Certified CNF Translations for Pseudo-Boolean Solving

Andy Oertel

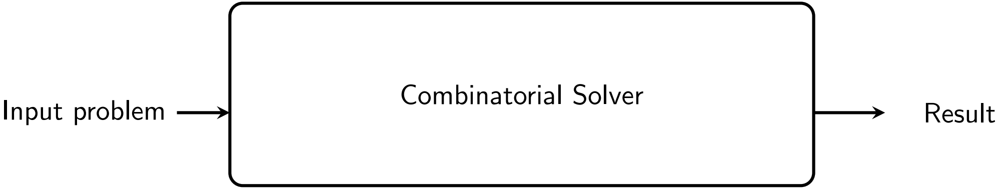
Lund University and
University of Copenhagen

SAT+SMT Winter School 2022

December 16, 2022

Joint SAT'22 paper with Stephan Gocht, Ruben Martins and Jakob Nordström

Combinatorial Solving



- Revolution in last couple of decades in **combinatorial solvers** for
  - Boolean satisfiability (SAT) [BHvMW21]
  - Constraint programming (CP) [RvBW06]
  - Mixed integer linear programming (MIP) [AW13]
- Solve hard problems in practice
- **Solvers are sometimes wrong** (even commercial ones) [BLB10, CKSW13, AGJ$^+$18, GSD19, GS19]

Correctness of Combinatorial Solving

Correctness of Combinatorial Solving

- Testing:
  - Can only show presence of bugs, not absence
  - No guarantees of correctness

Correctness of Combinatorial Solving

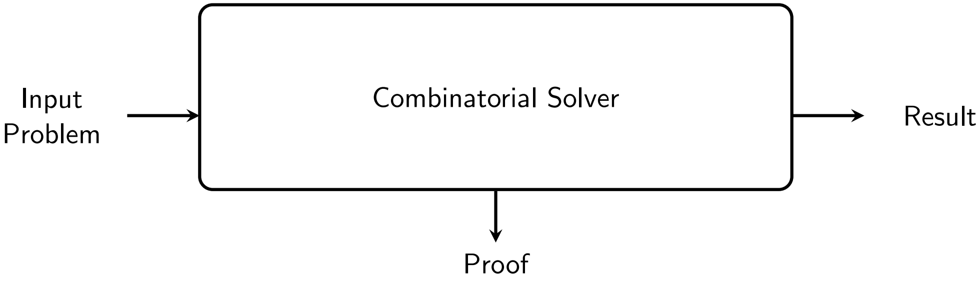
- Testing:
  - Can only show presence of bugs, not absence
  - No guarantees of correctness
- Formal verification:
  - Guarantee that solver adheres specification
  - Nowhere near doing this for modern solvers

Correctness of Combinatorial Solving

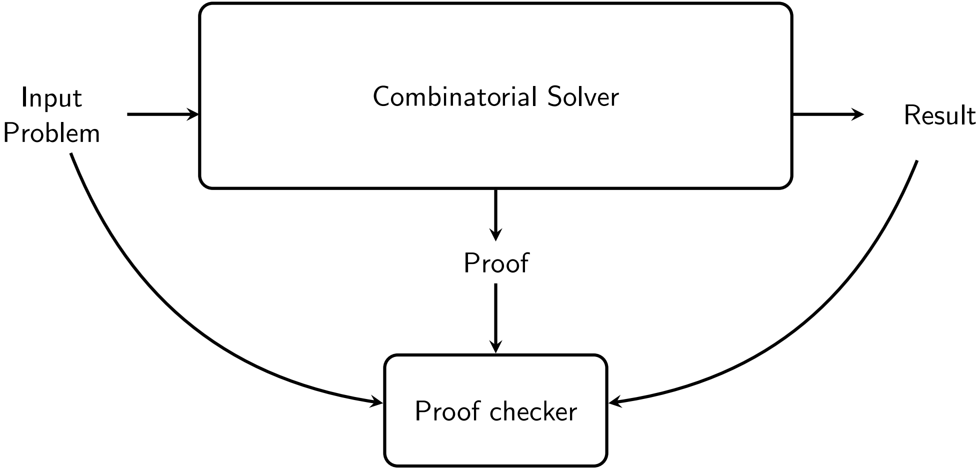
- Testing:
  - Can only show presence of bugs, not absence
  - No guarantees of correctness
- Formal verification:
  - Guarantee that solver adheres specification
  - Nowhere near doing this for modern solvers
- Proof logging (our approach):
  - Guarantee that execution was correct
  - Moderate overhead for writing solver

