Tutorial on Conflict-Driven Pseudo-Boolean Solving

Jakob Nordström

University of Copenhagen and Lund University

SAT + SMT Winter School
Chennai, India
December 15, 2022

Pseudo-Boolean?

Pseudo-Boolean (PB) function: $f : \{0,1\}^n \to \mathbb{R}$

Studied since 1960s in operations research and 0-1 integer linear programming [BH02]

Such a function f can always be represented as polynomial

Restriction for these lectures: f represented as linear form

Many problems expressible as optimizing value of linear pseudo-Boolean function under linear pseudo-Boolean constraints

Pseudo-Boolean vs. SAT

- PB format richer than conjunctive normal form (CNF)

Compare

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

and

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

Pseudo-Boolean vs. SAT

- PB format richer than conjunctive normal form (CNF)

Compare

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

and

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

- And pseudo-Boolean reasoning exponentially stronger than conflict-driven clause learning (CDCL)

Pseudo-Boolean vs. SAT

- PB format richer than conjunctive normal form (CNF)

Compare

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

and

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

- And pseudo-Boolean reasoning exponentially stronger than conflict-driven clause learning (CDCL)
- Yet close enough to SAT to benefit from SAT solving advances

Pseudo-Boolean vs. SAT

- PB format richer than conjunctive normal form (CNF)

Compare

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

and

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

- And pseudo-Boolean reasoning exponentially stronger than conflict-driven clause learning (CDCL)
- Yet close enough to SAT to benefit from SAT solving advances
- Also possible synergies with 0-1 integer linear programming (ILP)

Outline of Tutorial on Pseudo-Boolean Solving

Pseudo-Boolean Constraints and Normalized Form

For us, pseudo-Boolean constraints are always 0-1 integer linear constraints

$$\sum_i a_i \ell_i \bowtie A$$

- $\bowtie \in \{\geq, \leq, =, >, <\}$
- $a_i, A \in \mathbb{Z}$
- literals ℓ_i: x_i or $\overline{x}_i$ (where $x_i + \overline{x}_i = 1$)
- variables x_i take values $0 = \mathit{false}$ or $1 = \mathit{true}$

Pseudo-Boolean Constraints and Normalized Form

For us, pseudo-Boolean constraints are always 0-1 integer linear constraints

$$\sum_i a_i \ell_i \bowtie A$$

- $\bowtie \in \{\geq, \leq, =, >, <\}$
- $a_i, A \in \mathbb{Z}$
- literals ℓ_i: x_i or $\overline{x}_i$ (where $x_i + \overline{x}_i = 1$)
- variables x_i take values $0 = \mathit{false}$ or $1 = \mathit{true}$

Convenient to use normalized form [Bar95] (without loss of generality)

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

- constraint always greater-than-or-equal
- $a_i, A \in \mathbb{N}$
- $A = \mathit{deg}(\sum_i a_i \ell_i \geq A)$ referred to as degree (of falsity)

Some Types of Pseudo-Boolean Constraints

1. Clauses are pseudo-Boolean constraints

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

Some Types of Pseudo-Boolean Constraints

1. Clauses are pseudo-Boolean constraints

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

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

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

2. Cardinality constraints

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

3. General constraints

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

Conversion to Normalized Form: Example

Normalized form used for convenience and without loss of generality

$$-x_1 + 2x_2 - 3x_3 + 4x_4 - 5x_5 < 0$$

Conversion to Normalized Form: Example

Normalized form used for convenience and without loss of generality

$$-x_1 + 2x_2 - 3x_3 + 4x_4 - 5x_5 < 0$$

1. Make inequality non-strict

$$-x_1 + 2x_2 - 3x_3 + 4x_4 - 5x_5 \leq -1$$

Conversion to Normalized Form: Example

Normalized form used for convenience and without loss of generality

$$-x_1 + 2x_2 - 3x_3 + 4x_4 - 5x_5 < 0$$

1. Make inequality non-strict

$$-x_1 + 2x_2 - 3x_3 + 4x_4 - 5x_5 \leq -1$$

2. Multiply by -1 to get greater-than-or-equal

$$x_1 - 2x_2 + 3x_3 - 4x_4 + 5x_5 \geq 1$$

Conversion to Normalized Form: Example

Normalized form used for convenience and without loss of generality

$$-x_1 + 2x_2 - 3x_3 + 4x_4 - 5x_5 < 0$$

1. Make inequality non-strict

$$-x_1 + 2x_2 - 3x_3 + 4x_4 - 5x_5 \leq -1$$

2. Multiply by -1 to get greater-than-or-equal

$$x_1 - 2x_2 + 3x_3 - 4x_4 + 5x_5 \geq 1$$

3. Replace $-\ell$ by $-(1 - \overline{\ell})$ [where we define $\overline{\overline{x}} \doteq x$]

$$\begin{aligned} x_1 - 2(1 - \overline{x}_2) + 3x_3 - 4(1 - \overline{x}_4) + 5x_5 &\geq 1 \\ x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 &\geq 7 \end{aligned}$$

Conversion to Normalized Form: Example

Normalized form used for convenience and without loss of generality

$$-x_1 + 2x_2 - 3x_3 + 4x_4 - 5x_5 < 0$$

1. Make inequality non-strict

$$-x_1 + 2x_2 - 3x_3 + 4x_4 - 5x_5 \leq -1$$

2. Multiply by -1 to get greater-than-or-equal

$$x_1 - 2x_2 + 3x_3 - 4x_4 + 5x_5 \geq 1$$

3. Replace $-\ell$ by $-(1-\overline{\ell})$ [where we define $\overline{\overline{x}} \doteq x$]

$$x_1 - 2(1-\overline{x}_2) + 3x_3 - 4(1-\overline{x}_4) + 5x_5 \geq 1$$
$$x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$$

4. Replace "$=$" by two inequalities "$\geq$" and "$\leq$"

Formulas, Decision Problems, and Optimization Problems

Pseudo-Boolean (PB) formula

Conjunction of pseudo-Boolean constraints

$F \doteq C_1 \land C_2 \land \cdots \land C_m$

Formulas, Decision Problems, and Optimization Problems

Pseudo-Boolean (PB) formula

Conjunction of pseudo-Boolean constraints

$F \doteq C_1 \land C_2 \land \cdots \land C_m$

Pseudo-Boolean Solving (PBS)

Decide whether F is satisfiable/feasible

Formulas, Decision Problems, and Optimization Problems

Pseudo-Boolean (PB) formula
Conjunction of pseudo-Boolean constraints
$F \doteq C_1 \wedge C_2 \wedge \cdots \wedge C_m$

Pseudo-Boolean Solving (PBS)
Decide whether F is satisfiable/feasible

Pseudo-Boolean Optimization (PBO)
Find satisfying assignment to F minimizing objective function $\sum_i w_i \ell_i$
(Maximization: minimize $-\sum_i w_i \ell_i$)

Formulas, Decision Problems, and Optimization Problems

Pseudo-Boolean (PB) formula

Conjunction of pseudo-Boolean constraints

$F \doteq C_1 \wedge C_2 \wedge \cdots \wedge C_m$

Pseudo-Boolean Solving (PBS)

Decide whether F is satisfiable/feasible

Pseudo-Boolean Optimization (PBO)

Find satisfying assignment to F minimizing objective function $\sum_i w_i \ell_i$

(Maximization: minimize $-\sum_i w_i \ell_i$)

This lecture:

- Focus on pseudo-Boolean solving
- But not hard to extend to (simple) optimization algorithm

Some Problems Expressed as PBO (1/2)

Input:

- undirected graph $G = (V, E)$
- weight function $w : V \to \mathbb{N}^{+}$

Some Problems Expressed as PBO (1/2)

Input:

- undirected graph $G = (V, E)$
- weight function $w : V \to \mathbb{N}^+$

Weighted maximum clique

$$\min \; -\sum_{v \in V} w(v) \cdot x_v$$

$$\overline{x}_u + \overline{x}_v \geq 1 \qquad (u, v) \notin E$$

Some Problems Expressed as PBO (1/2)

Input:

- undirected graph $G = (V, E)$
- weight function $w : V \to \mathbb{N}^+$

Weighted maximum clique

$$\min \; -\textstyle\sum_{v \in V} w(v) \cdot x_v$$

$$\overline{x}_u + \overline{x}_v \geq 1 \qquad (u, v) \notin E$$

Weighted minimum vertex cover

$$\min \; \textstyle\sum_{v \in V} w(v) \cdot x_v$$

$$x_u + x_v \geq 1 \qquad (u, v) \in E$$

Some Problems Expressed as PBO (2/2)

Input:

- sets $S_1, \ldots, S_m \subseteq \mathcal{U}$
- weight function $w : \mathcal{U} \to \mathbb{N}^+$

Some Problems Expressed as PBO (2/2)

Input:

