r, s, c)-expander if $d(v) \leq s$ for all $v \in V$ and

$$V' \subseteq V, \quad |V'| \leq r \quad \Rightarrow \quad |\delta(V')| \geq c|V'|.$$

Lower bounds for PHP

Definition

G is an (r, s, c)-expander if $d(v) \leq s$ for all $v \in V$ and

$$V' \subseteq V, \quad |V'| \leq r \quad \Rightarrow \quad |\delta(V')| \geq c|V'|.$$

Theorem [Ben-Sasson & Wigderson 2001]

Every bipartite (r, s, c)-expander with $c \geq 1$ yields

$$w(G\text{-}PHP \vdash \Box) \geq \frac{rc}{2}.$$

Remark

There are bipartite graphs $G = (V \dot\cup U, E)$ with $|V| = n + 1$ and $|U| = n$ that are $(n/d_0, 5, 1)$-expanders for a constant $d_0 > 1$.

Corollary [Haken 1985]

$S(PHP_n^{n+1} \vdash \Box) = 2^{\Omega(n)}$

From width to size: proof of the corollary

Let $G = (V \dot\cup U, E)$ be an $(n/d_0, 5, 1)$-expander with $|V| = n+1$, $|U| = n$.

- The theorem gives $w(G\text{-}PHP \vdash \Box) \geq \frac{n}{2d_0} = \Omega(n)$.
- $G\text{-}PHP$ is small and narrow: it has $|E| \leq 5(n+1)$ variables and $w(G\text{-}PHP) \leq 5$ (pigeon clauses P_v have width $d(v) \leq 5$).
- The size-width relation (over $\leq 5(n+1)$ variables) yields

$$S(G\text{-}PHP \vdash \Box) \;=\; \exp\left(\Omega\left(\frac{(\Omega(n)-5)^2}{5(n+1)}\right)\right) \;=\; \exp(\Omega(n)) \;.$$

- $G \subseteq K_{n+1,n}$, so the lemma gives

$$S(PHP_n^{n+1} \vdash \Box) \;\geq\; S(G\text{-}PHP \vdash \Box) \;=\; 2^{\Omega(n)} \;. \qquad \Box$$

- The width bound is shown with the same strategy as for Tseitin formulas, using the measure $\mu(C) = \min \#$pigeon clauses P_v that imply C.

Summary

- Proof systems [Cook, Reckhow 1979]:
  TAUT has a polynomially bounded proof system iff NP $=$ coNP.
- Resolution is refutationally complete; simulation order:
  Tree-Resolution $\leq$ Resolution $\leq$ Cutting Planes $\leq$ Frege
- Prover-Delayer games [Pudlák & Impagliazzo 2000]:
  p points for Delayer force tree-like refutations of size $\geq 2^p$,
  e.g. $2^{\Omega(n)}$ for PHP_n^{n+1}.
- Pebbling formulas: dag-like size $O(n)$, tree-like size $2^{\Omega(n/\log n)}$ – separating tree-like from general Resolution [Ben-Sasson, Impagliazzo, Wigderson 2004].
- Size-width relation [Ben-Sasson & Wigderson 2001]:
  short proofs are narrow:
  $w(\Gamma \vdash \Box) \leq w(\Gamma) + O\left(\sqrt{n \log S(\Gamma \vdash \Box)}\right)$.
- Applications: width lower bounds on expanders
  give exponential size for Tseitin formulas [Urquhart 1987]
  and PHP_n^{n+1} [Haken 1985].