Certifying Results with Proof Logging

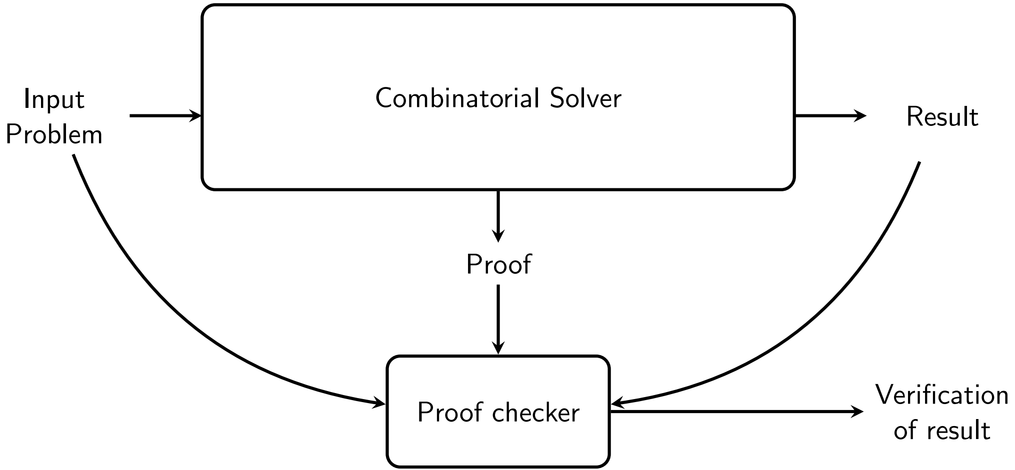
Certifying Results with Proof Logging



Certifying Results with Proof Logging



Certifying Results with Proof Logging



The Pseudo-Boolean (PB) Problem

Pseudo-Boolean Formula:

$$x_1 + x_2 + x_3 \geq 2$$
$$x_2 + x_3 + x_4 \geq 2$$
$$\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 + \overline{x}_4 \geq 3$$

→ Pseudo-Boolean Solver → Result: SAT/UNSAT

- **Input:** Pseudo-Boolean formula (or 0-1 integer linear program)
  - Collection of 0-1 integer linear constraints

The Pseudo-Boolean (PB) Problem

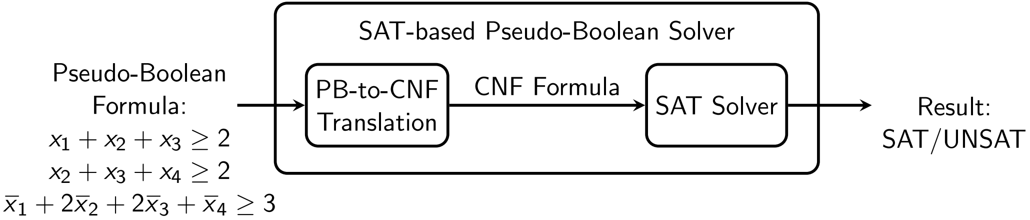
Pseudo-Boolean Formula:

$$x_1 + x_2 + x_3 \geq 2$$
$$x_2 + x_3 + x_4 \geq 2$$
$$\overline{x}_1 + 2\overline{x}_2 + 2\overline{x}_3 + \overline{x}_4 \geq 3$$

→ Pseudo-Boolean Solver → Result: SAT/UNSAT

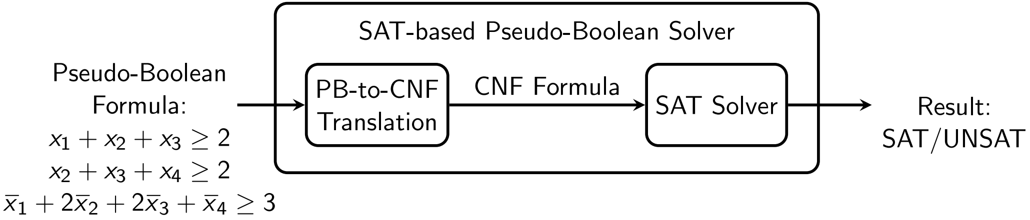
- Input: Pseudo-Boolean formula (or 0-1 integer linear program)
  - Collection of 0-1 integer linear constraints
- Pseudo-Boolean solver:
  - Native PB solver: Sat4j [LP10], RoundingSAT [EN18]
  - SAT-based PB solver: MiniSAT+ [ES06], Open-WBO [MML14], NaPS [SN15]

The Pseudo-Boolean (PB) Problem

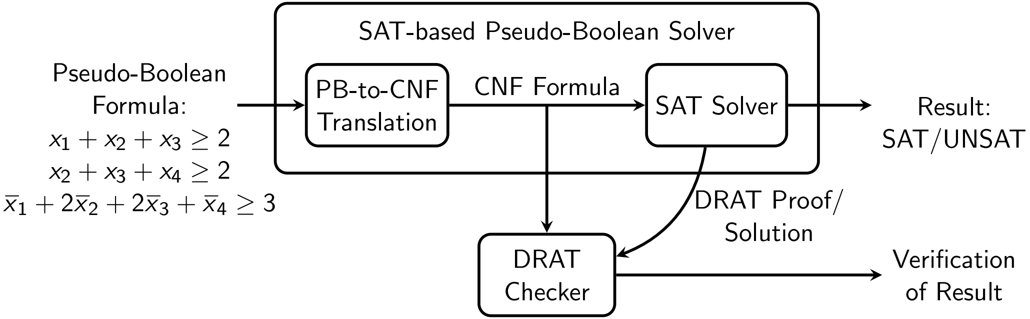


- Input: Pseudo-Boolean formula (or 0-1 integer linear program)
  - Collection of 0-1 integer linear constraints
- Pseudo-Boolean solver:
  - Native PB solver: Sat4j [LP10], RoundingSAT [EN18]
  - **SAT-based PB solver:** MiniSAT+ [ES06], Open-WBO [MML14], NaPS [SN15]

