On Division Versus Saturation in Cutting Planes

Jakob Nordström

University of Copenhagen
and KTH Royal Institute of Technology

Simons Institute for the Theory of Computing
December 10, 2019

Joint work with Stephan Gocht and Amir Yehudayoff

SAT in Theory and Practice

Computational complexity

- Satisfiability fundamental problem in theoretical computer science
- SAT canonical NP-complete problem [Coo71, Lev73]
- Hence totally intractable in worst case (probably)
- One of the million dollar "Millennium Problems"

SAT in Theory and Practice

Computational complexity

- Satisfiability fundamental problem in theoretical computer science
- SAT canonical NP-complete problem [Coo71, Lev73]
- Hence totally intractable in worst case (probably)
- One of the million dollar "Millennium Problems"

SAT solving

- Enormous progress in performance last 15–20 years
- So-called conflict-driven clause learning (CDCL) solvers can deal with millions of variables
- Used for hardware & software verification, OR, AI, ...
- But also exist tiny formulas that are totally beyond reach

SAT in Theory and Practice

Computational complexity

- Satisfiability fundamental problem in theoretical computer science
- SAT canonical NP-complete problem [Coo71, Lev73]
- Hence totally intractable in worst case (probably)
- One of the million dollar "Millennium Problems"

SAT solving

- Enormous progress in performance last 15–20 years
- So-called conflict-driven clause learning (CDCL) solvers can deal with millions of variables
- Used for hardware & software verification, OR, AI, ...
- But also exist tiny formulas that are totally beyond reach

Limitations of CDCL

1. Clauses weak formalism for encoding constraints
2. Also weak method of reasoning (resolution)

Pseudo-Boolean Reasoning (a.k.a. 0-1 Linear Programming)

- Pseudo-Boolean (PB) linear constraints are stronger than clauses

Compare

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \geq 3$$

with

$$\begin{aligned}
&(x_1 \lor x_2 \lor x_3 \lor x_4) \land (x_1 \lor x_2 \lor x_3 \lor x_5) \land (x_1 \lor x_2 \lor x_3 \lor x_6) \\
\land &(x_1 \lor x_2 \lor x_4 \lor x_5) \land (x_1 \lor x_2 \lor x_4 \lor x_6) \land (x_1 \lor x_2 \lor x_5 \lor x_6) \\
\land &(x_1 \lor x_3 \lor x_4 \lor x_5) \land (x_1 \lor x_3 \lor x_4 \lor x_6) \land (x_1 \lor x_3 \lor x_5 \lor x_6) \\
\land &(x_1 \lor x_4 \lor x_5 \lor x_6) \land (x_2 \lor x_3 \lor x_4 \lor x_5) \land (x_2 \lor x_3 \lor x_4 \lor x_6) \\
\land &(x_2 \lor x_3 \lor x_5 \lor x_6) \land (x_2 \lor x_4 \lor x_5 \lor x_6) \land (x_3 \lor x_4 \lor x_5 \lor x_6)
\end{aligned}$$

Pseudo-Boolean Reasoning (a.k.a. 0-1 Linear Programming)

- Pseudo-Boolean (PB) linear constraints are stronger than clauses

Compare

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \geq 3$$

with

$$\begin{aligned}
&(x_1 \lor x_2 \lor x_3 \lor x_4) \land (x_1 \lor x_2 \lor x_3 \lor x_5) \land (x_1 \lor x_2 \lor x_3 \lor x_6) \\
\land&(x_1 \lor x_2 \lor x_4 \lor x_5) \land (x_1 \lor x_2 \lor x_4 \lor x_6) \land (x_1 \lor x_2 \lor x_5 \lor x_6) \\
\land&(x_1 \lor x_3 \lor x_4 \lor x_5) \land (x_1 \lor x_3 \lor x_4 \lor x_6) \land (x_1 \lor x_3 \lor x_5 \lor x_6) \\
\land&(x_1 \lor x_4 \lor x_5 \lor x_6) \land (x_2 \lor x_3 \lor x_4 \lor x_5) \land (x_2 \lor x_3 \lor x_4 \lor x_6) \\
\land&(x_2 \lor x_3 \lor x_5 \lor x_6) \land (x_2 \lor x_4 \lor x_5 \lor x_6) \land (x_3 \lor x_4 \lor x_5 \lor x_6)
\end{aligned}$$

- And pseudo-Boolean reasoning exponentially more powerful in theory

Pseudo-Boolean Reasoning (a.k.a. 0-1 Linear Programming)

- Pseudo-Boolean (PB) linear constraints are stronger than clauses

Compare

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \geq 3$$

with

$$\begin{aligned}
&(x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (x_1 \vee x_2 \vee x_3 \vee x_5) \wedge (x_1 \vee x_2 \vee x_3 \vee x_6) \\
\wedge &(x_1 \vee x_2 \vee x_4 \vee x_5) \wedge (x_1 \vee x_2 \vee x_4 \vee x_6) \wedge (x_1 \vee x_2 \vee x_5 \vee x_6) \\
\wedge &(x_1 \vee x_3 \vee x_4 \vee x_5) \wedge (x_1 \vee x_3 \vee x_4 \vee x_6) \wedge (x_1 \vee x_3 \vee x_5 \vee x_6) \\
\wedge &(x_1 \vee x_4 \vee x_5 \vee x_6) \wedge (x_2 \vee x_3 \vee x_4 \vee x_5) \wedge (x_2 \vee x_3 \vee x_4 \vee x_6) \\
\wedge &(x_2 \vee x_3 \vee x_5 \vee x_6) \wedge (x_2 \vee x_4 \vee x_5 \vee x_6) \wedge (x_3 \vee x_4 \vee x_5 \vee x_6)
\end{aligned}$$

- And pseudo-Boolean reasoning exponentially more powerful in theory

- But PB solvers less efficient than CDCL in practice!?

Our Work

- Study pseudo-Boolean rules of reasoning used in practice
- How do they compare to cutting planes proof system?
- In particular, what is the power of division versus saturation?

Our Work

- Study pseudo-Boolean rules of reasoning used in practice
- How do they compare to cutting planes proof system?
- In particular, what is the power of division versus saturation?

Broader message

- For many (most?) computational problems worst-case Turing machine model not terribly relevant
- But there are lots of interesting algorithms in need of rigorous analysis
- Can help out more applied colleagues, and at the same time do complexity theory for bounded computational models (what's not to like?)