- sets $S_1, \ldots, S_m \subseteq \mathcal{U}$
- weight function $w : \mathcal{U} \to \mathbb{N}^+$

Weighted minimum hitting set

Find $H \subseteq \mathcal{U}$ such that

- $H \cap S_i \neq \emptyset$ for all $i \in [m]$ (H is a hitting set)
- $\sum_{h \in H} w(h)$ is minimal

Some Problems Expressed as PBO (2/2)

Input:

- sets $S_1, \ldots, S_m \subseteq \mathcal{U}$
- weight function $w : \mathcal{U} \to \mathbb{N}^+$

Weighted minimum hitting set

Find $H \subseteq \mathcal{U}$ such that

- $H \cap S_i \neq \emptyset$ for all $i \in [m]$ (H is a hitting set)
- $\sum_{h \in H} w(h)$ is minimal

$$\min \sum_{e \in \mathcal{U}} w(e) \cdot x_e$$

$$\sum_{e \in S_i} x_e \geq 1 \qquad i \in [m]$$

Some Problems Expressed as PBO (2/2)

Input:

- sets $S_1, \ldots, S_m \subseteq \mathcal{U}$
- weight function $w : \mathcal{U} \to \mathbb{N}^+$

Weighted minimum hitting set

Find $H \subseteq \mathcal{U}$ such that

- $H \cap S_i \neq \emptyset$ for all $i \in [m]$ (H is a hitting set)
- $\sum_{h \in H} w(h)$ is minimal

$$\min \sum_{e \in \mathcal{U}} w(e) \cdot x_e$$
$$\sum_{e \in S_i} x_e \geq 1 \qquad i \in [m]$$

Note: In all of these examples, the problem is to

- optimize a linear function
- subject to a CNF formula (all constraints are clausal)

Already expressive framework!

Approaches for Pseudo-Boolean Problems

What we will discuss in the coming lectures:

1. Pseudo-Boolean (PB) solving and optimization
2. MaxSAT solving
3. Integer linear programming (ILP) — or, more generally, mixed integer linear programming (MIP)

Approaches for Pseudo-Boolean Problems

What we will discuss in the coming lectures:

1. Pseudo-Boolean (PB) solving and optimization
2. MaxSAT solving
3. Integer linear programming (ILP) — or, more generally, mixed integer linear programming (MIP)

Rough conceptual difference:

- **PB/SAT:** Focus on integral solutions, try to find optimal one
- **ILP/MIP:** Find optimal non-integer solution; search for integral solutions nearby

Basic trade-off: Inference power vs. inference speed

A Quick Recap of Modern SAT Solving

DPLL method [DP60, DLL62]

- Assign values to variables (in some smart way)
- Backtrack when conflict with falsified clause

A Quick Recap of Modern SAT Solving

DPLL method [DP60, DLL62]

- Assign values to variables (in some smart way)
- Backtrack when conflict with falsified clause

Conflict-driven clause learning (CDCL) [MS99, MMZ$^+$01]

- Analyse conflicts in more detail — add new clauses to formula
- More efficient backtracking
- Also let conflicts guide other heuristics

A Quick Recap of Modern SAT Solving

DPLL method [DP60, DLL62]

- Assign values to variables (in some smart way)
- Backtrack when conflict with falsified clause

Conflict-driven clause learning (CDCL) [MS99, MMZ+01]

- Analyse conflicts in more detail — add new clauses to formula
- More efficient backtracking
- Also let conflicts guide other heuristics

CDCL Main Loop Pseudocode

CDCL(F)

$\mathcal{D} \leftarrow F$; `// initialize clause database to contain formula`
$\rho \leftarrow \emptyset$; `// initialize assignment trail to empty`
forever do
    if *ρ falsifies some clause $C \in \mathcal{D}$* **then**
        $A \leftarrow$ analyzeConflict$(\mathcal{D}, \rho, C)$;
        if $A = \bot$ **then** output `UNSATISFIABLE` and exit;
        else
            add A to $\mathcal{D}$ and backjump by shrinking ρ ;
    else if *exists clause $C \in \mathcal{D}$ unit propagating x to $b \in \{0, 1\}$ under ρ* **then**
        add propagated assignment $x \overset{D}{=} b$ to ρ ;
    else if *time to restart* **then** $\rho \leftarrow \emptyset$;
    else if *time for clause database reduction* **then**
        erase (roughly) half of learned clauses in $\mathcal{D} \setminus F$ from $\mathcal{D}$
    else if *all variables assigned* **then** output `SATISFIABLE` and exit;
    else
        use decision scheme to choose assignment $x \overset{\mathsf{d}}{=} b$ to add to ρ ;

CDCL Main Loop Pseudocode

CDCL(F)

$\mathcal{D} \leftarrow F$; `// initialize clause database to contain formula`
$\rho \leftarrow \emptyset$; `// initialize assignment trail to empty`
forever do
    if *ρ falsifies some clause $C \in \mathcal{D}$* **then**
        $A \leftarrow$ analyzeConflict$(\mathcal{D}, \rho, C)$;
        if $A = \perp$ **then** output `UNSATISFIABLE` and exit;
        else
            add A to $\mathcal{D}$ and backjump by shrinking ρ ;
    else if *exists clause $C \in \mathcal{D}$ unit propagating x to $b \in \{0,1\}$ under ρ* **then**
        add propagated assignment $x \stackrel{D}{=} b$ to ρ ;
    else if *time to restart* **then** $\rho \leftarrow \emptyset$;
    else if *time for clause database reduction* **then**
        erase (roughly) half of learned clauses in $\mathcal{D} \setminus F$ from $\mathcal{D}$
    else if *all variables assigned* **then** output `SATISFIABLE` and exit;
    else
        use decision scheme to choose assignment $x \stackrel{\mathrm{d}}{=} b$ to add to ρ ;

Conflict Analysis Pseudocode

analyzeConflict($\mathcal{D}, \rho, C_{\mathrm{confl}}$)

$C_{\mathrm{learn}} \leftarrow C_{\mathrm{confl}}$;
while C_{learn} *not UIP clause* **and** $C_{\mathrm{learn}} \neq \bot$ **do**
  $\ell \leftarrow$ literal assigned last on trail ρ;
  if ℓ *propagated* **and** $\overline{\ell}$ *occurs in* C_{learn} **then**
    $C_{\mathrm{reason}} \leftarrow \mathrm{reason}(\ell, \rho, \mathcal{D})$;
    $C_{\mathrm{learn}} \leftarrow \mathrm{resolve}(C_{\mathrm{learn}}, C_{\mathrm{reason}})$;
  $\rho \leftarrow \rho \setminus \{\ell\}$;
return C_{learn};

SAT-Based Approaches to Pseudo-Boolean Solving

Conversion to disjunctive clauses

- Lazy approach: learn clauses from PB constraints
  - SAT4J [LP10] (one of versions in library)

SAT-Based 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]

SAT-Based 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]

SAT-Based 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]

"Native" Pseudo-Boolean Conflict-Driven Search

Want to do "same thing" as in conflict-driven clause learning (CDCL) SAT solving but with pseudo-Boolean constraints without re-encoding

"Native" Pseudo-Boolean Conflict-Driven Search

Want to do "same thing" as in conflict-driven clause learning (CDCL) SAT solving but with pseudo-Boolean constraints without re-encoding

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

"Native" Pseudo-Boolean Conflict-Driven Search

Want to do "same thing" as in conflict-driven clause learning (CDCL) SAT solving but with pseudo-Boolean constraints without re-encoding

- Variable assignments
  1. Always propagate forced assignment if possible
  2. Otherwise make assignment using decision heuristic
- At conflict
  1. Do conflict analysis to derive new constraint
  2. Add new constraint to constraint database
  3. Backjump by rolling back decisions so that learned constraint propagates asserting literal (flipping it to opposite value)

Propagation, Conflict, and Slack

Let ρ current assignment of solver (a.k.a. trail)
Represent as $\rho = \{$(ordered) set of literals assigned true$\}$

Propagation, Conflict, and Slack

Let ρ current assignment of solver (a.k.a. trail)
Represent as $\rho = \{$(ordered) set of literals assigned true$\}$

Slack measures how far 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

Let ρ current assignment of solver (a.k.a. trail)
Represent as $\rho = \{$(ordered) set of literals assigned true$\}$

Slack measures how far 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 $C \doteq x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$

ρ	$slack(C; \rho)$	comment

Propagation, Conflict, and Slack

Let ρ current assignment of solver (a.k.a. trail)
Represent as $\rho = \{$(ordered) set of literals assigned true$\}$

Slack measures how far 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 $C \doteq x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$

ρ	$slack(C;\rho)$	comment
$\{\}$	8	

Propagation, Conflict, and Slack

Let ρ current assignment of solver (a.k.a. trail)
Represent as $\rho = \{$(ordered) set of literals assigned true$\}$

Slack measures how far 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 $C \doteq x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$

