Mini-Tutorial on Weak Proof Systems and Connections to SAT Solving

Jakob Nordström

KTH Royal Institute of Technology
Stockholm, Sweden

Theoretical Foundations of Applied SAT Solving
Banff International Research Station
January 19–24, 2014

Focus of This Mini-Tutorial

Proof systems behind some current approaches to SAT solving:

- Conflict-driven clause learning — resolution
- Gröbner basis computations — polynomial calculus
- Pseudo-Boolean solvers — cutting planes

Survey (some of) what is known about these proof systems

Show some of the "benchmark formulas" used

By necessity, selective and somewhat subjective coverage — apologies in advance for omissions

Outline

1. Resolution
   - Preliminaries
   - Length, Width and Space
   - Complexity Measures and CDCL Hardness
2. Stronger Proof Systems Than Resolution
   - Polynomial Calculus
   - Cutting Planes
   - And Beyond...
3. CDCL and Efficient Proof Search

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 $F = C_1 \land \cdots \land C_m$: conjunction of clauses
- k-CNF formula: CNF formula with clauses of size $\leq k$
  (where k is some constant)
- Mostly assume formulas k-CNFs (for simplicity of exposition)
  Conversion to 3-CNF (most often) doesn't change much
- N denotes size of formula (# literals, which is $\approx$ # clauses)

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
- DAG

Tree-like resolution if DAG is tree

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.	$\perp$	Res$(7,8)$

The Resolution Proof System

Goal: refute unsatisfiable CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

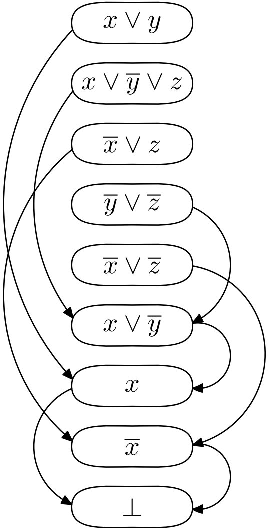
$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\bot$ derived

Can represent refutation as

- annotated list or
- DAG

Tree-like resolution if DAG is tree



The Resolution Proof System

Goal: refute unsatisfiable CNF

Start with clauses of formula (axioms)

Derive new clauses by resolution rule

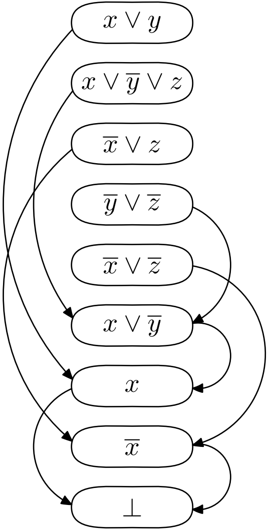
$$\frac{C \lor x \qquad D \lor \overline{x}}{C \lor D}$$

Refutation ends when empty clause $\bot$ derived

Can represent refutation as

- annotated list or
- DAG

Tree-like resolution if DAG is tree



Resolution Size/Length

Size/length = # clauses in refutation

Most fundamental measure in proof complexity

Lower bound on CDCL running time

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

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

Examples of Hard Formulas w.r.t Resolution Length (1/2)

Pigeonhole principle (PHP) [Hak85]

"$n+1$ pigeons don't fit into n holes"

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

Can also add "functionality" and "onto" axioms

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

Even Onto-FPHP formula is hard for resolution

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

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

Tseitin formulas [Urq87]

"Sum of degrees of vertices in graph is even"

- Let variables = edges
- Label every vertex 0/1 so that sum of labels odd
- Write CNF requiring parity of edges around vertex = label

Requires length $\exp(\Omega(N))$ on well-connected so-called expanders

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

Tseitin formulas [Urq87]

"Sum of degrees of vertices in graph is even"

- Let variables $=$ edges
- Label every vertex $0/1$ so that sum of labels odd
- Write CNF requiring parity of edges around vertex $=$ label

Requires length $\exp(\Omega(N))$ on well-connected so-called expanders

Random k-CNF formulas [CS88]

Randomly sample Δn k-clauses over n variables
($\Delta \gtrsim 4.5$ sufficient for $k = 3$ to get unsatisfiable CNF w.h.p.)
Again lower bound $\exp(\Omega(N))$

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 $\Rightarrow$ length upper bound