Certifying Correctness

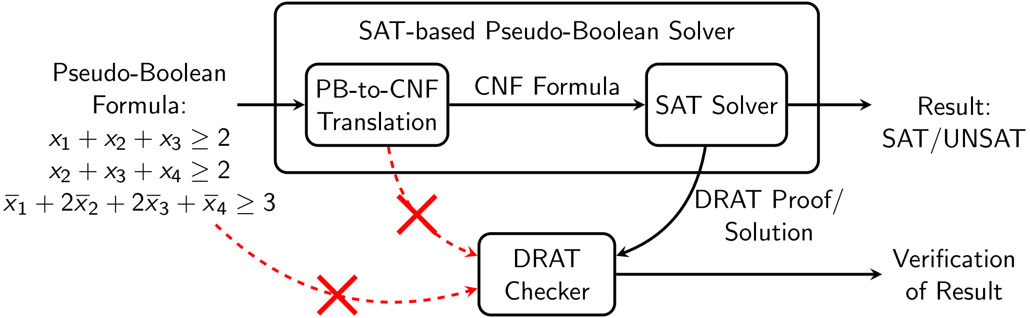


Certifying Correctness



- Correctness of SAT solver result can be certified [HHW13a, HHW13b, WHH14]

Certifying Correctness



- Correctness of SAT solver result can be certified [HHW13a, HHW13b, WHH14]
- **PB-to-CNF translation uncertified!**

Pseudo-Boolean Proof Logging

- **Multi-purpose** proof format
- Allows easy proof logging for
  - Reasoning with pseudo-Boolean constraints (by design)
  - SAT solving (including advanced techniques) [GN21, BGMN22]
  - Constraint Programming [EGMN20, GMN22]
  - Subgraph problems [GMN20, GMM+20]

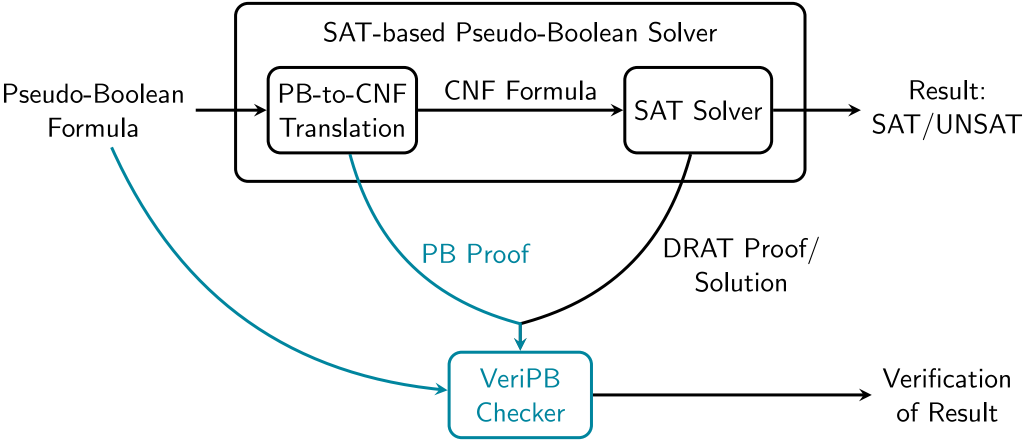
Pseudo-Boolean Proof Logging

- **Multi-purpose** proof format
- Allows easy proof logging for
  - Reasoning with pseudo-Boolean constraints (by design)
  - SAT solving (including advanced techniques) [GN21, BGMN22]
  - Constraint Programming [EGMN20, GMN22]
  - Subgraph problems [GMN20, GMM+20]

This Work

- Proof logging for translating pseudo-Boolean constraints to CNF
- **General framework** to certify many different encodings
- Promising foundation for certifying MaxSAT solving and PB optimization

Workflow


SAT-based Pseudo-Boolean Solver
Pseudo-Boolean Formula
PB-to-CNF Translation
CNF Formula
SAT Solver
Result: SAT/UNSAT
PB Proof
DRAT Proof/ Solution
VeriPB Checker
Verification of Result

Idea of Proof Logging

General idea: Want proof that result of a solver is correct

- How does a proof look like?