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

Propagation, Conflict, and Slack

Let ρ current assignment of solver (a.k.a. trail)
Represent as $\rho = \{$(ordered) set of literals assigned true$\}$

Slack measures how far 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 $C \doteq x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$

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

Let ρ current assignment of solver (a.k.a. trail)
Represent as $\rho = \{$(ordered) set of literals assigned true$\}$

Slack measures how far 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 $C \doteq x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$

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

Let ρ current assignment of solver (a.k.a. trail)
Represent as $\rho = \{$(ordered) set of literals assigned true$\}$

Slack measures how far 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 $C \doteq x_1 + 2\overline{x}_2 + 3x_3 + 4\overline{x}_4 + 5x_5 \geq 7$

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)

Note: constraint can be conflicting though not all variables assigned

Conflict Analysis Invariant

Consider example CDCL conflict analysis from SAT solving lecture

$$(p \lor \overline{u}) \land (q \lor r) \land (\overline{r} \lor w) \land (u \lor x \lor y) \land (x \lor \overline{y} \lor z) \land (\overline{x} \lor z) \land (\overline{y} \lor \overline{z}) \land (\overline{x} \lor \overline{z}) \land (\overline{p} \lor \overline{u})$$

$p \overset{\mathsf{d}}{=} 0$

$u \overset{p \lor \overline{u}}{=} 0$

$q \overset{\mathsf{d}}{=} 0$

$r \overset{q \lor r}{=} 1$

$w \overset{\overline{r} \lor w}{=} 1$

$x \overset{\mathsf{d}}{=} 0$

$y \overset{u \lor x \lor y}{=} 1$

$z \overset{x \lor \overline{y} \lor z}{=} 1$

$\overset{\overline{y} \lor \overline{z}}{\bot}$

Conflict Analysis Invariant

Consider example CDCL conflict analysis from SAT solving lecture

$$(p \lor \overline{u}) \land (q \lor r) \land (\overline{r} \lor w) \land (u \lor x \lor y) \land (x \lor \overline{y} \lor z) \land (\overline{x} \lor z) \land (\overline{y} \lor \overline{z}) \land (\overline{x} \lor \overline{z}) \land (\overline{p} \lor \overline{u})$$

$p \overset{\mathsf{d}}{=} 0$

$u \overset{p \lor \overline{u}}{=} 0$

$q \overset{\mathsf{d}}{=} 0$

$r \overset{q \lor r}{=} 1$

$w \overset{\overline{r} \lor w}{=} 1$

$x \overset{\mathsf{d}}{=} 0$

$y \overset{u \lor x \lor y}{=} 1$

$z \overset{x \lor \overline{y} \lor z}{=} 1$

$\overset{\overline{y} \lor \overline{z}}{\bot}$

Assignment "left on trail" always falsifies derived clause

Conflict Analysis Invariant

Consider example CDCL conflict analysis from SAT solving lecture

$$(p \lor \overline{u}) \land (q \lor r) \land (\overline{r} \lor w) \land (u \lor x \lor y) \land (x \lor \overline{y} \lor z) \land (\overline{x} \lor z) \land (\overline{y} \lor \overline{z}) \land (\overline{x} \lor \overline{z}) \land (\overline{p} \lor \overline{u})$$

$p \overset{\mathsf{d}}{=} 0$

$u \overset{p \lor \overline{u}}{=} 0$

$q \overset{\mathsf{d}}{=} 0$

$r \overset{q \lor r}{=} 1$

$w \overset{\overline{r} \lor w}{=} 1$

$x \overset{\mathsf{d}}{=} 0$

$y \overset{u \lor x \lor y}{=} 1$

$z \overset{x \lor \overline{y} \lor z}{=} 1$

$\overset{\overline{y} \lor \overline{z}}{\bot}$

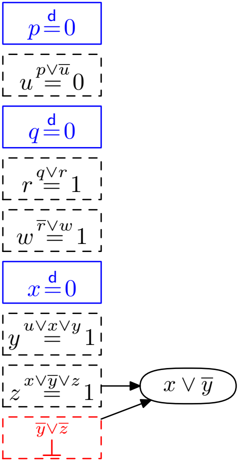
Assignment "left on trail" always falsifies derived clause

$\overline{y} \lor \overline{z}$ falsified by
trail $\rho = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}, y, z\}$

Conflict Analysis Invariant

Consider example CDCL conflict analysis from SAT solving lecture

$$(p \lor \overline{u}) \land (q \lor r) \land (\overline{r} \lor w) \land (u \lor x \lor y) \land (x \lor \overline{y} \lor z) \land (\overline{x} \lor z) \land (\overline{y} \lor \overline{z}) \land (\overline{x} \lor \overline{z}) \land (\overline{p} \lor \overline{u})$$



Assignment "left on trail" always falsifies derived clause

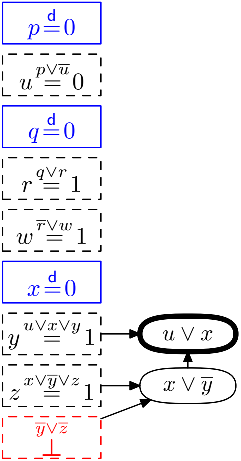
$x \lor \overline{y}$ falsified by trail $\rho' = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}, y\}$

$\overline{y} \lor \overline{z}$ falsified by trail $\rho = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}, y, z\}$

Conflict Analysis Invariant

Consider example CDCL conflict analysis from SAT solving lecture

$$(p \vee \overline{u}) \wedge (q \vee r) \wedge (\overline{r} \vee w) \wedge (u \vee x \vee y) \wedge (x \vee \overline{y} \vee z) \wedge (\overline{x} \vee z) \wedge (\overline{y} \vee \overline{z}) \wedge (\overline{x} \vee \overline{z}) \wedge (\overline{p} \vee \overline{u})$$



Assignment "left on trail" always falsifies derived clause

$u \vee x$ falsified by trail $\rho'' = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}\}$

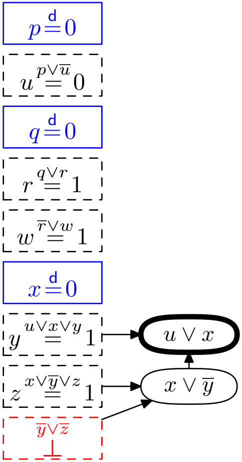
$x \vee \overline{y}$ falsified by trail $\rho' = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}, y\}$

$\overline{y} \vee \overline{z}$ falsified by trail $\rho = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}, y, z\}$

Conflict Analysis Invariant

Consider example CDCL conflict analysis from SAT solving lecture

$$(p \lor \overline{u}) \land (q \lor r) \land (\overline{r} \lor w) \land (u \lor x \lor y) \land (x \lor \overline{y} \lor z) \land (\overline{x} \lor z) \land (\overline{y} \lor \overline{z}) \land (\overline{x} \lor \overline{z}) \land (\overline{p} \lor \overline{u})$$



$u \lor x$ falsified by trail $\rho'' = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}\}$

$x \lor \overline{y}$ falsified by trail $\rho' = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}, y\}$

$\overline{y} \lor \overline{z}$ falsified by trail $\rho = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}, y, z\}$

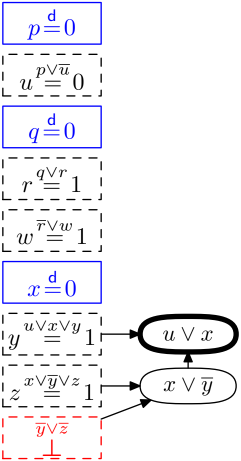
Assignment "left on trail" always falsifies derived clause

$\Rightarrow$ derived clause "explains" conflict

Conflict Analysis Invariant

Consider example CDCL conflict analysis from SAT solving lecture

$$(p \vee \overline{u}) \wedge (q \vee r) \wedge (\overline{r} \vee w) \wedge (u \vee x \vee y) \wedge (x \vee \overline{y} \vee z) \wedge (\overline{x} \vee z) \wedge (\overline{y} \vee \overline{z}) \wedge (\overline{x} \vee \overline{z}) \wedge (\overline{p} \vee \overline{u})$$



$u \vee x$ falsified by trail $\rho'' = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}\}$

$x \vee \overline{y}$ falsified by trail $\rho' = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}, y\}$

$\overline{y} \vee \overline{z}$ falsified by trail $\rho = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}, y, z\}$

Assignment "left on trail" always falsifies derived clause

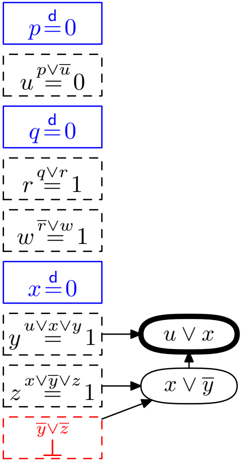
$\Rightarrow$ derived clause "explains" conflict