Proof: at most $(2 \cdot \#\text{variables})^{\text{width}}$ distinct clauses
(This simple counting argument is essentially tight [ALN13])

Resolution Width

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

Width upper bound $\Rightarrow$ length upper bound

Proof: at most $(2 \cdot \#\text{variables})^{\text{width}}$ distinct clauses
(This simple counting argument is essentially tight [ALN13])

Width lower bound $\Rightarrow$ length lower bound

Theorem ([BW01])

$$\textit{width} \leq \mathcal{O}\left(\sqrt{(\textit{formula size } N) \cdot \log(\textit{length})}\right)$$

Resolution Width

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

Width upper bound $\Rightarrow$ length upper bound

Proof: at most $(2 \cdot \#\text{variables})^{\text{width}}$ distinct clauses
(This simple counting argument is essentially tight [ALN13])

Width lower bound $\Rightarrow$ length lower bound

Theorem ([BW01])

$$\mathit{width} \leq \mathcal{O}\left(\sqrt{(\mathit{formula\ size}\ N) \cdot \log(\mathit{length})}\right)$$

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

Optimality of the Length-Width Lower Bound

For tree-like resolution have width $\leq \mathcal{O}(\log(\text{length}))$ [BW01]

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

Optimality of the Length-Width Lower Bound

For tree-like resolution have width $\leq \mathcal{O}(\log(\text{length}))$ [BW01]

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

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

$\overline{x}_{i,j} \lor \overline{x}_{j,i}$	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

Can also add "total order" axioms

$x_{i,j} \lor x_{j,i}$	totality; either $e_i < e_j$ or $e_j < e_i$

Doable in length $\mathcal{O}(N)$ but needs width $\Omega\left(\sqrt[3]{N}\right)$ (3-CNF version)

Resolution Space

Space = max # clauses in memory when performing refutation

Motivated by considerations of SAT solver memory usage

Also intrinsically interesting for proof complexity

Can be measured in different ways — focus here on most common measure clause space

Space at step t: # clauses at steps $\leq t$ used at steps $\geq t$

Example: Space at step 7 ...

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$(7,8)$

Resolution Space

Space = max # clauses in memory when performing refutation

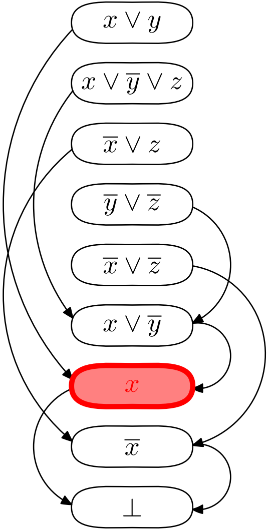
Motivated by considerations of SAT solver memory usage

Also intrinsically interesting for proof complexity

Can be measured in different ways — focus here on most common measure clause space

Space at step t: # clauses at steps $\leq t$ used at steps $\geq t$

Example: Space at step 7 . . .



Resolution Space

Space = max # clauses in memory when performing refutation

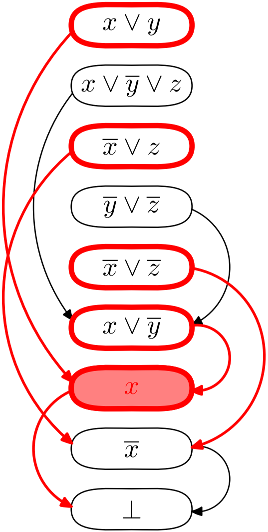
Motivated by considerations of SAT solver memory usage

Also intrinsically interesting for proof complexity

Can be measured in different ways — focus here on most common measure clause space

Space at step t: # clauses at steps $\leq t$ used at steps $\geq t$

Example: Space at step 7 is 5



Bounds on Resolution Space

Space always at most $N + \mathcal{O}(1)$ [ET01]

Lower bounds for

- Pigeonhole principle [ABRW02, ET01]
- Tseitin formulas [ABRW02, ET01]
- Random k-CNFs [BG03]

Bounds on Resolution Space

Space always at most $N + \mathcal{O}(1)$ [ET01]

Lower bounds for

- Pigeonhole principle [ABRW02, ET01]
- Tseitin formulas [ABRW02, ET01]
- Random k-CNFs [BG03]

Results always matching width bounds

And proofs of very similar flavour... What is going on?