Pseudo-Boolean Constraints and Normalized Form

In this talk, "pseudo-Boolean" (PB) refers to 0-1 integer linear constraints

Pseudo-Boolean Constraints and Normalized Form

In this talk, "pseudo-Boolean" (PB) refers to 0-1 integer linear constraints

Convenient to use non-negative linear combinations of literals, a.k.a. normalized form

$$\sum_i a_i \ell_i \geq A$$

- coefficients a_i: non-negative integers
- degree (of falsity) A: positive integer
- literals ℓ_i: x_i or $\overline{x}_i$ (where $x_i + \overline{x}_i = 1$)

(In what follows, all constraints assumed to be implicitly normalized)

Some Types of Pseudo-Boolean Constraints

1. Clauses are pseudo-Boolean constraints

$$x \vee \overline{y} \vee z \quad \Leftrightarrow \quad x + \overline{y} + z \geq 1$$

Refer to collection of such constraints as "CNF formula"

Some Types of Pseudo-Boolean Constraints

1. Clauses are pseudo-Boolean constraints

$$x \vee \overline{y} \vee z \quad \Leftrightarrow \quad x + \overline{y} + z \geq 1$$

   Refer to collection of such constraints as "CNF formula"

2. Cardinality constraints

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \geq 3$$

Some Types of Pseudo-Boolean Constraints

1. Clauses are pseudo-Boolean constraints

$$x \vee \overline{y} \vee z \quad \Leftrightarrow \quad x + \overline{y} + z \geq 1$$

   Refer to collection of such constraints as "CNF formula"

2. Cardinality constraints

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \geq 3$$

3. General constraints

$$x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$$

Approaches to Pseudo-Boolean Solving

Conversion to disjunctive clauses

- Lazy approach: learn clauses from PB constraints
  - *Sat4j* [LP10] (one of versions in library)
- Eager approach: re-encode to clauses and run CDCL
  - *MiniSat+* [ES06]
  - *Open-WBO* [MML14]
  - *NaPS* [SN15]

Approaches to Pseudo-Boolean Solving

Conversion to disjunctive clauses

- Lazy approach: learn clauses from PB constraints
  - *Sat4j* [LP10] (one of versions in library)
- Eager approach: re-encode to clauses and run CDCL
  - *MiniSat+* [ES06]
  - *Open-WBO* [MML14]
  - *NaPS* [SN15]

Native reasoning with pseudo-Boolean constraints

- *PRS* [DG02]
- *Galena* [CK05]
- *Pueblo* [SS06]
- *Sat4j* [LP10]
- *RoundingSat* [EN18]

Approaches to Pseudo-Boolean Solving

Conversion to disjunctive clauses

- Lazy approach: learn clauses from PB constraints
  - *Sat4j* [LP10] (one of versions in library)
- Eager approach: re-encode to clauses and run CDCL
  - *MiniSat+* [ES06]
  - *Open-WBO* [MML14]
  - *NaPS* [SN15]

Native reasoning with pseudo-Boolean constraints

- *PRS* [DG02]
- *Galena* [CK05]
- *Pueblo* [SS06]
- *Sat4j* [LP10]
- *RoundingSat* [EN18]

Conflict-Driven Search in a Pseudo-Boolean Setting

“Forward phase” — variable assignments

1. Always propagate forced assignment if possible
2. Otherwise make assignment using decision heuristic

Conflict-Driven Search in a Pseudo-Boolean Setting

“Forward phase” — variable assignments

1. Always propagate forced assignment if possible
2. Otherwise make assignment using decision heuristic

“Backward phase” — conflict analysis

1. When constraint violated (= conflict), derive new constraint that explains what went wrong
2. Add new constraint to instance ⇒ avoid same mistake again
3. Backtrack until no constraint violated

Propagation, Conflict, and Slack

Slack measures how far current assignment ρ is from falsifying $\sum_i a_i \ell_i \geq A$

Propagation, Conflict, and Slack

Slack measures how far current assignment ρ is from falsifying $\sum_i a_i\ell_i \geq A$

$$slack(\textstyle\sum_i a_i\ell_i \geq A; \rho) = \displaystyle\sum_{\ell_i \text{ not falsified by } \rho} a_i - A$$

Propagation, Conflict, and Slack

Slack measures how far current assignment ρ is from falsifying $\sum_i a_i \ell_i \geq A$

$$slack(\textstyle\sum_i a_i \ell_i \geq A; \rho) = \displaystyle\sum_{\ell_i \text{ not falsified by } \rho} a_i - A$$

Consider $C : x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$

ρ	$slack(C;\rho)$	comment

Propagation, Conflict, and Slack

Slack measures how far current assignment ρ is from falsifying $\sum_i a_i \ell_i \geq A$

$$slack(\sum_i a_i \ell_i \geq A; \rho) = \sum_{\ell_i \text{ not falsified by } \rho} a_i - A$$

Consider $C : x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$

ρ	$slack(C; \rho)$	comment
$\{\}$	8	

Propagation, Conflict, and Slack

Slack measures how far current assignment ρ is from falsifying $\sum_i a_i \ell_i \geq A$

$$slack(\textstyle\sum_i a_i \ell_i \geq A; \rho) = \displaystyle\sum_{\ell_i \text{ not falsified by } \rho} a_i - A$$

Consider $C : x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$

ρ	$slack(C;\rho)$	comment
$\{\}$	8	
$\{\overline{x}_5\}$	3	propagates $\overline{x}_4$ (coefficient $>$ slack)

Propagation, Conflict, and Slack

Slack measures how far current assignment ρ is from falsifying $\sum_i a_i\ell_i \geq A$

$$slack(\textstyle\sum_i a_i\ell_i \geq A; \rho) = \displaystyle\sum_{\ell_i \text{ not falsified by } \rho} a_i - A$$

Consider $C : x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$

ρ	$slack(C;\rho)$	comment
$\{\}$	8	
$\{\overline{x}_5\}$	3	propagates $\overline{x}_4$ (coefficient $>$ slack)
$\{\overline{x}_5, \overline{x}_4\}$	3	propagation doesn't change slack

Propagation, Conflict, and Slack

Slack measures how far current assignment ρ is from falsifying $\sum_i a_i \ell_i \geq A$

$$slack(\textstyle\sum_i a_i \ell_i \geq A; \rho) = \displaystyle\sum_{\ell_i \text{ not falsified by } \rho} a_i - A$$

Consider $C : x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$

ρ	$slack(C;\rho)$	comment
$\{\}$	8	
$\{\overline{x}_5\}$	3	propagates $\overline{x}_4$ (coefficient $>$ slack)
$\{\overline{x}_5, \overline{x}_4\}$	3	propagation doesn't change slack
$\{\overline{x}_5, \overline{x}_4, \overline{x}_3, x_2\}$	-2	conflict (slack < 0)

Propagation, Conflict, and Slack

Slack measures how far current assignment ρ is from falsifying $\sum_i a_i \ell_i \geq A$

$$slack(\textstyle\sum_i a_i \ell_i \geq A; \rho) = \displaystyle\sum_{\ell_i \text{ not falsified by } \rho} a_i - A$$

Consider $C : x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$

ρ	$slack(C;\rho)$	comment
$\{\}$	8	
$\{\overline{x}_5\}$	3	propagates $\overline{x}_4$ (coefficient $>$ slack)
$\{\overline{x}_5, \overline{x}_4\}$	3	propagation doesn't change slack
$\{\overline{x}_5, \overline{x}_4, \overline{x}_3, x_2\}$	-2	conflict (slack < 0)

At conflict, derive new constraint from conflict and propagating constraints

The Cutting Planes Proof System [CCT87]