Terminate analysis when explanation "looks nice"

Conflict Analysis Invariant

Consider example CDCL conflict analysis from SAT solving lecture

$$(p \vee \overline{u}) \wedge (q \vee r) \wedge (\overline{r} \vee w) \wedge (u \vee x \vee y) \wedge (x \vee \overline{y} \vee z) \wedge (\overline{x} \vee z) \wedge (\overline{y} \vee \overline{z}) \wedge (\overline{x} \vee \overline{z}) \wedge (\overline{p} \vee \overline{u})$$



$u \vee x$ falsified by trail $\rho'' = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}\}$

$x \vee \overline{y}$ falsified by trail $\rho' = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}, y\}$

$\overline{y} \vee \overline{z}$ falsified by trail $\rho = \{\overline{p}, \overline{u}, \overline{q}, r, w, \overline{x}, y, z\}$

Assignment "left on trail" always falsifies derived clause

$\Rightarrow$ derived clause "explains" conflict

Terminate analysis when explanation "looks nice"

Namely: after backjump, some variable guaranteed to flip

Generalized Resolution

Can mimic resolution step

$$\frac{x \vee \overline{y} \vee z \qquad \overline{y} \vee \overline{z}}{x \vee \overline{y}}$$

Generalized Resolution

Can mimic resolution step

$$\frac{x \vee \overline{y} \vee z \qquad \overline{y} \vee \overline{z}}{x \vee \overline{y}}$$

by adding clauses as pseudo-Boolean constraints

$$\frac{x + \overline{y} + z \geq 1 \qquad \overline{y} + \overline{z} \geq 1}{x + 2\overline{y} \geq 1}$$

(Recall $z + \overline{z} = 1$)

Generalized Resolution

Can mimic resolution step

$$\frac{x \lor \overline{y} \lor z \qquad \overline{y} \lor \overline{z}}{x \lor \overline{y}}$$

by adding clauses as pseudo-Boolean constraints

$$\frac{x + \overline{y} + z \geq 1 \qquad \overline{y} + \overline{z} \geq 1}{x + 2\overline{y} \geq 1}$$

(Recall $z + \overline{z} = 1$)

Generalized resolution rule (from [Hoo88, Hoo92])
Positive linear combination so that some variable cancels

$$\frac{a_1 x_1 + \sum_{i \geq 2} a_i \ell_i \geq A \qquad b_1 \overline{x}_1 + \sum_{i \geq 2} b_i \ell_i \geq B}{\sum_{i \geq 2} \left(\frac{c}{a_1} a_i + \frac{c}{b_1} b_i\right) \ell_i \geq \frac{c}{a_1} A + \frac{c}{b_1} B - c} \; [c = \mathrm{lcm}(a_1, b_1)]$$

Saturation

Actually, not quite the right constraint in mimicking of resolution

$$\frac{x + \overline{y} + z \geq 1 \qquad \overline{y} + \overline{z} \geq 1}{x + 2\overline{y} \geq 1}$$

Saturation

Actually, not quite the right constraint in mimicking of resolution

$$\frac{x + \overline{y} + z \geq 1 \qquad \overline{y} + \overline{z} \geq 1}{x + 2\overline{y} \geq 1}$$

But clearly valid to conclude

$$\frac{x + 2\overline{y} \geq 1}{x + \overline{y} \geq 1}$$

Saturation

Actually, not quite the right constraint in mimicking of resolution

$$\frac{x + \overline{y} + z \geq 1 \qquad \overline{y} + \overline{z} \geq 1}{x + 2\overline{y} \geq 1}$$

But clearly valid to conclude

$$\frac{x + 2\overline{y} \geq 1}{x + \overline{y} \geq 1}$$

Saturation rule

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

Sound over integers, not over reals (need such rules for SAT solving)

Saturation

Actually, not quite the right constraint in mimicking of resolution

$$\frac{x + \overline{y} + z \geq 1 \qquad \overline{y} + \overline{z} \geq 1}{x + 2\overline{y} \geq 1}$$

But clearly valid to conclude

$$\frac{x + 2\overline{y} \geq 1}{x + \overline{y} \geq 1}$$

Saturation rule

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

Sound over integers, not over reals (need such rules for SAT solving)

[Generalized resolution as defined in [Hoo88, Hoo92] includes fix above, but convenient here to make the two separate steps explicit]

Analyze Conflict with Generalized Resolution + Saturation!

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Analyze Conflict with Generalized Resolution + Saturation!

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \stackrel{\mathsf{d}}{=} 0, x_2 \stackrel{C_1}{=} 1, x_3 \stackrel{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2
(Note: same constraint can propagate several times!)

Analyze Conflict with Generalized Resolution + Saturation!

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \overset{\mathsf{d}}{=} 0, x_2 \overset{C_1}{=} 1, x_3 \overset{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2
(Note: same constraint can propagate several times!)

- Resolve reason$(x_3, \rho) = C_1$ with C_2 over x_3 to get resolve(C_1, C_2, x_3)

$$\frac{2x_1 + 2x_2 + 2x_3 + x_4 \geq 4 \qquad 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3}{x_4 \geq 1}$$

Analyze Conflict with Generalized Resolution + Saturation!

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \stackrel{\mathsf{d}}{=} 0, x_2 \stackrel{C_1}{=} 1, x_3 \stackrel{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2
(Note: same constraint can propagate several times!)

- Resolve reason$(x_3, \rho) = C_1$ with C_2 over x_3 to get resolve(C_1, C_2, x_3)

$$\frac{2x_1 + 2x_2 + 2x_3 + x_4 \geq 4 \qquad 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3}{x_4 \geq 1}$$

- Applying saturate$(x_4 \geq 1)$ does nothing

Analyze Conflict with Generalized Resolution + Saturation!

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \stackrel{\mathsf{d}}{=} 0, x_2 \stackrel{C_1}{=} 1, x_3 \stackrel{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2
(Note: same constraint can propagate several times!)

- Resolve reason$(x_3, \rho) = C_1$ with C_2 over x_3 to get resolve(C_1, C_2, x_3)

$$\frac{2x_1 + 2x_2 + 2x_3 + x_4 \geq 4 \qquad 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3}{x_4 \geq 1}$$

- Applying saturate$(x_4 \geq 1)$ does nothing
- Non-negative slack w.r.t. $\rho' = \{x_1 \stackrel{\mathsf{d}}{=} 0, x_2 \stackrel{C_1}{=} 1\}$
  Not conflicting! Does not explain mistake in assignment

What Went Wrong? And What to Do About It?

Accident report

- Generalized resolution sound over the reals
- Given $\rho' = \{x_1 = 0, x_2 = 1\}$, over the reals have
  - $C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$ propagates $x_3 \geq \frac{1}{2}$
  - $C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$ satisfied by $x_3 \leq \frac{1}{2}$
- So after resolving away x_3 no conflict left!

What Went Wrong? And What to Do About It?

Accident report

- Generalized resolution sound over the reals
- Given $\rho' = \{x_1 = 0, x_2 = 1\}$, over the reals have
  - $C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$ propagates $x_3 \geq \frac{1}{2}$
  - $C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$ satisfied by $x_3 \leq \frac{1}{2}$
- So after resolving away x_3 no conflict left!

Remedial action

- Strengthen propagation to $x_3 \geq 1$ also over the reals
- I.e., want reason C with $slack(C; \rho') = 0$
- **Fix** (non-obvious): Apply weakening

$$\text{weaken}\left(\textstyle\sum_i a_i\ell_i \geq A, \ell_j\right) \doteq \textstyle\sum_{i \neq j} a_i\ell_i \geq A - a_j$$

  to reason constraint and then saturate
- Approach in [CK05] (goes back to observations in [Wil76])

Try to Reduce the Reason Constraint

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \stackrel{\mathsf{d}}{=} 0, x_2 \stackrel{C_1}{=} 1, x_3 \stackrel{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2

Let's try to

1. Weaken reason on non-falsified literal (but not last propagated)
2. Saturate weakened constraint
3. Resolve with conflicting constraint over propagated literal

Try to Reduce the Reason Constraint

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \stackrel{\mathsf{d}}{=} 0, x_2 \stackrel{C_1}{=} 1, x_3 \stackrel{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2

Let's try to

1. Weaken reason on non-falsified literal (but not last propagated)
2. Saturate weakened constraint
3. Resolve with conflicting constraint over propagated literal

$$\dfrac{\text{resolve } x_3\ \dfrac{\text{saturate}\ \dfrac{\text{weaken } x_2\ \dfrac{2x_1 + 2x_2 + 2x_3 + x_4 \geq 4}{2x_1 + 2x_3 + x_4 \geq 2}}{2x_1 + 2x_3 + x_4 \geq 2} \qquad 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3}{2\overline{x}_2 + x_4 \geq 1}$$

Try to Reduce the Reason Constraint

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \overset{\mathsf{d}}{=} 0, x_2 \overset{C_1}{=} 1, x_3 \overset{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2

Let's try to

1. Weaken reason on non-falsified literal (but not last propagated)
2. Saturate weakened constraint
3. Resolve with conflicting constraint over propagated literal

$$\cfrac{\cfrac{\cfrac{2x_1 + 2x_2 + 2x_3 + x_4 \geq 4}{2x_1 + 2x_3 + x_4 \geq 2}\ \text{weaken } x_2}{2x_1 + 2x_3 + x_4 \geq 2}\ \text{saturate} \qquad 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3}{2\overline{x}_2 + x_4 \geq 1}\ \text{resolve } x_3$$

Bummer! Still non-negative slack — not conflicting

Try Again to Reduce the Reason Constraint. . .