Idea of Proof Logging

General idea: Want proof that result of a solver is correct

- How does a proof look like?
- For decision problems (like SAT):
  - Satisfiable instance: assignment that satisfies formula
  - Unsatisfiable instance:
    - derivation of contradiction (empty clause) using formal proof system
    - contradiction means original formula must be inconsistent

Basic Notation

- Boolean variable x: with domain 0 (false) and 1 (true)
- Literal ℓ: x or negation $\overline{x} = 1 - x$
- Clause: disjunction of literals

Proof Logging in SAT

How to extract proof from SAT solver execution, e.g. CDCL?

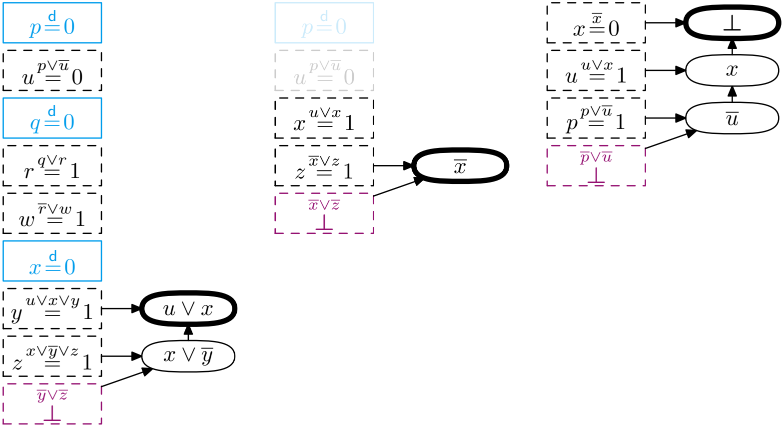
- Deriving $\perp$ proves unsatisfiability

Proof Logging in SAT

How to extract proof from SAT solver execution, e.g. CDCL?

- Deriving $\perp$ proves unsatisfiability

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



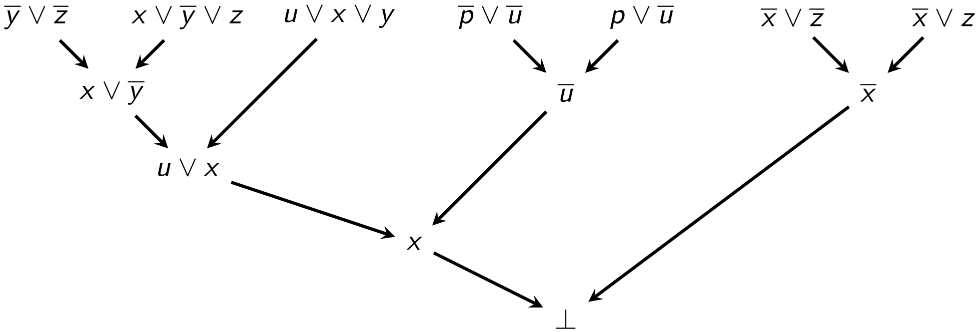
Proof Logging in SAT

How to extract proof from SAT solver execution, e.g. CDCL?

- Deriving $\perp$ proves unsatisfiability

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

- Naïve approach: write down every resolution step



Reverse Unit Propagation (RUP)

Reverse unit propagation (RUP) clause [GN03, Van08a]

C is a **reverse unit propagation (RUP)** clause w.r.t. the formula F if

- assigning C to false,
- then unit propagating on F until saturation
- leads to contradiction

If so, F clearly implies C, and condition easy to verify efficiently

Reverse Unit Propagation (RUP)

Reverse unit propagation (RUP) clause [GN03, Van08a]

C is a **reverse unit propagation (RUP)** clause w.r.t. the formula F if

- assigning C to false,
- then unit propagating on F until saturation
- leads to contradiction

If so, F clearly implies C, and condition easy to verify efficiently

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

Sequence of reverse unit propagation (RUP) clauses:

1. $u \lor x$
2. $\overline{x}$
3. $\perp$

Reverse Unit Propagation (RUP)

Reverse unit propagation (RUP) clause [GN03, Van08a]

C is a **reverse unit propagation (RUP)** clause w.r.t. the formula F if

- assigning C to false,
- then unit propagating on F until saturation
- leads to contradiction

If so, F clearly implies C, and condition easy to verify efficiently

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

Sequence of reverse unit propagation (RUP) clauses:

1. $u \vee x$
2. $\overline{x}$
3. $\perp$

Reverse Unit Propagation (RUP)

Reverse unit propagation (RUP) clause [GN03, Van08a]

C is a **reverse unit propagation (RUP)** clause w.r.t. the formula F if

- assigning C to false,
- then unit propagating on F until saturation
- leads to contradiction

If so, F clearly implies C, and condition easy to verify efficiently

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

Sequence of reverse unit propagation (RUP) clauses:

1. $u \vee x$
2. $\overline{x}$
3. $\perp$

Reverse Unit Propagation (RUP)

Reverse unit propagation (RUP) clause [GN03, Van08a]

C is a **reverse unit propagation (RUP)** clause w.r.t. the formula F if

- assigning C to false,
- then unit propagating on F until saturation
- leads to contradiction