Literal axioms $$\frac{}{\ell_i \geq 0}$$

Linear combination $$\frac{\sum_i a_i \ell_i \geq A \qquad \sum_i b_i \ell_i \geq B}{\sum_i (c_A a_i + c_B b_i)\ell_i \geq c_A A + c_B B} \quad [c_A, c_B \geq 0]$$

Division $$\frac{\sum_i a_i \ell_i \geq A}{\sum_i \lceil a_i / c \rceil \ell_i \geq \lceil A/c \rceil} \quad [c > 0]$$

The Cutting Planes Proof System [CCT87]

Literal axioms
$$\frac{}{\ell_i \geq 0}$$

Linear combination
$$\frac{\sum_i a_i \ell_i \geq A \qquad \sum_i b_i \ell_i \geq B}{\sum_i (c_A a_i + c_B b_i) \ell_i \geq c_A A + c_B B} \quad [c_A, c_B \geq 0]$$

Division
$$\frac{\sum_i a_i \ell_i \geq A}{\sum_i \lceil a_i / c \rceil \ell_i \geq \lceil A/c \rceil} \quad [c > 0]$$

Setting in this talk

Input: Set of pseudo-Boolean constraints without 0-1 solution

Goal: Prove unsatisfiability by deriving $0 \geq 1$ using cutting planes

The Cutting Planes Proof System [CCT87]

Literal axioms $$\frac{}{\ell_i \geq 0}$$

Linear combination $$\frac{\sum_i a_i \ell_i \geq A \qquad \sum_i b_i \ell_i \geq B}{\sum_i (c_A a_i + c_B b_i) \ell_i \geq c_A A + c_B B} \quad [c_A, c_B \geq 0]$$

Division $$\frac{\sum_i a_i \ell_i \geq A}{\sum_i \lceil a_i / c \rceil \ell_i \geq \lceil A/c \rceil} \quad [c > 0]$$

Setting in this talk

Input: Set of pseudo-Boolean constraints without 0-1 solution

Goal: Prove unsatisfiability by deriving $0 \geq 1$ using cutting planes

Ignore algorithmic aspects — heuristics beyond rigorous analysis — and assume optimal use of derivation rules

More About Cutting Planes

A toy example:

$$\frac{6x + 2y + 3z \geq 5 \qquad\qquad x + 2y + w \geq 1}{(6x + 2y + 3z) + 2(x + 2y + w) \geq 5 + 2 \cdot 1} \text{ Linear combination}$$

More About Cutting Planes

A toy example:

$$\frac{6x + 2y + 3z \geq 5 \qquad x + 2y + w \geq 1}{8x + 6y + 3z + 2w \geq 7} \text{ Linear combination}$$

More About Cutting Planes

A toy example:

$$\dfrac{\dfrac{6x + 2y + 3z \geq 5 \qquad x + 2y + w \geq 1}{8x + 6y + 3z + 2w \geq 7}\ \text{Linear combination}}{3x + 2y + z + w \geq 3}\ \text{Division}$$

More About Cutting Planes

A toy example:

$$\dfrac{\dfrac{6x + 2y + 3z \geq 5 \qquad x + 2y + w \geq 1}{8x + 6y + 3z + 2w \geq 7}\ \text{Linear combination}}{3x + 2y + z + w \geq 3}\ \text{Division}$$

- Literal axioms and linear combinations sound also over the reals
- Division is where the power of cutting planes lies
- Exponentially stronger than resolution/CDCL [Hak85, CCT87]

Generalized Resolution

In conflict-driven search, linear combination always made to cancel variable (on which constraints disagree)

Generalized resolution rule [Hoo88, Hoo92]

$$\frac{a_j x_j + \sum_{i \neq j} a_i \ell_i \geq A \qquad b_j \overline{x}_j + \sum_{i \neq j} b_i \ell_i \geq B}{\sum_{i \neq j} (\frac{c}{a_j} a_i + \frac{c}{b_j} b_i) \ell_i \geq \frac{c}{a_j} A + \frac{c}{b_j} B - c} \quad [c = \mathrm{lcm}(a_j, b_j)]$$

Generalized Resolution

In conflict-driven search, linear combination always made to cancel variable (on which constraints disagree)

Generalized resolution rule [Hoo88, Hoo92]

$$\frac{a_j x_j + \sum_{i \neq j} a_i \ell_i \geq A \qquad b_j \overline{x}_j + \sum_{i \neq j} b_i \ell_i \geq B}{\sum_{i \neq j} \left(\frac{c}{a_j} a_i + \frac{c}{b_j} b_i\right) \ell_i \geq \frac{c}{a_j} A + \frac{c}{b_j} B - c} \quad [c = \mathrm{lcm}(a_j, b_j)]$$

Another toy example:

$$\frac{2x + y + z \geq 2 \qquad\qquad 3\overline{x} + 2y + u + w \geq 3}{3(y+z) + 2(2y + u + w) \geq 3 \cdot 2 + 2 \cdot 3 - 6(x + \overline{x})} \quad \text{General resolution on } x$$

Generalized Resolution

In conflict-driven search, linear combination always made to cancel variable (on which constraints disagree)

Generalized resolution rule [Hoo88, Hoo92]

$$\frac{a_j x_j + \sum_{i \neq j} a_i \ell_i \geq A \qquad b_j \overline{x}_j + \sum_{i \neq j} b_i \ell_i \geq B}{\sum_{i \neq j} \left(\frac{c}{a_j} a_i + \frac{c}{b_j} b_i\right) \ell_i \geq \frac{c}{a_j} A + \frac{c}{b_j} B - c} \quad [c = \operatorname{lcm}(a_j, b_j)]$$

Another toy example:

$$\frac{2x + y + z \geq 2 \qquad 3\overline{x} + 2y + u + w \geq 3}{(3y + 3z) + (4y + 2u + 2w) \geq 12 - 6} \quad \text{General resolution on } x$$

Generalized Resolution

In conflict-driven search, linear combination always made to cancel variable (on which constraints disagree)

Generalized resolution rule [Hoo88, Hoo92]

$$\frac{a_j x_j + \sum_{i \neq j} a_i \ell_i \geq A \qquad b_j \overline{x}_j + \sum_{i \neq j} b_i \ell_i \geq B}{\sum_{i \neq j} \left(\frac{c}{a_j} a_i + \frac{c}{b_j} b_i\right) \ell_i \geq \frac{c}{a_j} A + \frac{c}{b_j} B - c} \quad [c = \mathrm{lcm}(a_j, b_j)]$$

Another toy example:

$$\frac{2x + y + z \geq 2 \qquad 3\overline{x} + 2y + u + w \geq 3}{7y + 3z + 2u + 2w \geq 6} \text{ General resolution on } x$$

Saturation

What's more, pseudo-Boolean solvers based on [CK05] do **not** do division

Instead use that no variable coefficient need be larger than maximum contribution required from that variable

Saturation rule

$$\frac{\sum_i a_i \ell_i \geq A}{\sum_i \min\{a_i, A\} \cdot \ell_i \geq A}$$

Saturation

What's more, pseudo-Boolean solvers based on [CK05] do **not** do division

Instead use that no variable coefficient need be larger than maximum contribution required from that variable

Saturation rule

$$\frac{\sum_i a_i \ell_i \geq A}{\sum_i \min\{a_i, A\} \cdot \ell_i \geq A}$$

Continuing our example:

$$\frac{7y + 3z + 2u + 2w \geq 6}{6y + 3z + 2u + 2w \geq 6}$$

Theoretical Understanding of Applied PB Reasoning?