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \stackrel{\mathsf{d}}{=} 0, x_2 \stackrel{C_1}{=} 1, x_3 \stackrel{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2

Try Again to Reduce the Reason Constraint...

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \overset{\mathsf{d}}{=} 0, x_2 \overset{C_1}{=} 1, x_3 \overset{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2

$$\dfrac{\text{resolve } x_3 \dfrac{\text{saturate } \dfrac{\text{weaken } \{x_2, x_4\} \dfrac{2x_1 + 2x_2 + 2x_3 + x_4 \geq 4}{2x_1 + 2x_3 \geq 1}}{x_1 + x_3 \geq 1} \qquad 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3}{2\overline{x}_2 \geq 1}}{}$$

Try Again to Reduce the Reason Constraint...

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \stackrel{\mathsf{d}}{=} 0, x_2 \stackrel{C_1}{=} 1, x_3 \stackrel{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2

$$\dfrac{\text{resolve } x_3 \dfrac{\text{saturate } \dfrac{\text{weaken } \{x_2, x_4\} \dfrac{2x_1 + 2x_2 + 2x_3 + x_4 \geq 4}{2x_1 + 2x_3 \geq 1}}{x_1 + x_3 \geq 1} \qquad 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3}{2\overline{x}_2 \geq 1}$$

Negative slack — conflicting! Shows setting x_2 true was a mistake

Try Again to Reduce the Reason Constraint...

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \stackrel{\mathsf{d}}{=} 0, x_2 \stackrel{C_1}{=} 1, x_3 \stackrel{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2

$$\dfrac{\text{resolve } x_3 \quad \dfrac{\text{saturate} \quad \dfrac{\text{weaken } \{x_2, x_4\} \quad \dfrac{2x_1 + 2x_2 + 2x_3 + x_4 \geq 4}{2x_1 + 2x_3 \geq 1}}{x_1 + x_3 \geq 1}}{} \qquad 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3}{2\overline{x}_2 \geq 1}$$

Negative slack — conflicting! Shows setting x_2 true was a mistake

Backjump propagates to conflict without solver making any decisions
Done! Next conflict analysis will derive contradiction
(Or, in practice, terminate immediately at conflict without decisions)

Reason Reduction Using Saturation [CK05]

reduceSat($C_{\text{reason}}, C_{\text{learn}}, \ell, \rho$)

```
while slack(resolve(C_learn, C_reason, ℓ); ρ) ≥ 0 do
    ℓ' ← literal in C_reason \ {ℓ} not falsified by ρ;
    C_reason ← saturate(weaken(C_reason, ℓ'));
return C_reason;
```

Reason Reduction Using Saturation [CK05]

reduceSat$(C_{\text{reason}}, C_{\text{learn}}, \ell, \rho)$

while $\mathit{slack}(\text{resolve}(C_{\text{learn}}, C_{\text{reason}}, \ell); \rho) \geq 0$ **do**
  $\ell' \leftarrow$ literal in $C_{\text{reason}} \setminus \{\ell\}$ not falsified by ρ;
  $C_{\text{reason}} \leftarrow \text{saturate}(\text{weaken}(C_{\text{reason}}, \ell'))$;
return C_{reason};

Why does this work?

- Slack is subadditive

$$\mathit{slack}(c \cdot C + d \cdot D; \rho) \leq c \cdot \mathit{slack}(C; \rho) + d \cdot \mathit{slack}(D; \rho)$$

Reason Reduction Using Saturation [CK05]

reduceSat($C_{\mathrm{reason}}, C_{\mathrm{learn}}, \ell, \rho$)

while $\mathit{slack}(\mathsf{resolve}(C_{\mathrm{learn}}, C_{\mathrm{reason}}, \ell); \rho) \geq 0$ **do**
  $\ell' \leftarrow$ literal in $C_{\mathrm{reason}} \setminus \{\ell\}$ not falsified by ρ;
  $C_{\mathrm{reason}} \leftarrow \mathsf{saturate}(\mathsf{weaken}(C_{\mathrm{reason}}, \ell'))$;
return C_{reason};

Why does this work?

- Slack is subadditive

$$\mathit{slack}(c \cdot C + d \cdot D; \rho) \leq c \cdot \mathit{slack}(C; \rho) + d \cdot \mathit{slack}(D; \rho)$$

- By invariant have $\mathit{slack}(C_{\mathrm{learn}}; \rho) < 0$

Reason Reduction Using Saturation [CK05]

reduceSat$(C_{\mathrm{reason}}, C_{\mathrm{learn}}, \ell, \rho)$

while $\mathit{slack}(\mathsf{resolve}(C_{\mathrm{learn}}, C_{\mathrm{reason}}, \ell); \rho) \geq 0$ **do**
  $\ell' \leftarrow$ literal in $C_{\mathrm{reason}} \setminus \{\ell\}$ not falsified by ρ;
  $C_{\mathrm{reason}} \leftarrow \mathsf{saturate}(\mathsf{weaken}(C_{\mathrm{reason}}, \ell'))$;
return C_{reason};

Why does this work?

- Slack is subadditive

$$\mathit{slack}(c \cdot C + d \cdot D; \rho) \leq c \cdot \mathit{slack}(C; \rho) + d \cdot \mathit{slack}(D; \rho)$$

- By invariant have $\mathit{slack}(C_{\mathrm{learn}}; \rho) < 0$
- Weakening leaves $\mathit{slack}(C_{\mathrm{reason}}; \rho)$ unchanged

Reason Reduction Using Saturation [CK05]

reduceSat$(C_{\text{reason}}, C_{\text{learn}}, \ell, \rho)$

while $\mathit{slack}(\text{resolve}(C_{\text{learn}}, C_{\text{reason}}, \ell); \rho) \geq 0$ **do**
  $\ell' \leftarrow$ literal in $C_{\text{reason}} \setminus \{\ell\}$ not falsified by ρ;
  $C_{\text{reason}} \leftarrow \text{saturate}(\text{weaken}(C_{\text{reason}}, \ell'))$;
return C_{reason};

Why does this work?

- Slack is subadditive

$$\mathit{slack}(c \cdot C + d \cdot D; \rho) \leq c \cdot \mathit{slack}(C; \rho) + d \cdot \mathit{slack}(D; \rho)$$

- By invariant have $\mathit{slack}(C_{\text{learn}}; \rho) < 0$
- Weakening leaves $\mathit{slack}(C_{\text{reason}}; \rho)$ unchanged
- Saturation decreases slack — hit 0 when max #literals weakened

Pseudo-Boolean Conflict Analysis Pseudocode

analyzePBconflict($\mathcal{D}, \rho, C_{\mathrm{confl}}$)

$C_{\mathrm{learn}} \leftarrow C_{\mathrm{confl}}$;
while C_{learn} *not asserting* **and** $C_{\mathrm{learn}} \neq \perp$ **do**
  $\ell \leftarrow$ literal assigned last on trail ρ;
  if ℓ *propagated* **and** $\overline{\ell}$ *occurs in* C_{learn} **then**
    $C_{\mathrm{reason}} \leftarrow$ reason($\ell, \rho, \mathcal{D}$);
    $C_{\mathrm{reason}} \leftarrow$ **reduceSat**($C_{\mathrm{reason}}, C_{\mathrm{learn}}, \ell, \rho$);
    $C_{\mathrm{learn}} \leftarrow$ resolve($C_{\mathrm{learn}}, C_{\mathrm{reason}}, \ell$);
    $C_{\mathrm{learn}} \leftarrow$ saturate(C_{learn});
  $\rho \leftarrow \rho \setminus \{\ell\}$;
return C_{learn};

Reduction of reason new compared to CDCL — otherwise the same
Essentially conflict analysis used in SAT4J [LP10]

Some Problems Compared to CDCL

- Compared to clauses harder to detect propagation for constraints like
$$\sum_{i=1}^{n} x_i \geq n-1$$

Some Problems Compared to CDCL

- Compared to clauses harder to detect propagation for constraints like

$$\sum_{i=1}^{n} x_i \geq n - 1$$

- Generalized resolution for general pseudo-Boolean constraints
  $\Rightarrow$ lots of lcm computations
  $\Rightarrow$ coefficient sizes can explode (expensive arithmetic)