If so, F clearly implies C, and condition easy to verify efficiently

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

Sequence of reverse unit propagation (RUP) clauses:

1. $u \vee x$
2. $\overline{x}$
3. $\perp$

Reverse Unit Propagation (RUP)

Reverse unit propagation (RUP) clause [GN03, Van08a]

C is a **reverse unit propagation (RUP)** clause w.r.t. the formula F if

- assigning C to false,
- then unit propagating on F until saturation
- leads to contradiction

If so, F clearly implies C, and condition easy to verify efficiently

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

Sequence of reverse unit propagation (RUP) clauses:

1. $u \lor x$
2. $\overline{x}$
3. $\perp$

Reverse Unit Propagation (RUP)

Reverse unit propagation (RUP) clause [GN03, Van08a]

C is a **reverse unit propagation (RUP)** clause w.r.t. the formula F if

- assigning C to false,
- then unit propagating on F until saturation
- leads to contradiction

If so, F clearly implies C, and condition easy to verify efficiently

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

Sequence of reverse unit propagation (RUP) clauses:

1. $u \lor x$
2. $\overline{x}$
3. $\perp$

Reverse Unit Propagation (RUP)

Reverse unit propagation (RUP) clause [GN03, Van08a]

C is a **reverse unit propagation (RUP)** clause w.r.t. the formula F if

- assigning C to false,
- then unit propagating on F until saturation
- leads to contradiction

If so, F clearly implies C, and condition easy to verify efficiently

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

Sequence of reverse unit propagation (RUP) clauses:

1. $u \vee x$
2. $\overline{x}$
3. $\perp$

Reverse Unit Propagation (RUP)

Reverse unit propagation (RUP) clause [GN03, Van08a]

C is a **reverse unit propagation (RUP)** clause w.r.t. the formula F if

- assigning C to false,
- then unit propagating on F until saturation
- leads to contradiction

If so, F clearly implies C, and condition easy to verify efficiently

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

Sequence of reverse unit propagation (RUP) clauses:

1. $u \vee x$
2. $\overline{x}$
3. $\perp$

Reverse Unit Propagation (RUP)

Reverse unit propagation (RUP) clause [GN03, Van08a]

C is a **reverse unit propagation (RUP)** clause w.r.t. the formula F if

- assigning C to false,
- then unit propagating on F until saturation
- leads to contradiction

If so, F clearly implies C, and condition easy to verify efficiently

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

Sequence of reverse unit propagation (RUP) clauses:

1. $u \vee x$
2. $\overline{x}$
3. $\perp$

Reverse Unit Propagation (RUP)

Reverse unit propagation (RUP) clause [GN03, Van08a]

C is a **reverse unit propagation (RUP)** clause w.r.t. the formula F if

- assigning C to false,
- then unit propagating on F until saturation
- leads to contradiction

If so, F clearly implies C, and condition easy to verify efficiently

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

Sequence of reverse unit propagation (RUP) clauses:

1. $u \vee x$
2. $\overline{x}$
3. $\perp$

Reverse Unit Propagation (RUP)

Reverse unit propagation (RUP) clause [GN03, Van08a]

C is a **reverse unit propagation (RUP)** clause w.r.t. the formula F if

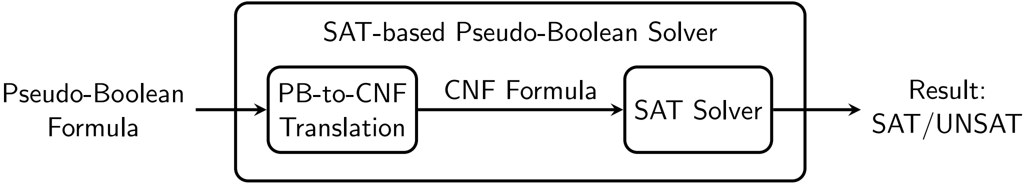
- assigning C to false,
- then unit propagating on F until saturation
- leads to contradiction

If so, F clearly implies C, and condition easy to verify efficiently

Fact

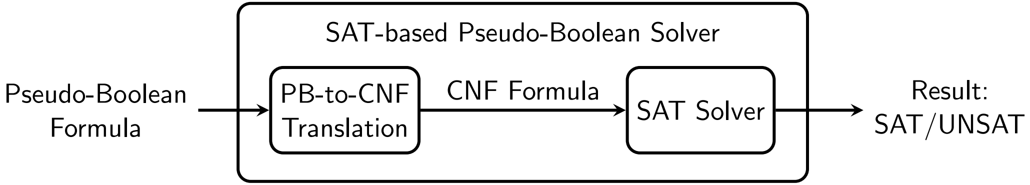
All learned clauses generated by CDCL solver are RUP clauses.