Flavours of cutting planes in practice:

1. Boolean rule: (a) saturation or (b) division
2. Linear combinations: (a) generalized resolution or (b) no restrictions

Theoretical Understanding of Applied PB Reasoning?

Flavours of cutting planes in practice:

1. Boolean rule: (a) saturation or (b) division
2. Linear combinations: (a) generalized resolution or (b) no restrictions

Using generalized resolution seems inherent in conflict-driven setting
But what about Boolean rule?

- Saturation most popular (in [CK05, LP10], et cetera)
- Division used only recently in [EN18]
- What are the relative strengths of these rules?

Theoretical Understanding of Applied PB Reasoning?

Flavours of cutting planes in practice:

1. Boolean rule: (a) saturation or (b) division
2. Linear combinations: (a) generalized resolution or (b) no restrictions

Using generalized resolution seems inherent in conflict-driven setting
But what about Boolean rule?

- Saturation most popular (in [CK05, LP10], et cetera)
- Division used only recently in [EN18]
- What are the relative strengths of these rules? Nothing known...

Theoretical Understanding of Applied PB Reasoning?

Flavours of cutting planes in practice:

1. Boolean rule: (a) saturation or (b) division
2. Linear combinations: (a) generalized resolution or (b) no restrictions

Using generalized resolution seems inherent in conflict-driven setting
But what about Boolean rule?

- Saturation most popular (in [CK05, LP10], et cetera)
- Division used only recently in [EN18]
- What are the relative strengths of these rules? Nothing known...

Striking contrast to long line of work on resolution and CDCL ([BKS04, HBPV08, BHJ08, AFT11, PD11] ...)

Our Results

1. For CNF, saturation no stronger than resolution proof system / CDCL (even for unrestricted linear combinations)

Our Results

1. For CNF, saturation no stronger than resolution proof system / CDCL (even for unrestricted linear combinations)
2. Division + generalized resolution can be exponentially stronger than saturation + unrestricted linear combinations

Our Results

1. For CNF, saturation no stronger than resolution proof system / CDCL (even for unrestricted linear combinations)
2. Division + generalized resolution can be exponentially stronger than saturation + unrestricted linear combinations
3. Single saturation step can require unbounded # divisions to simulate, even with unrestricted linear combinations

Our Results

1. For CNF, saturation no stronger than resolution proof system / CDCL (even for unrestricted linear combinations)
2. Division + generalized resolution can be exponentially stronger than saturation + unrestricted linear combinations
3. Single saturation step can require unbounded # divisions to simulate, even with unrestricted linear combinations

1st result strengthens [VEG$^+$18]
Focus on 2nd and 3rd results — first of its kind

Cutting Planes and Implicational Completeness

- All flavours of cutting planes except division + unrestricted linear combinations as in [CCT87] collapse to resolution for CNFs
- Full cutting planes implicationally complete — can recover, e.g., cardinality constraints from CNF

Cutting Planes and Implicational Completeness

- All flavours of cutting planes except division + unrestricted linear combinations as in [CCT87] collapse to resolution for CNFs
- Full cutting planes implicationally complete — can recover, e.g., cardinality constraints from CNF

$$\dfrac{\dfrac{\begin{array}{r} x + y \phantom{{}+z} \geq 1 \\ x + \phantom{y+{}} z \geq 1 \\ y + z \geq 1 \end{array}}{2x + 2y + 2z \geq 3}\;\text{[2 non-cancelling additions]}}{x + y + z \geq 2}\;\text{[Divide by } 2\text{]}$$

Cutting Planes and Implicational Completeness

- All flavours of cutting planes except division + unrestricted linear combinations as in [CCT87] collapse to resolution for CNFs
- Full cutting planes implicationally complete — can recover, e.g., cardinality constraints from CNF

$$\begin{array}{rl}
x + y \phantom{{}+z} \geq 1 & \\
x + \phantom{y+{}} z \geq 1 & \\
y + z \geq 1 & \\
\hline
2x + 2y + 2z \geq 3 & \text{[2 non-cancelling additions]} \\
\hline
x + y + z \geq 2 & \text{[Divide by } 2\text{]}
\end{array}$$

- Impossible with generalized resolution!
- So pigeonhole principle (PHP) in CNF hard for PB solvers
- CNFs make life hard for both saturation and division — but we want to show that division can be stronger!

Cutting Planes and Implicational Completeness

- All flavours of cutting planes except division + unrestricted linear combinations as in [CCT87] collapse to resolution for CNFs
- Full cutting planes implicationally complete — can recover, e.g., cardinality constraints from CNF

$$\begin{array}{rl}
x + y \phantom{{}+z} & \geq 1 \\
x + \phantom{y+{}} z & \geq 1 \\
y + z & \geq 1 \\
\hline
2x + 2y + 2z & \geq 3 \quad \text{[2 non-cancelling additions]} \\
\hline
x + y + z & \geq 2 \quad \text{[Divide by } 2\text{]}
\end{array}$$

- Impossible with generalized resolution!
- So pigeonhole principle (PHP) in CNF hard for PB solvers
- CNFs make life hard for both saturation and division — but we want to show that division can be stronger! *Can do so by cheating...*

Division + Resolution Can Be Stronger Than Saturation

Take formula requiring recovery of cardinality constraints from CNF

Division + Resolution Can Be Stronger Than Saturation

Take formula requiring recovery of cardinality constraints from CNF

$$
\begin{array}{rr}
x + y \phantom{{}+z} \geq 1 & \\
x + \phantom{y+{}} z \geq 1 & \\
y + z \geq 1 & \\
\hline
2x + 2y + 2z \geq 3 & \text{[2 non-cancelling additions]} \\
\hline
x + y + z \geq 2 & \text{[Divide by } 2\text{]}
\end{array}
$$

Division + Resolution Can Be Stronger Than Saturation

Take formula requiring recovery of cardinality constraints from CNF

$$\begin{array}{r}
h_1 + h_2 + x + y \geq 1 \\
\overline{h}_1 + x + z \geq 2 \\
\overline{h}_2 + y + z \geq 2 \\
\hline
2x + 2y + 2z \geq 3 \quad \text{[2 generalized resolution steps]} \\
\hline
x + y + z \geq 2 \quad \text{[Divide by } 2\text{]}
\end{array}$$

Add helper variables to make all linear combinations cancelling
$\Rightarrow$ Now easy for division + resolution, since easy for full cutting planes