Some Problems Compared to CDCL

- Compared to clauses harder to detect propagation for constraints like

$$\sum_{i=1}^{n} x_i \geq n-1$$

- Generalized resolution for general pseudo-Boolean constraints
  $\Rightarrow$ lots of lcm computations
  $\Rightarrow$ coefficient sizes can explode (expensive arithmetic)

- For CNF inputs, degenerates to resolution!
  $\Rightarrow$ CDCL but with super-expensive data structures

The Cutting Planes Proof System

Cutting planes as defined in theory literature [CCT87] doesn't use saturation but instead division (a.k.a. Chvátal-Gomory cut)

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

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

Division $$\frac{\sum_i a_i \ell_i \geq A}{\sum_i \lceil a_i / c \rceil \ell_i \geq \lceil A/c \rceil}$$

The Cutting Planes Proof System

Cutting planes as defined in theory literature [CCT87] doesn't use saturation but instead division (a.k.a. Chvátal-Gomory cut)

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

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

Division $$\frac{\sum_i a_i \ell_i \geq A}{\sum_i \lceil a_i / c \rceil \ell_i \geq \lceil A/c \rceil}$$

- Cutting planes with division implicationally complete
- Cutting planes with saturation is **not** [VEG$^+$18]
- Can division yield stronger conflict analysis?

The Cutting Planes Proof System

Cutting planes as defined in theory literature [CCT87] doesn't use saturation but instead division (a.k.a. Chvátal-Gomory cut)

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

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

Division $$\frac{\sum_i a_i \ell_i \geq A}{\sum_i \lceil a_i / c \rceil \ell_i \geq \lceil A/c \rceil}$$

- Cutting planes with division implicationally complete
- Cutting planes with saturation is **not** [VEG+18]
- Can division yield stronger conflict analysis?
  (Used for integer linear programming in CutSat [JdM13])

Using Division to Reduce the Reason

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \stackrel{\mathsf{d}}{=} 0, x_2 \stackrel{C_1}{=} 1, x_3 \stackrel{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2

Using Division to Reduce the Reason

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \overset{\mathsf{d}}{=} 0, x_2 \overset{C_1}{=} 1, x_3 \overset{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2

1. Weaken reason on non-falsified literal(s) with coefficient not divisible by propagating literal coefficient
2. Divide weakened constraint by propagating literal coefficient
3. Resolve with conflicting constraint over propagated literal

Using Division to Reduce the Reason

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \overset{\mathsf{d}}{=} 0, x_2 \overset{C_1}{=} 1, x_3 \overset{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2

1. Weaken reason on non-falsified literal(s) with coefficient not divisible by propagating literal coefficient
2. Divide weakened constraint by propagating literal coefficient
3. Resolve with conflicting constraint over propagated literal

$$\dfrac{\text{resolve } x_3 \quad \dfrac{\text{divide by } 2 \quad \dfrac{\text{weaken } x_4 \quad 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4}{2x_1 + 2x_2 + 2x_3 \geq 3}}{x_1 + x_2 + x_3 \geq 2} \qquad 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3}{0 \geq 1}$$

Using Division to Reduce the Reason

$$C_1 \doteq 2x_1 + 2x_2 + 2x_3 + x_4 \geq 4$$
$$C_2 \doteq 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3$$

Trail $\rho = \{x_1 \stackrel{\mathsf{d}}{=} 0, x_2 \stackrel{C_1}{=} 1, x_3 \stackrel{C_1}{=} 1\}$ $\Rightarrow$ Conflict with C_2

1. Weaken reason on non-falsified literal(s) with coefficient not divisible by propagating literal coefficient
2. Divide weakened constraint by propagating literal coefficient
3. Resolve with conflicting constraint over propagated literal

$$\dfrac{\text{divide by } 2\ \dfrac{\text{weaken } x_4\ \dfrac{2x_1 + 2x_2 + 2x_3 + x_4 \geq 4}{2x_1 + 2x_2 + 2x_3 \geq 3}}{x_1 + x_2 + x_3 \geq 2} \qquad 2\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3}{0 \geq 1}\ \text{resolve } x_3$$

Terminate immediately!

Reason Reduction Using Division [EN18]

reduceDiv$(C_{\text{reason}}, C_{\text{learn}}, \ell, \rho)$

$c \leftarrow \mathit{coeff}(C_{\text{reason}}, \ell)$;
while $\mathit{slack}(\text{resolve}(C_{\text{learn}}, \text{divide}(C_{\text{reason}}, c), \ell); \rho) \geq 0$ **do**
    $\ell_j \leftarrow$ literal in $C_{\text{reason}} \setminus \{\ell\}$ such that $\overline{\ell}_j \notin \rho$ and $c \nmid \mathit{coeff}(C, \ell_j)$;
    $C_{\text{reason}} \leftarrow \text{weaken}(C_{\text{reason}}, \ell_j)$;
return $\text{divide}(C_{\text{reason}}, c)$;

Reason Reduction Using Division [EN18]

reduceDiv($C_{\text{reason}}, C_{\text{learn}}, \ell, \rho$)

$c \leftarrow \mathit{coeff}(C_{\text{reason}}, \ell)$;
while $\mathit{slack}(\text{resolve}(C_{\text{learn}}, \text{divide}(C_{\text{reason}}, c), \ell); \rho) \geq 0$ **do**
    $\ell_j \leftarrow$ literal in $C_{\text{reason}} \setminus \{\ell\}$ such that $\overline{\ell}_j \notin \rho$ and $c \nmid \mathit{coeff}(C, \ell_j)$;
    $C_{\text{reason}} \leftarrow \text{weaken}(C_{\text{reason}}, \ell_j)$;
return $\text{divide}(C_{\text{reason}}, c)$;

So now why does **this** work?

- Sufficient to get reason with slack 0 since
  1. $\mathit{slack}(C_{\text{learn}}; \rho) < 0$
  2. slack is subadditive

Reason Reduction Using Division [EN18]

reduceDiv$(C_{\text{reason}}, C_{\text{learn}}, \ell, \rho)$

$c \leftarrow \mathit{coeff}(C_{\text{reason}}, \ell)$;
while $\mathit{slack}(\text{resolve}(C_{\text{learn}}, \text{divide}(C_{\text{reason}}, c), \ell); \rho) \geq 0$ **do**
    $\ell_j \leftarrow$ literal in $C_{\text{reason}} \setminus \{\ell\}$ such that $\overline{\ell}_j \notin \rho$ and $c \nmid \mathit{coeff}(C, \ell_j)$;
    $C_{\text{reason}} \leftarrow \text{weaken}(C_{\text{reason}}, \ell_j)$;
return $\text{divide}(C_{\text{reason}}, c)$;

So now why does **this** work?

- Sufficient to get reason with slack 0 since
  1. $\mathit{slack}(C_{\text{learn}}; \rho) < 0$
  2. slack is subadditive
- Slack same after weakening $\Rightarrow$ always $0 \leq \mathit{slack}(C_{\text{reason}}; \rho) < c$

Reason Reduction Using Division [EN18]

reduceDiv$(C_{\text{reason}}, C_{\text{learn}}, \ell, \rho)$

$c \leftarrow \mathit{coeff}(C_{\text{reason}}, \ell)$;
while $\mathit{slack}(\text{resolve}(C_{\text{learn}}, \text{divide}(C_{\text{reason}}, c), \ell); \rho) \geq 0$ **do**
  $\ell_j \leftarrow$ literal in $C_{\text{reason}} \setminus \{\ell\}$ such that $\overline{\ell}_j \notin \rho$ and $c \nmid \mathit{coeff}(C, \ell_j)$;
  $C_{\text{reason}} \leftarrow \text{weaken}(C_{\text{reason}}, \ell_j)$;
return $\text{divide}(C_{\text{reason}}, c)$;

So now why does **this** work?

- Sufficient to get reason with slack 0 since
  1. $\mathit{slack}(C_{\text{learn}}; \rho) < 0$
  2. slack is subadditive
- Slack same after weakening $\Rightarrow$ always $0 \leq \mathit{slack}(C_{\text{reason}}; \rho) < c$
- After max #weakenings have $0 \leq \mathit{slack}(\text{divide}(C_{\text{reason}}, c); \rho) < 1$

Division vs. Saturation

- Higher conflict speed when PB reasoning doesn't help [EN18]
- Seems to perform better when PB reasoning crucial [EGNV18]
- Keeps coefficients small — can (often) do fixed-precision arithmetic
- But SAT4J still better for some circuit verification problems [LBD+20]
- And still equally hard to detect propagation
- Also, still degenerates to resolution for CNF inputs
- Sometimes very poor performance even on infeasible 0-1 LPs!