Space vs. Width

Theorem ([AD08])

$$\mathit{space} \geq \mathit{width} + \mathcal{O}(1)$$

Space vs. Width

Theorem ([AD08])

$$\mathit{space} \geq \mathit{width} + \mathcal{O}(1)$$

Are space and width asymptotically always the same? No!

Space vs. Width

Theorem ([AD08])

$$\textit{space} \geq \textit{width} + \mathcal{O}(1)$$

Are space and width asymptotically always the same? No!

Pebbling formulas [BN08]

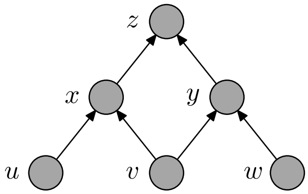
- Can be refuted in width $\mathcal{O}(1)$
- May require space $\Omega(N/\log N)$

A bit more involved to describe than previous benchmarks. . .

Pebbling Formulas: Vanilla Version

CNF formulas encoding so-called pebble games on DAGs

1. u
2. v
3. w
4. $\overline{u} \lor \overline{v} \lor x$
5. $\overline{v} \lor \overline{w} \lor y$
6. $\overline{x} \lor \overline{y} \lor z$
7. $\overline{z}$

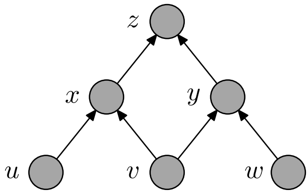


- sources are true
- truth propagates upwards
- but sink is false

Pebbling Formulas: Vanilla Version

CNF formulas encoding so-called pebble games on DAGs

1. u
2. v
3. w
4. $\overline{u} \lor \overline{v} \lor x$
5. $\overline{v} \lor \overline{w} \lor y$
6. $\overline{x} \lor \overline{y} \lor z$
7. $\overline{z}$

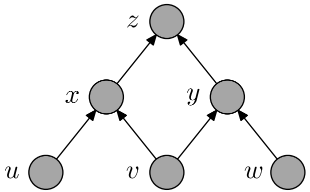


- sources are true
- truth propagates upwards
- but sink is false

Pebbling Formulas: Vanilla Version

CNF formulas encoding so-called pebble games on DAGs

1. u
2. v
3. w
4. $\overline{u} \lor \overline{v} \lor x$
5. $\overline{v} \lor \overline{w} \lor y$
6. $\overline{x} \lor \overline{y} \lor z$
7. $\overline{z}$

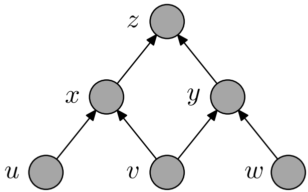


- sources are true
- truth propagates upwards
- but sink is false

Pebbling Formulas: Vanilla Version

CNF formulas encoding so-called pebble games on DAGs

1. u
2. v
3. w
4. $\overline{u} \lor \overline{v} \lor x$
5. $\overline{v} \lor \overline{w} \lor y$
6. $\overline{x} \lor \overline{y} \lor z$
7. $\overline{z}$

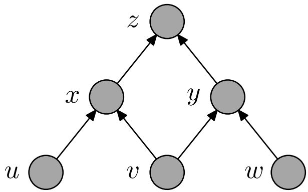


- sources are true
- truth propagates upwards
- but sink is false

Pebbling Formulas: Vanilla Version

CNF formulas encoding so-called pebble games on DAGs

1. u
2. v
3. w
4. $\overline{u} \lor \overline{v} \lor x$
5. $\overline{v} \lor \overline{w} \lor y$
6. $\overline{x} \lor \overline{y} \lor z$
7. $\overline{z}$



- sources are true
- truth propagates upwards
- but sink is false

Extensive literature on pebbling space and time-space trade-offs from 1970s and 80s

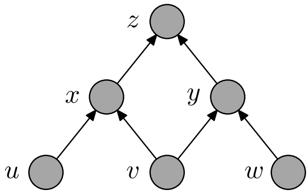
Have been useful in proof complexity before in various contexts

Hope that pebbling properties of DAG somehow carry over to resolution refutations of pebbling formulas.