Division + Resolution Can Be Stronger Than Saturation

Take formula requiring recovery of cardinality constraints from CNF

$$
\begin{array}{rl}
h_1 + h_2 + x + y & \geq 1 \\
\overline{h}_1 + x + z & \geq 2 \\
\overline{h}_2 + y + z & \geq 2 \\
\hline
2x + 2y + 2z & \geq 3 \quad \text{[2 generalized resolution steps]} \\
\hline
x + y + z & \geq 2 \quad \text{[Divide by } 2\text{]}
\end{array}
$$

Add helper variables to make all linear combinations cancelling
$\Rightarrow$ Now easy for division + resolution, since easy for full cutting planes

Assigning helper variables $= 0$ gives back CNF encoding
$\Rightarrow$ Cutting planes proofs preserved under partial assignments
$\Rightarrow$ Still hard for saturation, even with unrestricted linear combinations

"Cheating" Applied To Subset Cardinality Formulas

Variables = 1s in matrix with four 1s per row/column + extra 1
Row ⇒ majority of variables true; column ⇒ majority false

$$\begin{pmatrix}
\mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\
0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\
0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 \\
0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} \\
\mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 \\
0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 \\
0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 \\
0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} \\
\mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 \\
0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} \\
\mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & 0 & \mathbf{1}
\end{pmatrix}$$

$$\begin{aligned}
& (x_{1,1} \lor x_{1,2} \lor x_{1,4}) \\
\land\ & (x_{1,1} \lor x_{1,2} \lor x_{1,8}) \\
\land\ & (x_{1,1} \lor x_{1,4} \lor x_{1,8}) \\
\land\ & (x_{1,2} \lor x_{1,4} \lor x_{1,8}) \\
& \vdots \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{8,11} \lor \overline{x}_{10,11}) \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{8,11} \lor \overline{x}_{11,11}) \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{10,11} \lor \overline{x}_{11,11}) \\
\land\ & (\overline{x}_{8,11} \lor \overline{x}_{10,11} \lor \overline{x}_{11,11})
\end{aligned}$$

"Cheating" Applied To Subset Cardinality Formulas

Variables = 1s in matrix with four 1s per row/column **+ extra 1**
Row ⇒ majority of variables true; column ⇒ majority false

$$\begin{pmatrix}
\mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\
0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\
0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 \\
0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} \\
\mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 \\
0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 \\
0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 \\
0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} \\
\mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 \\
0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} \\
\mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & 0 & \mathbf{1}
\end{pmatrix}$$

$$\begin{aligned}
& (x_{1,1} \vee x_{1,2} \vee x_{1,4}) \\
\wedge\ & (x_{1,1} \vee x_{1,2} \vee x_{1,8}) \\
\wedge\ & (x_{1,1} \vee x_{1,4} \vee x_{1,8}) \\
\wedge\ & (x_{1,2} \vee x_{1,4} \vee x_{1,8}) \\
& \vdots \\
\wedge\ & (\overline{x}_{4,11} \vee \overline{x}_{8,11} \vee \overline{x}_{10,11}) \\
\wedge\ & (\overline{x}_{4,11} \vee \overline{x}_{8,11} \vee \overline{x}_{11,11}) \\
\wedge\ & (\overline{x}_{4,11} \vee \overline{x}_{10,11} \vee \overline{x}_{11,11}) \\
\wedge\ & (\overline{x}_{8,11} \vee \overline{x}_{10,11} \vee \overline{x}_{11,11})
\end{aligned}$$

"Cheating" Applied To Subset Cardinality Formulas

Variables = 1s in matrix with four 1s per row/column **+ extra 1**
Row ⇒ majority of variables true; column ⇒ majority false

$$\begin{pmatrix}
\mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\
0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\
0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 \\
0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} \\
\mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 \\
0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 \\
0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 \\
0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} \\
\mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 \\
0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} \\
\mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & 0 & \mathbf{1}
\end{pmatrix}$$

$$\begin{aligned}
& (x_{1,1} \lor x_{1,2} \lor x_{1,4}) \\
\land\ & (x_{1,1} \lor x_{1,2} \lor x_{1,8}) \\
\land\ & (x_{1,1} \lor x_{1,4} \lor x_{1,8}) \\
\land\ & (x_{1,2} \lor x_{1,4} \lor x_{1,8}) \\
& \vdots \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{8,11} \lor \overline{x}_{10,11}) \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{8,11} \lor \overline{x}_{11,11}) \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{10,11} \lor \overline{x}_{11,11}) \\
\land\ & (\overline{x}_{8,11} \lor \overline{x}_{10,11} \lor \overline{x}_{11,11})
\end{aligned}$$

“Cheating” Applied To Subset Cardinality Formulas

Variables = 1s in matrix with four 1s per row/column **+ extra 1**

Row ⇒ majority of variables true; column ⇒ majority false

$$
\begin{pmatrix}
\mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\
0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\
0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 \\
0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \color{red}{\mathbf{1}} \\
\mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 \\
0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 \\
0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 \\
0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \color{red}{\mathbf{1}} \\
\mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 \\
0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \color{red}{\mathbf{1}} \\
\mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & 0 & \color{red}{\mathbf{1}}
\end{pmatrix}
$$

$$
\begin{aligned}
& (x_{1,1} \lor x_{1,2} \lor x_{1,4}) \\
\land\ & (x_{1,1} \lor x_{1,2} \lor x_{1,8}) \\
\land\ & (x_{1,1} \lor x_{1,4} \lor x_{1,8}) \\
\land\ & (x_{1,2} \lor x_{1,4} \lor x_{1,8}) \\
& \vdots \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{8,11} \lor \overline{x}_{10,11}) \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{8,11} \lor \overline{x}_{11,11}) \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{10,11} \lor \overline{x}_{11,11}) \\
\land\ & (\overline{x}_{8,11} \lor \overline{x}_{10,11} \lor \overline{x}_{11,11})
\end{aligned}
$$

"Cheating" Applied To Subset Cardinality Formulas

Variables = 1s in matrix with four 1s per row/column **+ extra 1**

Row ⇒ majority of variables true; column ⇒ majority false

$$\begin{pmatrix}
\mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\
0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\
0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 \\
0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} \\
\mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 \\
0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 \\
0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 \\
0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} \\
\mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 \\
0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} \\
\mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & 0 & \mathbf{1}
\end{pmatrix}$$

$$\begin{aligned}
& (x_{1,1} \lor x_{1,2} \lor x_{1,4}) \\
\land\ & (x_{1,1} \lor x_{1,2} \lor x_{1,8}) \\
\land\ & (x_{1,1} \lor x_{1,4} \lor x_{1,8}) \\
\land\ & (x_{1,2} \lor x_{1,4} \lor x_{1,8}) \\
& \vdots \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{8,11} \lor \overline{x}_{10,11}) \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{8,11} \lor \overline{x}_{11,11}) \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{10,11} \lor \overline{x}_{11,11}) \\
\land\ & (\overline{x}_{8,11} \lor \overline{x}_{10,11} \lor \overline{x}_{11,11})
\end{aligned}$$