Proof Logging for SAT-based Pseudo-Boolean Solving



- Assignment is still sufficient if satisfiable

Proof Logging for SAT-based Pseudo-Boolean Solving



- Assignment is still sufficient if satisfiable
- Derivation of contradiction from pseudo-Boolean formula if unsatisfiable
  - SAT solver proof only derives contradiction from CNF translation
  - Proof that derives CNF translation is necessary

Pseudo-Boolean Notation

- **Pseudo-Boolean (PB) constraint:** integer linear inequality over literals

$$3x_1 + 2x_2 + 5\overline{x}_3 \geq 5$$

- **Equality constraint:** syntactic sugar for 2 inequalities

$$3x_1 + 2x_2 + 5\overline{x}_3 = 5 \longrightarrow \begin{array}{l} 3x_1 + 2x_2 + 5\overline{x}_3 \geq 5 \\ 3x_1 + 2x_2 + 5\overline{x}_3 \leq 5 \end{array}$$

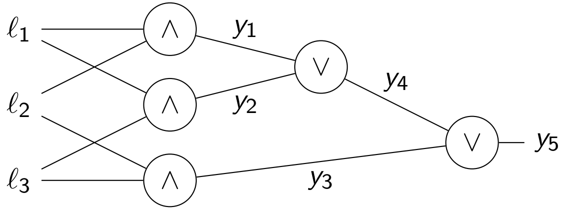
- **Clause:** disjunction of literals / at-least-one constraint

$$x_1 \vee \overline{x}_2 \vee \overline{x}_3 \iff x_1 + \overline{x}_2 + \overline{x}_3 \geq 1$$

Translating PB to CNF: Outline

$$\ell_1 + \ell_2 + \ell_3 \geq 2$$

1. Construct circuit evaluating left-hand side of pseudo-Boolean constraint



2. Encode circuit to CNF using so-called Tseitin translation

$$\overline{y_3} \vee \ell_2 \qquad \overline{y_3} \vee \ell_3 \qquad y_3 \vee \overline{\ell_2} \vee \overline{\ell_3} \qquad \cdots$$

3. Enforcing inequality

$$y_5$$

Translating PB to CNF: Step 1

1. **Construct circuit evaluating left-hand side of pseudo-Boolean constraint**

- Several approaches to construct logical circuit evaluating PB constraint
  - Sequential counter [Sin05], totalizer [BB03], adder network [ES06], ...

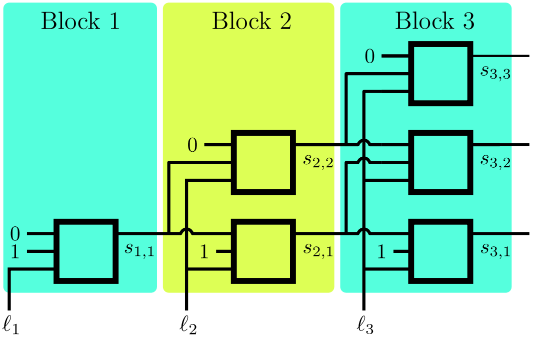
Translating PB to CNF: Step 1

1. **Construct circuit evaluating left-hand side of pseudo-Boolean constraint**

- Several approaches to construct logical circuit evaluating PB constraint
  - **Sequential counter** [Sin05], totalizer [BB03], adder network [ES06], ...

Example: $\ell_1 + \ell_2 + \ell_3 \geq 2$

Meaning of $s_{i,j}$ variable:
$s_{i,j}$ true if and only if
$\ell_1 + \ldots + \ell_i \geq j$



Translating PB to CNF: Step 2

2. **Encode circuit to CNF using so-called Tseitin translations**

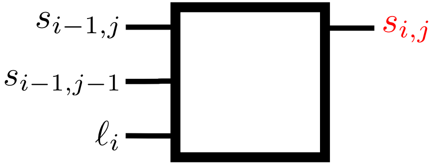
- Introduce fresh variable for each wire
- Encode using clauses describing behavior of each component

Translating PB to CNF: Step 2

2. **Encode circuit to CNF using so-called Tseitin translations**

- Introduce fresh variable for each wire
- Encode using clauses describing behavior of each component

Example: Sequential counter component



Specification of $s_{i,j}$

$$s_{i,j} \leftrightarrow (\ell_i \wedge s_{i-1,j-1}) \vee s_{i-1,j}$$

Clausal encoding

$$\bar{\ell}_i \vee \bar{s}_{i-1,j-1} \vee s_{i,j}$$
$$\bar{s}_{i-1,j} \vee s_{i,j}$$
$$\ell_i \vee s_{i-1,j} \vee \bar{s}_{i,j}$$
$$s_{i-1,j-1} \vee \bar{s}_{i,j}$$

Translating PB to CNF: Step 3

3. **Enforcing inequality**

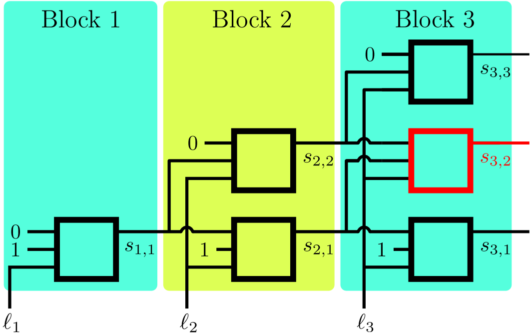
- Add clauses enforcing comparison with right-hand side

Translating PB to CNF: Step 3

3. **Enforcing inequality**

- Add clauses enforcing comparison with right-hand side

Example: $\ell_1 + \ell_2 + \ell_3 \geq 2$



At least 2 true literals if $s_{3,2}$ true.