Pebbling Formulas: Vanilla Version

CNF formulas encoding so-called pebble games on DAGs

1. u
2. v
3. w
4. $\overline{u} \lor \overline{v} \lor x$
5. $\overline{v} \lor \overline{w} \lor y$
6. $\overline{x} \lor \overline{y} \lor z$
7. $\overline{z}$



- sources are true
- truth propagates upwards
- but sink is false

Extensive literature on pebbling space and time-space trade-offs from 1970s and 80s

Have been useful in proof complexity before in various contexts

Hope that pebbling properties of DAG somehow carry over to resolution refutations of pebbling formulas. **Except. . .**

Substituted Pebbling Formulas

Won't work — solved by unit propagation, so supereasy

Make formula harder by substituting $x_1 \oplus x_2$ for every variable x (also works for other Boolean functions with "right" properties):

$$\overline{x} \vee y$$

$$\Downarrow$$

$$\neg(x_1 \oplus x_2) \vee (y_1 \oplus y_2)$$

$$\Downarrow$$

$$\begin{aligned} & (x_1 \vee \overline{x}_2 \vee y_1 \vee y_2) \\ \wedge\ & (x_1 \vee \overline{x}_2 \vee \overline{y}_1 \vee \overline{y}_2) \\ \wedge\ & (\overline{x}_1 \vee x_2 \vee y_1 \vee y_2) \\ \wedge\ & (\overline{x}_1 \vee x_2 \vee \overline{y}_1 \vee \overline{y}_2) \end{aligned}$$

Now CNF formula inherits pebbling graph properties!

Space-Width Trade-offs

Given a formula easy w.r.t. these complexity measures, can refutations be optimized for two or more measures?

For space vs. width, the answer is a strong no

Theorem ([Ben09])

There are formulas for which

- *exist refutations in width $\mathcal{O}(1)$*
- *exist refutations in space $\mathcal{O}(1)$*
- *optimization of one measure causes (essentially) worst-case behaviour for other measure*

Holds for vanilla version of pebbling formulas

Length-Space Trade-offs

Theorem ([BN11, BBI12, BNT13])

There are formulas for which

- *exist refutations in short length*
- *exist refutations in small space*
- *optimization of one measure* causes *dramatic blow-up for other measure*

Holds for

- Substituted pebbling formulas over the right graphs
- Tseitin formulas over long, narrow rectangular grids

So no meaningful simultaneous optimization possible for length and space in the worst case

Length-Width Trade-offs?

What about length versus width?

[BW01] transforms short refutation to narrow one, but blows up length exponentially

- Is this blow-up inherent?
- Or just an artifact of the proof?

Open Problem

Are there length-width trade-offs in resolution? Or can we search for a narrow refutation and be sure to find something not significantly longer than the shortest one?

Do These Measures Say Anything About CDCL Hardness?

Recall log(length) $\lesssim$ width $\lesssim$ space

Do These Measures Say Anything About CDCL Hardness?

Recall log(length) $\lesssim$ width $\lesssim$ space

Length

- Lower bound on running time for CDCL
- But short refutations may be intractable to find [AR08]

Do These Measures Say Anything About CDCL Hardness?

Recall log(length) $\lesssim$ width $\lesssim$ space

Length

- Lower bound on running time for CDCL
- But short refutations may be intractable to find [AR08]

Width

- Searching in small width known heuristic in AI community
- Small width $\Rightarrow$ CDCL solver will provably be fast [AFT11]

Do These Measures Say Anything About CDCL Hardness?

Recall log(length) $\lesssim$ width $\lesssim$ space

Length

- Lower bound on running time for CDCL
- But short refutations may be intractable to find [AR08]

Width

- Searching in small width known heuristic in AI community
- Small width $\Rightarrow$ CDCL solver will provably be fast [AFT11]

Space

- In practice, memory consumption important bottleneck
- Does space complexity correlate with hardness?

Practical Conclusions?

Experimental evaluation

- Proposed by [ABLM08]
- First(?) systematic attempt in [JMNŽ12]
- No firm conclusions — other structural properties involved?
- Ongoing work — so far both width and space seem relevant

Practical Conclusions?

Experimental evaluation

- Proposed by [ABLM08]
- First(?) systematic attempt in [JMNŽ12]
- No firm conclusions — other structural properties involved?
- Ongoing work — so far both width and space seem relevant

Broader lessons?