Some PB Solving Challenges I: Input Format

1. CNF: PB solvers degenerate to CDCL for CNF inputs — how to harness power of cutting planes in this setting?
   - Cardinality constraint detection proposed as preprocessing [BLLM14] or inprocessing [EN20]
   - Not yet competitive in practice

Some PB Solving Challenges I: Input Format

1. CNF: PB solvers degenerate to CDCL for CNF inputs — how to harness power of cutting planes in this setting?
   - Cardinality constraint detection proposed as preprocessing [BLLM14] or inprocessing [EN20]
   - Not yet competitive in practice
2. Linear programming: Sometimes very poor performance even on infeasible 0-1 LPs!
   - Unclear why — very easy for cutting planes in theory
   - Work on addressing this in [DGN21]

Some PB Solving Challenges I: Input Format

1. CNF: PB solvers degenerate to CDCL for CNF inputs — how to harness power of cutting planes in this setting?
   - Cardinality constraint detection proposed as preprocessing [BLLM14] or inprocessing [EN20]
   - Not yet competitive in practice
2. Linear programming: Sometimes very poor performance even on infeasible 0-1 LPs!
   - Unclear why — very easy for cutting planes in theory
   - Work on addressing this in [DGN21]
3. Preprocessing/presolving: Important in SAT solving and integer linear programming, but not done in PB solvers — why?
   - Follow up on preliminary work on PB preprocessing in [MLM09]?
   - Use presolver PaPILO [PaP] from mixed integer linear programming (MIP) solver SCIP [SCI]?

Some PB Solving Challenges I: Input Format

1. CNF: PB solvers degenerate to CDCL for CNF inputs — how to harness power of cutting planes in this setting?
   - Cardinality constraint detection proposed as preprocessing [BLLM14] or inprocessing [EN20]
   - Not yet competitive in practice
2. Linear programming: Sometimes very poor performance even on infeasible 0-1 LPs!
   - Unclear why — very easy for cutting planes in theory
   - Work on addressing this in [DGN21]
3. Preprocessing/presolving: Important in SAT solving and integer linear programming, but not done in PB solvers — why?
   - Follow up on preliminary work on PB preprocessing in [MLM09]?
   - Use presolver PaPILO [PaP] from mixed integer linear programming (MIP) solver SCIP [SCI]?
4. Robustness: Make PB solvers less sensitive to presence of extra constraints (anecdotally, CDCL solvers seem more stable)

Some PB Solving Challenges II: Conflict Analysis

1. Choice of Boolean rule:
    - Division, saturation, or select adaptively?
    - Or some other cut rule from ILP?
    - Try to avoid irrelevant literals? [LMMW20]

Some PB Solving Challenges II: Conflict Analysis

1. Choice of Boolean rule:
   - Division, saturation, or select adaptively?
   - Or some other cut rule from ILP?
   - Try to avoid irrelevant literals? [LMMW20]
2. Many more degrees of freedom than in CDCL:
   - Skip resolution steps when slack very negative?
   - How aggressively to weaken reason in reduction step? [LMW20]
   - Learn general PB constraints or more limited form?
   - How far to backjump when choice of several levels?
   - How large precision to use in integer arithmetic?

Some PB Solving Challenges II: Conflict Analysis

1. Choice of Boolean rule:
    - Division, saturation, or select adaptively?
    - Or some other cut rule from ILP?
    - Try to avoid irrelevant literals? [LMMW20]
2. Many more degrees of freedom than in CDCL:
    - Skip resolution steps when slack very negative?
    - How aggressively to weaken reason in reduction step? [LMW20]
    - Learn general PB constraints or more limited form?
    - How far to backjump when choice of several levels?
    - How large precision to use in integer arithmetic?
3. Do constraint minimization à la [SB09, HS09]?

Some PB Solving Challenges II: Conflict Analysis

1. Choice of Boolean rule:
   - Division, saturation, or select adaptively?
   - Or some other cut rule from ILP?
   - Try to avoid irrelevant literals? [LMMW20]
2. Many more degrees of freedom than in CDCL:
   - Skip resolution steps when slack very negative?
   - How aggressively to weaken reason in reduction step? [LMW20]
   - Learn general PB constraints or more limited form?
   - How far to backjump when choice of several levels?
   - How large precision to use in integer arithmetic?
3. Do constraint minimization à la [SB09, HS09]?
4. How to assess quality of learned constraints?

Some PB Solving Challenges II: Conflict Analysis

1. Choice of Boolean rule:
   - Division, saturation, or select adaptively?
   - Or some other cut rule from ILP?
   - Try to avoid irrelevant literals? [LMMW20]
2. Many more degrees of freedom than in CDCL:
   - Skip resolution steps when slack very negative?
   - How aggressively to weaken reason in reduction step? [LMW20]
   - Learn general PB constraints or more limited form?
   - How far to backjump when choice of several levels?
   - How large precision to use in integer arithmetic?
3. Do constraint minimization à la [SB09, HS09]?
4. How to assess quality of learned constraints?
5. Theoretical potential & limitations poorly understood [VEG+18]
   - Separations in power between different methods of PB reasoning?
   - In particular, is division-based reasoning stronger than saturation-based reasoning? [GNY19]

Some PB Solving Challenges III: Solver Heuristics

Many heuristics more or less copied from CDCL — maybe tailor more carefully to PB setting?

1. Variable selection: VSIDS [MMZ+01] or VMTF [Rya04] or something else?

Some PB Solving Challenges III: Solver Heuristics

Many heuristics more or less copied from CDCL — maybe tailor more carefully to PB setting?

1. Variable selection: VSIDS [MMZ+01] or VMTF [Rya04] or something else?
2. Variable bumping: Consider different bumping score depending on
   - whether literal falsified,
   - whether literal cancels,
   - coefficient of literal and/or degree of constraint?

Some PB Solving Challenges III: Solver Heuristics

Many heuristics more or less copied from CDCL — maybe tailor more carefully to PB setting?

1. Variable selection: VSIDS [MMZ+01] or VMTF [Rya04] or something else?
2. Variable bumping: Consider different bumping score depending on
   - whether literal falsified,
   - whether literal cancels,
   - coefficient of literal and/or degree of constraint?
3. Phase saving: Standard as in [PD07], multiple phases [BF20], or something else?

Some PB Solving Challenges III: Solver Heuristics

Many heuristics more or less copied from CDCL — maybe tailor more carefully to PB setting?

1. Variable selection: VSIDS [MMZ+01] or VMTF [Rya04] or something else?
2. Variable bumping: Consider different bumping score depending on
   - whether literal falsified,
   - whether literal cancels,
   - coefficient of literal and/or degree of constraint?
3. Phase saving: Standard as in [PD07], multiple phases [BF20], or something else?
4. Different "modes" for SAT-focused and UNSAT-focused search?

See [Wal20] for a first in-depth investigation of some of these questions

Some PB Solving Challenges IV: Efficiency and Correctness

1. Efficient unit propagation for PB constraints is a major challenge — latest news in [Dev20], but still much left to do

Some PB Solving Challenges IV: Efficiency and Correctness

1. Efficient unit propagation for PB constraints is a major challenge — latest news in [Dev20], but still much left to do
2. Efficient detection of assertiveness during conflict analysis

Some PB Solving Challenges IV: Efficiency and Correctness

1. Efficient unit propagation for PB constraints is a major challenge — latest news in [Dev20], but still much left to do
2. Efficient detection of assertiveness during conflict analysis
3. Efficient and concise proof logging for pseudo-Boolean solving (shameless self-plug: ongoing work on pseudo-Boolean proof checker VeriPB [Ver, GMN20b] in [EGMN20, GMN20a, GMM+20, GN21, BGMN22, GMNO22])

Some References for Further Reading (and Watching)

Handbook of Satisfiability [BHvMW21]

- Chapter 7: Proof Complexity and SAT Solving
- Chapter 23: MaxSAT, Hard and Soft Constraints
- Chapter 24: Maximum Satisfiability
- Chapter 28: Pseudo-Boolean and Cardinality Constraints

Some References for Further Reading (and Watching)

Handbook of Satisfiability [BHvMW21]

- Chapter 7: Proof Complexity and SAT Solving
- Chapter 23: MaxSAT, Hard and Soft Constraints
- Chapter 24: Maximum Satisfiability
- Chapter 28: Pseudo-Boolean and Cardinality Constraints

Video tutorials on pseudo-Boolean solving

From the *Satisfiability: Theory, Practice, and Beyond* program at UC Berkeley in spring 2021
`https://tinyurl.com/PBSATtutorial`
[Try to cover as much of this as possible today]

Summing up

- Pseudo-Boolean framework expressive and powerful
- Can be approached using successful conflict-driven paradigm from SAT solving
- In theory, potential for exponential increase in performance
- In practice, some highly nontrivial challenges regarding
  - Algorithm design
  - Efficient implementation
  - Theoretical understanding
- But maybe also quite a bit of low-hanging fruit? (And clause-based SAT solving took 50+ years to get right)
- In any case, lots of fun questions to work on! ☺

Summing up

- Pseudo-Boolean framework expressive and powerful
- Can be approached using successful conflict-driven paradigm from SAT solving
- In theory, potential for exponential increase in performance
- In practice, some highly nontrivial challenges regarding
  - Algorithm design
  - Efficient implementation
  - Theoretical understanding
- But maybe also quite a bit of low-hanging fruit?
  (And clause-based SAT solving took 50+ years to get right)
- In any case, lots of fun questions to work on! ☺

Thank you for your attention!