Add unary clause

$s_{3,2}$

Our Work: Translation Correct?

Is the translation correct?

Our Work: Translation Correct?

Is the translation correct?

- **Yes!** Sinz showed that [Sin05]
- But did we implement it correctly?

Our Work: Translation Correct?

Is the translation correct?

- **Yes!** Sinz showed that [Sin05]
- But did we implement it correctly?

How can we show correctness?

- Proof logging!
- In our case: Give formal derivation of clauses using Cutting Planes + reification

Our Work: Translation Correct?

Is the translation correct?

- **Yes!** Sinz showed that [Sin05]
- But did we implement it correctly?

How can we show correctness?

- Proof logging!
- In our case: Give formal derivation of clauses using Cutting Planes + reification

This means

- Pseudo-Boolean formula satisfiable $\Longrightarrow$ CNF translation satisfiable
- Solver finds no solution to CNF translation $\Longrightarrow$ PB formula has no solution

Our Work: Translation Correct?

Is the translation correct?

- **Yes!** Sinz showed that [Sin05]
- But did we implement it correctly?

How can we show correctness?

- Proof logging!
- In our case: Give formal derivation of clauses using Cutting Planes + reification

This means

- Pseudo-Boolean formula satisfiable $\Longrightarrow$ CNF translation satisfiable
- Solver finds no solution to CNF translation $\Longrightarrow$ PB formula has no solution

End-to-end verification of SAT-based pseudo-Boolean solvers!

Rest of This Talk: How Is This Done?

We develop general framework certifying PB-to-CNF translations!

- But let us stay with our example:

Sequential counter encoding of $\ell_1 + \ell_2 + \ell_3 \geq 2$

Cutting Planes Proof System [CCT87]

Rules:

- Literal axiom

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

Cutting Planes Proof System [CCT87]

Rules:

- Literal axiom

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

- Addition

$$\text{Addition}\ \frac{x_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3 \qquad \overline{x}_2 + 3x_3 \geq 3}{x_1 + 3\overline{x}_2 + x_3 \geq 4}$$

Cutting Planes Proof System [CCT87]

Rules:

- Literal axiom

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

- Addition

$$\text{Addition}\ \frac{x_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3 \qquad \overline{x}_2 + 3x_3 \geq 3}{x_1 + 3\overline{x}_2 + x_3 \geq 4}$$

- Multiplication

$$\text{Multiply by 2}\ \frac{x_1 + 2\overline{x}_2 \geq 3}{2x_1 + 4\overline{x}_2 \geq 6}$$

Cutting Planes Proof System [CCT87]

Rules:

- Literal axiom

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

- Addition

$$\text{Addition}\ \frac{x_1 + 2\overline{x}_2 + 2\overline{x}_3 \geq 3 \qquad \overline{x}_2 + 3x_3 \geq 3}{x_1 + 3\overline{x}_2 + x_3 \geq 4}$$

- Multiplication

$$\text{Multiply by 2}\ \frac{x_1 + 2\overline{x}_2 \geq 3}{2x_1 + 4\overline{x}_2 \geq 6}$$

- Division (and rounding up)

$$\text{Divide by 2}\ \frac{2x_1 + 2\overline{x}_2 + 4x_3 \geq 5}{x_1 + \overline{x}_2 + 2x_3 \geq 3}$$

Extended Cutting Planes: Reification

Extension rule to introduce fresh variables:

- Reification (special case of redundance rule in [GN21, BGMN22])

$$a \Leftrightarrow x_1 + \overline{x}_2 + 2x_3 \geq 2 \longrightarrow \begin{array}{ll} 2\overline{a} + x_1 + \overline{x}_2 + 2x_3 \geq 2 & (a \Rightarrow x_1 + \overline{x}_2 + 2x_3 \geq 2) \\ 3a + \overline{x}_1 + x_2 + 2\overline{x}_3 \geq 3 & (a \Leftarrow x_1 + \overline{x}_2 + 2x_3 \geq 2) \end{array}$$