Performance on combinatorial benchmarks sometimes surprising

- For PHP, worse behaviour with heuristics than without
- For ordering principles, highly dependent on specific solver

Open Problem

- *Could it be interesting to explain the above phenomena?*
- *Could controlled experiments on easily scalable theoretical benchmarks yield other interesting insights?*

Polynomial Calculus (or Actually PCR)

Introduced in [CEI96]; below modified version from [ABRW02]

Clauses interpreted as polynomial equations over finite field
Any field in theory; GF(2) in practice
Example: $x \vee y \vee \overline{z}$ gets translated to $x'y'z = 0$

Derivation rules

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

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

Size, Degree and Space

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

Size, Degree and Space

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

Size — analogue of resolution length
total # monomials in refutation (counted with repetitions)
Can also define length measure — might be much smaller

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

(Monomial) space — analogue of resolution (clause) space
max # monomials in memory during refutation (with repetitions)

Polynomial Calculus Strictly Stronger than Resolution

Polynomial calculus simulates resolution efficiently with respect to length/size, width/degree, and space simultaneously

- Can mimic resolution refutation step by step
- Hence worst-case upper bounds for resolution carry over

Polynomial Calculus Strictly Stronger than Resolution

Polynomial calculus simulates resolution efficiently with respect to length/size, width/degree, and space simultaneously

- Can mimic resolution refutation step by step
- Hence worst-case upper bounds for resolution carry over

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

- Tseitin formulas on expanders (just do Gaussian elimination)
- Onto functional pigeonhole principle [Rii93]

Polynomial Calculus Strictly Stronger than Resolution

Polynomial calculus simulates resolution efficiently with respect to length/size, width/degree, and space simultaneously

- Can mimic resolution refutation step by step
- Hence worst-case upper bounds for resolution carry over

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

- Tseitin formulas on expanders (just do Gaussian elimination)
- Onto functional pigeonhole principle [Rii93]

Open Problem

Show that polynomial calculus is strictly stronger than resolution w.r.t. space

Size vs. Degree

- Degree upper bound $\Rightarrow$ size upper bound [CEI96]
  Qualitatively similar to resolution bound
  A bit more involved argument
  Again essentially tight by [ALN13]

- Degree lower bound $\Rightarrow$ size lower bound [IPS99]
  Precursor of [BW01] — can do same proof to get same bound

- Size-degree lower bound essentially optimal [GL10]
  Example: again ordering principle formulas

- Most size lower bounds for polynomial calculus derived via degree lower bounds (but machinery less developed)

Examples of Hard Formulas w.r.t. Size (and Degree)

Pigeonhole principle formulas
Follows from [AR03]
Earlier work on other encodings in [Raz98, IPS99]

Tseitin formulas with "wrong modulus"
Can define Tseitin-like formulas counting mod p for $p \neq 2$
Hard if $p \neq$ characteristic of field [BGIP01]

Random k-CNF formulas
Hard in all characteristics except 2 [BI10] (conference version '99)
Lower bound for all characteristics in [AR03]

Bounds on Polynomial Calculus Space

Lower bound for PHP with wide clauses [ABRW02]

k-CNFs much trickier — sequence of lower bounds for

- Obfuscated 4-CNF versions of PHP [FLN+12]
- Random 4-CNFs [BG13]
- Tseitin formulas on (some) expanders [FLM+13]

Open Problem

- *Prove tight space lower bounds for Tseitin on any expander*
- *Prove any space lower bound on random 3-CNFs*
- *Prove any space lower bound for any 3-CNF!?*

Space vs. Degree

Open Problem (analogue of [AD08])

Is it true that space $\geq$ degree $+ \mathcal{O}(1)$?

Partial progress: if formula requires large resolution width, then XOR-substituted version requires large space [FLM+13]

Space vs. Degree

Open Problem (analogue of [AD08])

Is it true that space $\geq$ degree $+ \mathcal{O}(1)$?

Partial progress: if formula requires large resolution width, then XOR-substituted version requires large space [FLM^{+}13]

Optimal separation of space and degree in [FLM^{+}13] by flavour of Tseitin formulas which

- can be refuted in degree $\mathcal{O}(1)$
- require space $\Omega(N)$
- but separating formulas depend on characteristic of field

Open Problem

Prove space lower bounds for substituted pebbling formulas (would give space-degree separation independent of characteristic)

Trade-offs for Polynomial Calculus

- Strong, essentially optimal space-degree trade-offs [BNT13]
  Same vanilla pebbling formulas as for resolution
  Same parameters

- Strong size-space trade-offs [BNT13]
  Same formulas as for resolution
  Some loss in parameters

Open Problem

Are there size-degree trade-offs in polynomial calculus?