Exponentially hard for resolution for expanding matrices [MN14]

"Cheating" Applied To Subset Cardinality Formulas

Variables = 1s in matrix with four 1s per row/column **+ extra 1**
Row ⇒ majority of variables true; column ⇒ majority false

$$\begin{pmatrix}
\mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\
0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\
0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 \\
0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} \\
\mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 \\
0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 & 0 \\
0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} & 0 \\
0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & \mathbf{1} \\
\mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 \\
0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} \\
\mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} & 0 & 0 & \mathbf{1}
\end{pmatrix}$$

$$\begin{aligned}
& (x_{1,1} \lor x_{1,2} \lor x_{1,4}) \\
\land\ & (x_{1,1} \lor x_{1,2} \lor x_{1,8}) \\
\land\ & (x_{1,1} \lor x_{1,4} \lor x_{1,8}) \\
\land\ & (x_{1,2} \lor x_{1,4} \lor x_{1,8}) \\
& \vdots \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{8,11} \lor \overline{x}_{10,11}) \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{8,11} \lor \overline{x}_{11,11}) \\
\land\ & (\overline{x}_{4,11} \lor \overline{x}_{10,11} \lor \overline{x}_{11,11}) \\
\land\ & (\overline{x}_{8,11} \lor \overline{x}_{10,11} \lor \overline{x}_{11,11})
\end{aligned}$$

Exponentially hard for resolution for expanding matrices [MN14]
Easy for cutting planes: recover cardinality constraints and count

Simulating Saturation by Division

Division can simulate saturation by completeness — but how efficiently?

$$200x + 51y + 50z + 49w \geq 100$$

Simulating Saturation by Division

Division can simulate saturation by completeness — but how efficiently?

$$200x + \quad 51y + \quad 50z + \quad 49w \geq \quad 100$$

Simulating Saturation by Division

Division can simulate saturation by completeness — but how efficiently?

$$\frac{200x + 51y + 50z + 49w \geq 100}{20000x + 5100y + 5000z + 4900w \geq 10000}\ \text{Multiplication by } 100$$

Simulating Saturation by Division

Division can simulate saturation by completeness — but how efficiently?

$$\begin{array}{rrrrl} 200x + & 51y + & 50z + & 49w \geq & 100 \\ \hline 20000x + & 5100y + & 5000z + & 4900w \geq & 10000 \\ \hline 199x + & 51y + & 50z + & 49w \geq & 100 \end{array}$$

Multiplication by 100

Division by 101

Simulating Saturation by Division

Division can simulate saturation by completeness — but how efficiently?

$$\dfrac{\dfrac{\dfrac{200x + 51y + 50z + 49w \geq 100}{20000x + 5100y + 5000z + 4900w \geq 10000}\text{ Multiplication by }100}{199x + 51y + 50z + 49w \geq 100}\text{ Division by }101}{19900x + 5100y + 5000z + 4900w \geq 10000}\text{ Multiplication by }100$$

Simulating Saturation by Division

Division can simulate saturation by completeness — but how efficiently?

$$\dfrac{\dfrac{\dfrac{\dfrac{200x + 51y + 50z + 49w \geq 100}{20000x + 5100y + 5000z + 4900w \geq 10000}\ \text{Multiplication by } 100}{199x + 51y + 50z + 49w \geq 100}\ \text{Division by } 101}{19900x + 5100y + 5000z + 4900w \geq 10000}\ \text{Multiplication by } 100}{198x + 51y + 50z + 49w \geq 100}\ \text{Division by } 101$$

Simulating Saturation by Division

Division can simulate saturation by completeness — but how efficiently?

$$\dfrac{200x + 51y + 50z + 49w \geq 100}{20000x + 5100y + 5000z + 4900w \geq 10000}\ \text{Multiplication by } 100$$

$$\dfrac{20000x + 5100y + 5000z + 4900w \geq 10000}{199x + 51y + 50z + 49w \geq 100}\ \text{Division by } 101$$

$$\dfrac{199x + 51y + 50z + 49w \geq 100}{19900x + 5100y + 5000z + 4900w \geq 10000}\ \text{Multiplication by } 100$$

$$\dfrac{19900x + 5100y + 5000z + 4900w \geq 10000}{198x + 51y + 50z + 49w \geq 100}\ \text{Division by } 101$$

$$\dfrac{198x + 51y + 50z + 49w \geq 100}{19800x + 5100y + 5000z + 4900w \geq 10000}\ \text{Multiplication by } 100$$

Simulating Saturation by Division

Division can simulate saturation by completeness — but how efficiently?

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{200x + 51y + 50z + 49w \geq 100}{20000x + 5100y + 5000z + 4900w \geq 10000}\text{ Multiplication by } 100}{199x + 51y + 50z + 49w \geq 100}\text{ Division by } 101}{19900x + 5100y + 5000z + 4900w \geq 10000}\text{ Multiplication by } 100}{198x + 51y + 50z + 49w \geq 100}\text{ Division by } 101}{19800x + 5100y + 5000z + 4900w \geq 10000}\text{ Multiplication by } 100$$

$$\dfrac{19800x + 5100y + 5000z + 4900w \geq 10000}{197x + 51y + 50z + 49w \geq 100}\text{ Division by } 101$$

$$\vdots$$

Simulating Saturation by Division

Division can simulate saturation by completeness — but how efficiently?

$$
\begin{array}{l}
200x + 51y + 50z + 49w \geq 100 \\
\hline
20000x + 5100y + 5000z + 4900w \geq 10000 \\
\hline
199x + 51y + 50z + 49w \geq 100 \\
\hline
19900x + 5100y + 5000z + 4900w \geq 10000 \\
\hline
198x + 51y + 50z + 49w \geq 100 \\
\hline
19800x + 5100y + 5000z + 4900w \geq 10000 \\
\hline
197x + 51y + 50z + 49w \geq 100 \\
\hline
\vdots
\end{array}
\quad
\begin{array}{l}
\text{Multiplication by } 100 \\
\text{Division by } 101 \\
\text{Multiplication by } 100 \\
\text{Division by } 101 \\
\text{Multiplication by } 100 \\
\text{Division by } 101
\end{array}
$$

Exponentially many steps measured in bitsize of coefficents...

Simulating Saturation by Division

Division can simulate saturation by completeness — but how efficiently?

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{200x + 51y + 50z + 49w \geq 100}{20000x + 5100y + 5000z + 4900w \geq 10000}\ \text{Multiplication by } 100}{199x + 51y + 50z + 49w \geq 100}\ \text{Division by } 101}{19900x + 5100y + 5000z + 4900w \geq 10000}\ \text{Multiplication by } 100}{198x + 51y + 50z + 49w \geq 100}\ \text{Division by } 101}{19800x + 5100y + 5000z + 4900w \geq 10000}\ \text{Multiplication by } 100$$

$$\dfrac{19800x + 5100y + 5000z + 4900w \geq 10000}{197x + 51y + 50z + 49w \geq 100}\ \text{Division by } 101$$

$$\vdots$$

Exponentially many steps measured in bitsize of coefficents...