References I

[Bar95] Peter Barth. A Davis-Putnam based enumeration algorithm for linear pseudo-Boolean optimization. Technical Report MPI-I-95-2-003, Max-Planck-Institut für Informatik, January 1995.

[BF20] Armin Biere and Mathias Fleury. Chasing target phases. Presented at the workshop *Pragmatics of SAT 2020*. Paper available at `http://fmv.jku.at/papers/BiereFleury-POS20.pdf`, July 2020.

[BGMN22] Bart Bogaerts, Stephan Gocht, Ciaran McCreesh, and Jakob Nordström. Certified symmetry and dominance breaking for combinatorial optimisation. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI '22)*, pages 3698–3707, February 2022.

[BH02] Endre Boros and Peter L. Hammer. Pseudo-Boolean optimization. *Discrete Applied Mathematics*, 123(1–3):155–225, November 2002.

[BHvMW21] Armin Biere, Marijn J. H. Heule, Hans van Maaren, and Toby Walsh, editors. *Handbook of Satisfiability*, volume 336 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2nd edition, February 2021.

References II

[BLLM14] Armin Biere, Daniel Le Berre, Emmanuel Lonca, and Norbert Manthey. Detecting cardinality constraints in CNF. 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 285–301. Springer, July 2014.

[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*.

[Dev20] Jo Devriendt. Watched propagation of 0-1 integer linear constraints. In *Proceedings of the 26th International Conference on Principles and Practice of Constraint Programming (CP '20)*, volume 12333 of *Lecture Notes in Computer Science*, pages 160–176. Springer, September 2020.

References III

[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.

[DGN21] Jo Devriendt, Ambros Gleixner, and Jakob Nordström. Learn to relax: Integrating 0-1 integer linear programming with pseudo-Boolean conflict-driven search. *Constraints*, 26(1–4):26–55, October 2021. Preliminary version in *CPAIOR '20*.

[DLL62] Martin Davis, George Logemann, and Donald Loveland. A machine program for theorem proving. *Communications of the ACM*, 5(7):394–397, July 1962.

[DP60] Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7(3):201–215, 1960.

[EGMN20] Jan Elffers, Stephan Gocht, Ciaran McCreesh, and Jakob Nordström. Justifying all differences using pseudo-Boolean reasoning. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI '20)*, pages 1486–1494, February 2020.

References IV

[EGNV18] Jan Elffers, Jesús Giráldez-Cru, Jakob Nordström, and Marc Vinyals. Using combinatorial benchmarks to probe the reasoning power of pseudo-Boolean solvers. 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 75–93. Springer, July 2018.

[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)*, pages 1495–1503, February 2020.

[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.

References V

[GMM+20] Stephan Gocht, Ross McBride, Ciaran McCreesh, Jakob Nordström, Patrick Prosser, and James Trimble. Certifying solvers for clique and maximum common (connected) subgraph problems. In *Proceedings of the 26th International Conference on Principles and Practice of Constraint Programming (CP '20)*, volume 12333 of *Lecture Notes in Computer Science*, pages 338–357. Springer, September 2020.

[GMN20a] Stephan Gocht, Ciaran McCreesh, and Jakob Nordström. Subgraph isomorphism meets cutting planes: Solving with certified solutions. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI '20)*, pages 1134–1140, July 2020.

[GMN20b] Stephan Gocht, Ciaran McCreesh, and Jakob Nordström. VeriPB: The easy way to make your combinatorial search algorithm trustworthy. Presented at the workshop *From Constraint Programming to Trustworthy AI* at the *26th International Conference on Principles and Practice of Constraint Programming (CP '20)*. Paper available at `http://www.cs.ucc.ie/~bg6/cptai/2020/papers/CPTAI_2020_paper_2.pdf`, September 2020.

References VI

[GMNO22] Stephan Gocht, Ruben Martins, Jakob Nordström, and Andy Oertel. Certified CNF translations for pseudo-Boolean solving. In *Proceedings of the 25th International Conference on Theory and Applications of Satisfiability Testing (SAT '22)*, volume 236 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 16:1–16:25, August 2022.

[GN21] Stephan Gocht and Jakob Nordström. Certifying parity reasoning efficiently using pseudo-Boolean proofs. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI '21)*, pages 3768–3777, February 2021.

[GNY19] Stephan Gocht, Jakob Nordström, and Amir Yehudayoff. On division versus saturation in pseudo-Boolean solving. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI '19)*, pages 1711–1718, August 2019.

[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.

References VII

[HS09] Hyojung Han and Fabio Somenzi. On-the-fly clause improvement. In *Proceedings of the 12th International Conference on Theory and Applications of Satisfiability Testing (SAT '09)*, volume 5584 of *Lecture Notes in Computer Science*, pages 209–222. Springer, July 2009.

[JdM13] Dejan Jovanovic and Leonardo de Moura. Cutting to the chase solving linear integer arithmetic. *Journal of Automated Reasoning*, 51(1):79–108, June 2013. Preliminary version in *CADE-23*.

[LBD+20] Vincent Liew, Paul Beame, Jo Devriendt, Jan Elffers, and Jakob Nordström. Verifying properties of bit-vector multiplication using cutting planes reasoning. In *Proceedings of the 20th Conference on Formal Methods in Computer-Aided Design (FMCAD '20)*, pages 194–204, September 2020.

[LMMW20] Daniel Le Berre, Pierre Marquis, Stefan Mengel, and Romain Wallon. On irrelevant literals in pseudo-Boolean constraint learning. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI '20)*, pages 1148–1154, July 2020.

References VIII

[LMW20] Daniel Le Berre, Pierre Marquis, and Romain Wallon. On weakening strategies for PB solvers. In *Proceedings of the 23rd International Conference on Theory and Applications of Satisfiability Testing (SAT '20)*, volume 12178 of *Lecture Notes in Computer Science*, pages 322–331. Springer, July 2020.

[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.

[MLM09] Ruben Martins, Inês Lynce, and Vasco M. Manquinho. Preprocessing in pseudo-Boolean optimization: An experimental evaluation. In *Proceedings of the 8th International Workshop on Constraint Modelling and Reformulation (ModRef '09)*, pages 87–101, September 2009. Available at `https://www-users.cs.york.ac.uk/~frisch/ModRef/09/proceedings.pdf`.

[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 IX

[MMZ+01] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an efficient SAT solver. In *Proceedings of the 38th Design Automation Conference (DAC '01)*, pages 530–535, June 2001.

[MS99] João P. Marques-Silva and Karem A. Sakallah. GRASP: A search algorithm for propositional satisfiability. *IEEE Transactions on Computers*, 48(5):506–521, May 1999. Preliminary version in *ICCAD '96*.

[PaP] PaPILO — parallel presolve for integer and linear optimization. `https://github.com/lgottwald/PaPILO`.

[PD07] Knot Pipatsrisawat and Adnan Darwiche. A lightweight component caching scheme for satisfiability solvers. In *Proceedings of the 10th International Conference on Theory and Applications of Satisfiability Testing (SAT '07)*, volume 4501 of *Lecture Notes in Computer Science*, pages 294–299. Springer, May 2007.

[Rya04] Lawrence Ryan. Efficient algorithms for clause-learning SAT solvers. Master's thesis, Simon Fraser University, February 2004. Available at `https://www.cs.sfu.ca/~mitchell/papers/ryan-thesis.ps`.

References X

[SB09] Niklas Sörensson and Armin Biere. Minimizing learned clauses. In *Proceedings of the 12th International Conference on Theory and Applications of Satisfiability Testing (SAT '09)*, volume 5584 of *Lecture Notes in Computer Science*, pages 237–243. Springer, July 2009.

[SCI] SCIP: Solving constraint integer programs. `http://scip.zib.de/`.

[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.

References XI

[Ver] VeriPB: Verifier for pseudo-Boolean proofs. https://gitlab.com/MIAOresearch/software/VeriPB.

[Wal20] Romain Wallon. *Pseudo-Boolean Reasoning and Compilation*. PhD thesis, Université d'Artois, 2020. Available at http://www.lix.polytechnique.fr/~wallon/reports/phdthesis.pdf.

[Wil76] H. P. Williams. Fourier-Motzkin elimination extension to integer programming problems. *Journal of Combinatorial Theory, Series A*, 21(1):118–123, July 1976.