Algebraic SAT Solvers?

- Quite some excitement about Gröbner basis approach to SAT solving after [CEI96]
- Promise of performance improvement failed to deliver
- Meanwhile: the CDCL revolution...
- Is it harder to build good algebraic SAT solvers, or is it just that too little work has been done (or both)?
- Some shortcut seems to be needed — full Gröbner basis computation does too much work
- Priyank Kalla will give survey talk about algebraic approaches to SAT on Tuesday

Cutting Planes

Introduced in [CCT87]

Clauses interpreted as linear inequalities over the reals with integer coefficients

Example: $x \vee y \vee \overline{z}$ gets translated to $x + y + (1 - z) \geq 1$

Derivation rules

Variable axioms $\dfrac{}{0 \leq x \leq 1}$

Multiplication $\dfrac{\sum a_i x_i \geq A}{\sum c a_i x_i \geq cA}$

Addition $\dfrac{\sum a_i x_i \geq A \quad \sum b_i x_i \geq B}{\sum (a_i + b_i) x_i \geq A + B}$

Division $\dfrac{\sum c a_i x_i \geq A}{\sum a_i x_i \geq \lceil A/c \rceil}$

Goal: Derive $0 \geq 1 \Leftrightarrow$ formula unsatisfiable

Size, Length and Space

Length = total # lines/inequalities in refutation

Size = sum also size of coefficients

Space = max # lines in memory during refutation

No (useful) analogue of width/degree

Size, Length and Space

Length = total # lines/inequalities in refutation

Size = sum also size of coefficients

Space = max # lines in memory during refutation

No (useful) analogue of width/degree

Cutting planes

- simulates resolution efficiently w.r.t. length/size and space simultaneously
- is strictly stronger w.r.t. length/size — can refute PHP efficiently [CCT87]

Open Problem

Show cutting planes strictly stronger than resolution w.r.t. space

Hard Formulas w.r.t Cutting Planes Length

Clique-coclique formulas [Pud97]
"A graph with a k-clique is not $(k-1)$-colourable"
Lower bound via interpolation and circuit complexity

Open Problem

Prove cutting planes length lower bounds

- *for Tseitin formulas*
- *for random k-CNFs*
- *for any formula using other technique than interpolation*

Hard Formulas w.r.t Cutting Planes Space?

No space lower bounds known except conditional ones

All short cutting planes refutations of

- Tseitin formulas on expanders require large space [GP13]
  (But such short refutations probably don't exist anyway)

- (some) pebbling formulas require large space [HN12, GP13]
  (and such short refutations do exist; hard to see how exponential length could help bring down space)

Above results obtained via communication complexity

No (true) length-space trade-off results known
Although results above can also be phrased as trade-offs

Geometric SAT Solvers?

- Some work on pseudo-Boolean solvers using (subset of) cutting planes
- Seems hard to make competitive with CDCL on CNFs
- One key problem to recover cardinality constraints
- Daniel Le Berre will give survey talk about geometric approaches to SAT on Tuesday

Semialgebraic Proof Systems

- Proof systems using polynomial inequalities over the reals
- Kind of a combination/generalization of polynomial calculus and cutting planes
- Used to reason about (near-)optimality of combinatorial optimization
- Albert Atserias will give a separate mini-tutorial about semialgebraic proof systems on Tuesday

How Efficient Resolution Refutations Can CDCL Find?

DPLL (no clause learning)

Always yields tree-like refutations

Exponentially weaker than general resolution in worst case

How Efficient Resolution Refutations Can CDCL Find?

DPLL (no clause learning)

Always yields tree-like refutations

Exponentially weaker than general resolution in worst case

CDCL

Generates DAG-like refutations, but with very particular structure

- Clauses derived by "input resolution" w.r.t. clause database
- Learned clauses should be asserting
- Derivations look locally regular w.r.t. clause database
  (only resolve on each variable once along path)

Can CDCL be as efficient as general, unrestricted resolution?