Our result: Impossible to get rid of exponential dependence in general!

Division Can't Simulate Saturation Efficiently

Consider derivation

$$\dfrac{\dfrac{Rx + Ry + \sum_{i=1}^{R} z_i \geq R \qquad Rx + R\overline{y} + \sum_{i=R+1}^{2R} z_i \geq R}{2Rx + \sum_{i=1}^{2R} z_i \geq R}}{Rx + \sum_{i=1}^{2R} z_i \geq R}$$

Division Can't Simulate Saturation Efficiently

Consider derivation

$$\dfrac{\dfrac{\color{red}{Rx} + Ry + \sum_{i=1}^{R} z_i \geq R \qquad \color{red}{Rx} + R\overline{y} + \sum_{i=R+1}^{2R} z_i \geq R}{\color{red}{2Rx} + \sum_{i=1}^{2R} z_i \geq R}}{\color{red}{Rx} + \sum_{i=1}^{2R} z_i \geq R}$$

Theorem

Deriving

- $Rx + \sum_{i=1}^{2R} z_i \geq R$

from

- $Rx + Ry + \sum_{i=1}^{R} z_i \geq R$ *and*
- $Rx + R\overline{y} + \sum_{i=R+1}^{2R} z_i \geq R$

requires $\Omega(\sqrt{R})$ *division steps (even with unrestricted linear combinations)*

Proof Sketch

Define potential function

$$\mathcal{P}\left(ax + by + b'\overline{y} + \sum c_i z_i \geq A\right) = \ln\left((2a + b + b')/A\right)$$

Proof Sketch

Define potential function

$$\mathcal{P}\left(ax + by + b'\overline{y} + \sum c_i z_i \geq A\right) = \ln\left((2a + b + b')/A\right)$$

At start: $\mathcal{P}\left(Rx + Ry + \sum_{i=1}^{R} z_i \geq R\right) = \ln(3)$

At end: $\mathcal{P}\left(Rx + \sum_{i=1}^{2R} z_i \geq R\right) = \ln(2)$

Proof Sketch

Define potential function

$$\mathcal{P}\left(ax + by + b'\overline{y} + \sum c_i z_i \geq A\right) = \ln\left((2a + b + b')/A\right)$$

At start: $\mathcal{P}\left(Rx + Ry + \sum_{i=1}^{R} z_i \geq R\right) = \ln(3)$

At end: $\mathcal{P}\left(Rx + \sum_{i=1}^{2R} z_i \geq R\right) = \ln(2)$

Properties:

1. Potential needs to drop by $\geq 1/6$

Proof Sketch

Define potential function

$$\mathcal{P}\left(ax + by + b'\overline{y} + \sum c_i z_i \geq A\right) = \ln\left((2a + b + b')/A\right)$$

At start: $\mathcal{P}\left(Rx + Ry + \sum_{i=1}^{R} z_i \geq R\right) = \ln(3)$

At end: $\mathcal{P}\left(Rx + \sum_{i=1}^{2R} z_i \geq R\right) = \ln(2)$

Properties:

1. Potential needs to drop by $\geq 1/6$
2. But linear combination C_1 & $C_2 \rightarrow C'$ doesn't decrease potential: $\mathcal{P}(C') \geq \min\{\mathcal{P}(C_1), \mathcal{P}(C_2)\}$

Proof Sketch

Define potential function

$$\mathcal{P}\left(ax + by + b'\overline{y} + \sum c_i z_i \geq A\right) = \ln\left((2a + b + b')/A\right)$$

At start: $\mathcal{P}\left(Rx + Ry + \sum_{i=1}^{R} z_i \geq R\right) = \ln(3)$

At end: $\mathcal{P}\left(Rx + \sum_{i=1}^{2R} z_i \geq R\right) = \ln(2)$

Properties:

1. Potential needs to drop by $\geq 1/6$
2. But linear combination C_1 & $C_2 \to C'$ doesn't decrease potential: $\mathcal{P}(C') \geq \min\{\mathcal{P}(C_1), \mathcal{P}(C_2)\}$
3. And division $C \to C'$ only decreases potential by small amount: $\mathcal{P}(C') \geq \mathcal{P}(C) - 1/\sqrt{R}$

Proof Sketch

Define potential function

$$\mathcal{P}\left(ax + by + b'\overline{y} + \sum c_i z_i \geq A\right) = \ln\left((2a + b + b')/A\right)$$

At start: $\mathcal{P}\left(Rx + Ry + \sum_{i=1}^{R} z_i \geq R\right) = \ln(3)$

At end: $\mathcal{P}\left(Rx + \sum_{i=1}^{2R} z_i \geq R\right) = \ln(2)$

Properties:

1. Potential needs to drop by $\geq 1/6$
2. But linear combination C_1 & $C_2 \to C'$ doesn't decrease potential: $\mathcal{P}(C') \geq \min\{\mathcal{P}(C_1), \mathcal{P}(C_2)\}$
3. And division $C \to C'$ only decreases potential by small amount: $\mathcal{P}(C') \geq \mathcal{P}(C) - 1/\sqrt{R}$

Hence $\Omega(\sqrt{R})$ division steps needed

Saturation + Resolution Can Be Stronger Than Division?

- Does this show that saturation + generalized resolution can be exponentially stronger than division? **No!**

Saturation + Resolution Can Be Stronger Than Division?

- Does this show that saturation + generalized resolution can be exponentially stronger than division? **No!**
- Only shows that saturation step can't be simulated efficiently

Saturation + Resolution Can Be Stronger Than Division?

- Does this show that saturation + generalized resolution can be exponentially stronger than division? **No!**
- Only shows that saturation step can't be simulated efficiently
- Doesn't rule out that cutting planes with division could prove unsatisfiability of benchmarks in completely different way

Saturation + Resolution Can Be Stronger Than Division?

- Does this show that saturation + generalized resolution can be exponentially stronger than division? **No!**
- Only shows that saturation step can't be simulated efficiently
- Doesn't rule out that cutting planes with division could prove unsatisfiability of benchmarks in completely different way
- But if division is always as good as saturation, then it seems like proof of this can't be simple step-by-step simulation

Some Experimental Results

Strength of Division

- When division better than saturation, *RoundingSat* [EN18] can run much faster than *Sat4j* [LP10]
- But very sensitive to how helper variables encoded

Some Experimental Results

Strength of Division

- When division better than saturation, *RoundingSat* [EN18] can run much faster than *Sat4j* [LP10]
- But very sensitive to how helper variables encoded

Strength of Saturation

- Have easy benchmarks for saturation that look tricky for division — in theory
- In practice, the benchmarks we tried so far are hard for both division- and saturation-based solvers

Some Experimental Results

Strength of Division

- When division better than saturation, *RoundingSat* [EN18] can run much faster than *Sat4j* [LP10]
- But very sensitive to how helper variables encoded

Strength of Saturation

- Have easy benchmarks for saturation that look tricky for division — in theory
- In practice, the benchmarks we tried so far are hard for both division- and saturation-based solvers

Caveat: obviously artificial benchmarks — we just want to see if separations can happen in actual solvers

Directions for Future Research

Division versus saturation

- Can cutting planes with saturation be more powerful than division at proving unsatisfiability?
- Can we find good algorithms combining division and saturation?
- Can potential functions be a more general approach for proving lower bounds?