How Measure Efficiency? CDCL as a Proof System

1. **Automatizability**
   - Run in time polynomial in smallest possible refutation
   - Seems too strict a requirement even for resolution [AR08]

2. **More relaxed notion**
   - Can CDCL run in time polynomial in smallest possible refutation assuming that all free decisions are made optimally?
   - I.e., does CDCL polynomially simulate resolution viewed as a proof system?
   - Intuitively: No worst-case guarantees, but promise to work well if one can get heuristics right

CDCL Polynomially Simulates Resolution

Answer: yes, polynomial simulation! [BKS04, BHJ08, HBPV08]
But with varying restrictions on model:

- Non-standard learning schemes
- Decisions flipping propagated variables
- Decisions past conflicts
- Preprocessing of formula (with new variables)

CDCL Polynomially Simulates Resolution

Answer: yes, polynomial simulation! [BKS04, BHJ08, HBPV08]
But with varying restrictions on model:

- Non-standard learning schemes
- Decisions flipping propagated variables
- Decisions past conflicts
- Preprocessing of formula (with new variables)

Theorem ([PD11])

Natural model of CDCL polynomially simulates resolution

CDCL Polynomially Simulates Resolution

Answer: yes, polynomial simulation! [BKS04, BHJ08, HBPV08]
But with varying restrictions on model:

- Non-standard learning schemes
- Decisions flipping propagated variables
- Decisions past conflicts
- Preprocessing of formula (with new variables)

Theorem ([PD11])

Natural model of CDCL polynomially simulates resolution

Theorem ([AFT11])

If in addition resolution width is small, then CDCL solver with enough randomness will find good refutation with high probability

Assumptions Behind Effectiveness of CDCL

1. **Frequent restarts**
   How efficient is CDCL without restarts?
   Can it simulate resolution or not?

2. **Never forget clauses**
   Not how CDCL solvers actually operate
   Just technical condition or necessary for proofs to go through?

3. **Randomness**
   Not used much in practice
   Seems necessary for theoretical results in [AFT11]

Further Questions About CDCL Proof System

- Possible to get more "syntactic" description of proof system in [AFT11, PD11]? (Now more like execution trace of solver)
- Can one model (clause database) space in such a proof system in some nice way?
- Do upper and lower bounds and trade-offs results carry over from general resolution?

Summing up

- Survey of resolution, polynomial calculus and cutting planes
- Resolution fairly well understood
- Polynomial calculus less so
- Cutting planes almost not at all
- Could there be interesting connections between proof complexity measures and hardness of SAT?
- How can we build efficient SAT solvers on stronger proof systems than resolution?

Summing up

- Survey of resolution, polynomial calculus and cutting planes
- Resolution fairly well understood
- Polynomial calculus less so
- Cutting planes almost not at all
- Could there be interesting connections between proof complexity measures and hardness of SAT?
- How can we build efficient SAT solvers on stronger proof systems than resolution?

Thank you for your attention!

References I

- Carlos Ansótegui, María Luisa Bonet, Jordi Levy, and Felip Manyà. Measuring the hardness of SAT instances. In *Proceedings of the 23rd National Conference on Artificial Intelligence (AAAI '08)*, pages 222–228, 2008.

- 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*.

- Albert Atserias and Víctor Dalmau. A combinatorial characterization of resolution width. *Journal of Computer and System Sciences*, 74(3):323–334, 2008. Preliminary version in *CCC '03*.

- Albert Atserias, Johannes Klaus Fichte, and Marc Thurley. Clause-learning algorithms with many restarts and bounded-width resolution. *Journal of Artificial Intelligence Research*, 40:353–373, 2011. Preliminary version in *SAT '09*.

- Albert Atserias, Massimo Lauria, and Jakob Nordström. Narrow proofs may be maximally long. Submitted, 2013.

References II

- 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. Preliminary version in *FOCS '01*.
- Michael Alekhnovich and Alexander A. Razborov. Resolution is not automatizable unless W[P] is tractable. *SIAM Journal on Computing*, 38(4):1347–1363, 2008. Preliminary version in *FOCS '01*.
- Paul Beame, Chris Beck, and Russell Impagliazzo. Time-space tradeoffs in resolution: Superpolynomial lower bounds for superlinear space. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC '12)*, pages 213–232, 2012.
- Eli Ben-Sasson. Size-space tradeoffs for resolution. *SIAM Journal on Computing*, 38(6):2511–2525, 2009. Preliminary version in *STOC '02*.
- María Luisa Bonet and Nicola Galesi. Optimality of size-width tradeoffs for resolution. *Computational Complexity*, 10(4):261–276, 2001. Preliminary version in *FOCS '99*.

References III

Eli Ben-Sasson and Nicola Galesi. Space complexity of random formulae in resolution. *Random Structures and Algorithms*, 23(1):92–109, 2003. Preliminary version in *CCC '01*.

Ilario Bonacina and Nicola Galesi. Pseudo-partitions, transversality and locality: A combinatorial characterization for the space measure in algebraic proof systems. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science (ITCS '13)*, pages 455–472, 2013.

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, 2001. Preliminary version in *CCC '99*.

Samuel R. Buss, Jan Hoffmann, and Jan Johannsen. Resolution trees with lemmas: Resolution refinements that characterize DLL-algorithms with clause learning. *Logical Methods in Computer Science*, 4(4:13), 2008.

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

References IV

Paul Beame, Henry Kautz, and Ashish Sabharwal. Towards understanding and harnessing the potential of clause learning. *Journal of Artificial Intelligence Research*, 22:319–351, 2004.

Eli Ben-Sasson and Jakob Nordström. Short proofs may be spacious: An optimal separation of space and length in resolution. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS '08)*, pages 709–718, 2008.

Eli Ben-Sasson and Jakob Nordström. Understanding space in proof complexity: Separations and trade-offs via substitutions. In *Proceedings of the 2nd Symposium on Innovations in Computer Science (ICS '11)*, pages 401–416, 2011.

Chris Beck, Jakob Nordström, and Bangsheng Tang. Some trade-off results for polynomial calculus. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC '13)*, pages 813–822, 2013.

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

References V

- William Cook, Collette Rene Coullard, and Gyorgy Turán. On the complexity of cutting-plane proofs. *Discrete Applied Mathematics*, 18(1):25–38, 1987.
- 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, 1996.
- Vašek Chvátal and Endre Szemerédi. Many hard examples for resolution. *Journal of the ACM*, 35(4):759–768, 1988.
- Juan Luis Esteban and Jacobo Torán. Space bounds for resolution. *Information and Computation*, 171(1):84–97, 2001. Preliminary versions of these results in *STACS '99* and *CSL '99*.
- Yuval Filmus, Massimo Lauria, Mladen Mikša, Jakob Nordström, and Marc Vinyals. Towards an understanding of polynomial calculus: New separations and lower bounds. In *Proceedings of the 40th International Colloquium on Automata, Languages and Programming (ICALP '13)*, pages 437–448. 2013.

References VI

Yuval Filmus, Massimo Lauria, Jakob Nordström, Neil Thapen, and Noga Ron-Zewi. Space complexity in polynomial calculus. In *Proceedings of the 27th Annual IEEE Conference on Computational Complexity (CCC '12)*, pages 334–344, 2012.

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

Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. Technical Report 1311.2355, arXiv.org, 2013.

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

Philipp Hertel, Fahiem Bacchus, Toniann Pitassi, and Allen Van Gelder. Clause learning can effectively P-simulate general propositional resolution. In *Proceedings of the 23rd National Conference on Artificial Intelligence (AAAI '08)*, pages 283–290, 2008.

References VII

Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: Amplifying communication complexity hardness to time-space trade-offs in proof complexity. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC '12)*, pages 233–248, 2012.

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

Matti Järvisalo, Arie Matsliah, Jakob Nordström, and Stanislav Živný. Relating proof complexity measures and practical hardness of SAT. In *Proceedings of the 18th International Conference on Principles and Practice of Constraint Programming (CP '12)*, pages 316–331. 2012.

Knot Pipatsrisawat and Adnan Darwiche. On the power of clause-learning SAT solvers as resolution engines. *Artificial Intelligence*, 175:512–525, 2011. Preliminary version in *CP '09*.

Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *Journal of Symbolic Logic*, 62(3):981–998, 1997.

References VIII

- Alexander A. Razborov. Lower bounds for the polynomial calculus. *Computational Complexity*, 7(4):291–324, 1998.
- Søren Riis. *Independence in Bounded Arithmetic*. PhD thesis, University of Oxford, 1993.
- Gunnar Stålmarck. Short resolution proofs for a sequence of tricky formulas. *Acta Informatica*, 33(3):277–280, 1996.
- Alasdair Urquhart. Hard examples for resolution. *Journal of the ACM*, 34(1):209–219, 1987.