Directions for Future Research

Division versus saturation

- Can cutting planes with saturation be more powerful than division at proving unsatisfiability?
- Can we find good algorithms combining division and saturation?
- Can potential functions be a more general approach for proving lower bounds?

Fundamental challenges

- All PB solvers degenerate to resolution for CNF inputs
- Sometimes very poor performance even when LP relaxation infeasible! Combine with mixed integer linear programming (MIP) techniques?
- Ongoing work [DGN19, EN20]...

Take-Home Message

- Porting conflict-driven paradigm to pseudo-Boolean solving (0-1 ILP) has potential for exponential gains

Take-Home Message

- Porting conflict-driven paradigm to pseudo-Boolean solving (0-1 ILP) has potential for exponential gains
- Mostly haven't materialized — instead nontrivial challenges regarding
  - Efficient implementation
  - Theoretical understanding

Take-Home Message

- Porting conflict-driven paradigm to pseudo-Boolean solving (0-1 ILP) has potential for exponential gains
- Mostly haven't materialized — instead nontrivial challenges regarding
    - Efficient implementation
    - Theoretical understanding
- This work: Show that division and saturation rules are incomparable

Take-Home Message

- Porting conflict-driven paradigm to pseudo-Boolean solving (0-1 ILP) has potential for exponential gains
- Mostly haven't materialized — instead nontrivial challenges regarding
  - Efficient implementation
  - Theoretical understanding
- This work: Show that division and saturation rules are incomparable
- Future research: Better algorithms and lower bounds needed

Take-Home Message

- Porting conflict-driven paradigm to pseudo-Boolean solving (0-1 ILP) has potential for exponential gains
- Mostly haven't materialized — instead nontrivial challenges regarding
  - Efficient implementation
  - Theoretical understanding
- This work: Show that division and saturation rules are incomparable
- Future research: Better algorithms and lower bounds needed
- Along the way lots of fun questions to work on! ☺

Take-Home Message

- Porting conflict-driven paradigm to pseudo-Boolean solving (0-1 ILP) has potential for exponential gains
- Mostly haven't materialized — instead nontrivial challenges regarding
    - Efficient implementation
    - Theoretical understanding
- This work: Show that division and saturation rules are incomparable
- Future research: Better algorithms and lower bounds needed
- Along the way lots of fun questions to work on! ☺

Advertisement: University of Copenhagen is hiring!

- PhD students, postdocs, and faculty
- World-leading in algorithms — expanding into complexity theory

Take-Home Message

- Porting conflict-driven paradigm to pseudo-Boolean solving (0-1 ILP) has potential for exponential gains
- Mostly haven't materialized — instead nontrivial challenges regarding
  - Efficient implementation
  - Theoretical understanding
- This work: Show that division and saturation rules are incomparable
- Future research: Better algorithms and lower bounds needed
- Along the way lots of fun questions to work on! ☺

Advertisement: University of Copenhagen is hiring!

- PhD students, postdocs, and faculty
- World-leading in algorithms — expanding into complexity theory

Thank you for your attention!

References I

[AFT11] 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, January 2011. Preliminary version in *SAT '09*.

[BHJ08] 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), December 2008.

[BKS04] Paul Beame, Henry Kautz, and Ashish Sabharwal. Towards understanding and harnessing the potential of clause learning. *Journal of Artificial Intelligence Research*, 22:319–351, December 2004. Preliminary version in *IJCAI '03*.

[CCT87] William Cook, Collette Rene Coullard, and György Turán. On the complexity of cutting-plane proofs. *Discrete Applied Mathematics*, 18(1):25–38, November 1987.

[CK05] Donald Chai and Andreas Kuehlmann. A fast pseudo-Boolean constraint solver. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24(3):305–317, March 2005. Preliminary version in *DAC '03*.

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

References II

[DG02] Heidi E. Dixon and Matthew L. Ginsberg. Inference methods for a pseudo-Boolean satisfiability solver. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI '02)*, pages 635–640, July 2002.

[DGN19] Jo Devriendt, Ambros Gleixner, and Jakob Nordström. Learn to relax: Integrating integer linear programming with conflict-driven search. Submitted manuscript, 2019.

[EN18] Jan Elffers and Jakob Nordström. Divide and conquer: Towards faster pseudo-Boolean solving. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI '18)*, pages 1291–1299, July 2018.

[EN20] Jan Elffers and Jakob Nordström. A cardinal improvement to pseudo-Boolean solving. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI '20)*, February 2020. To appear.

[ES06] Niklas Eén and Niklas Sörensson. Translating pseudo-Boolean constraints into SAT. *Journal on Satisfiability, Boolean Modeling and Computation*, 2(1-4):1–26, March 2006.

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

References III

[HBPV08] 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, July 2008.

[Hoo88] John N. Hooker. Generalized resolution and cutting planes. *Annals of Operations Research*, 12(1):217–239, December 1988.

[Hoo92] John N. Hooker. Generalized resolution for 0-1 linear inequalities. *Annals of Mathematics and Artificial Intelligence*, 6(1):271–286, March 1992.

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

[LP10] Daniel Le Berre and Anne Parrain. The Sat4j library, release 2.2. *Journal on Satisfiability, Boolean Modeling and Computation*, 7:59–64, July 2010.

[MML14] Ruben Martins, Vasco M. Manquinho, and Inês Lynce. Open-WBO: A modular MaxSAT solver. In *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT '14)*, volume 8561 of *Lecture Notes in Computer Science*, pages 438–445. Springer, July 2014.

References IV

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

[PD11] Knot Pipatsrisawat and Adnan Darwiche. On the power of clause-learning SAT solvers as resolution engines. *Artificial Intelligence*, 175(2):512–525, February 2011. Preliminary version in *CP '09*.

[SN15] Masahiko Sakai and Hidetomo Nabeshima. Construction of an ROBDD for a PB-constraint in band form and related techniques for PB-solvers. *IEICE Transactions on Information and Systems*, 98-D(6):1121–1127, June 2015.

[SS06] Hossein M. Sheini and Karem A. Sakallah. Pueblo: A hybrid pseudo-Boolean SAT solver. *Journal on Satisfiability, Boolean Modeling and Computation*, 2(1-4):165–189, March 2006. Preliminary version in *DATE '05*.

[VEG+18] Marc Vinyals, Jan Elffers, Jesús Giráldez-Cru, Stephan Gocht, and Jakob Nordström. In between resolution and cutting planes: A study of proof systems for pseudo-Boolean SAT solving. In *Proceedings of the 21st International Conference on Theory and Applications of Satisfiability Testing (SAT '18)*, volume 10929 of *Lecture Notes in Computer Science*, pages 292–310. Springer, July 2018.