- Variable a was never used before in proof

Circuit Specification in Pseudo-Boolean Form

We want to derive

- Specification of $s_{i,j}$ variables

$$s_{i,j} \Leftrightarrow \sum_{k=1}^{i-1} s_{i-1,k} + \ell_i \geq j$$

- Ordering of $s_{i,j}$ variables

$$s_{i,j} \geq s_{i,j+1}$$

- Preservation of sum

$$\sum_{k=1}^{i} s_{i,k} = \sum_{k=1}^{i-1} s_{i-1,k} + \ell_i$$

Proof Logging Tseitin Variables $s_{i,j}$

- Introduction of fresh variables $s_{i,j}$ as reification

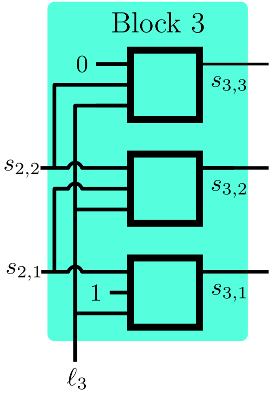
$$s_{i,j} \Leftrightarrow \sum_{k=1}^{i-1} s_{i-1,k} + \ell_i \geq j$$

Proof Logging Tseitin Variables $s_{i,j}$

- Introduction of fresh variables $s_{i,j}$ as reification

$$s_{i,j} \Leftrightarrow \sum_{k=1}^{i-1} s_{i-1,k} + \ell_i \geq j$$

- For block 3 we have:



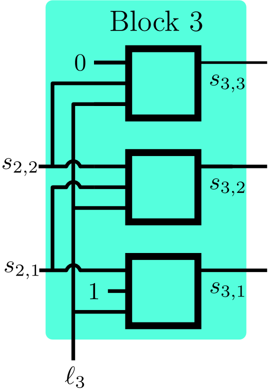
Proof Logging Tseitin Variables $s_{i,j}$

- Introduction of fresh variables $s_{i,j}$ as reification

$$s_{i,j} \Leftrightarrow \sum_{k=1}^{i-1} s_{i-1,k} + \ell_i \geq j$$

- For block 3 we have:

$$\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \quad (s_{3,1} \Rightarrow s_{2,1} + s_{2,2} + \ell_3 \geq 1)$$



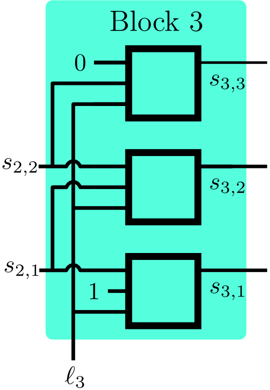
Proof Logging Tseitin Variables $s_{i,j}$

- Introduction of fresh variables $s_{i,j}$ as reification

$$s_{i,j} \Leftrightarrow \sum_{k=1}^{i-1} s_{i-1,k} + \ell_i \geq j$$

- For block 3 we have:

$$\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \quad (s_{3,1} \Rightarrow s_{2,1} + s_{2,2} + \ell_3 \geq 1)$$
$$3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3 \quad (s_{3,1} \Leftarrow s_{2,1} + s_{2,2} + \ell_3 \geq 1)$$



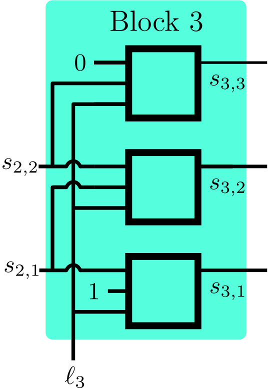
Proof Logging Tseitin Variables $s_{i,j}$

- Introduction of fresh variables $s_{i,j}$ as reification

$$s_{i,j} \Leftrightarrow \sum_{k=1}^{i-1} s_{i-1,k} + \ell_i \geq j$$

- For block 3 we have:

$$\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \quad (s_{3,1} \Rightarrow s_{2,1} + s_{2,2} + \ell_3 \geq 1)$$
$$3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3 \quad (s_{3,1} \Leftarrow s_{2,1} + s_{2,2} + \ell_3 \geq 1)$$
$$2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 \quad (s_{3,2} \Rightarrow s_{2,1} + s_{2,2} + \ell_3 \geq 2)$$



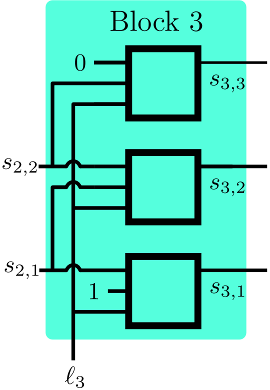
Proof Logging Tseitin Variables $s_{i,j}$

- Introduction of fresh variables $s_{i,j}$ as reification

$$s_{i,j} \Leftrightarrow \sum_{k=1}^{i-1} s_{i-1,k} + \ell_i \geq j$$

- For block 3 we have:

$$\begin{aligned}
\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 &\geq 1 \quad (s_{3,1} \Rightarrow s_{2,1} + s_{2,2} + \ell_3 \geq 1) \\
3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 &\geq 3 \quad (s_{3,1} \Leftarrow s_{2,1} + s_{2,2} + \ell_3 \geq 1) \\
2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 &\geq 2 \quad (s_{3,2} \Rightarrow s_{2,1} + s_{2,2} + \ell_3 \geq 2) \\
2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 &\geq 2 \quad (s_{3,2} \Leftarrow s_{2,1} + s_{2,2} + \ell_3 \geq 2)
\end{aligned}$$



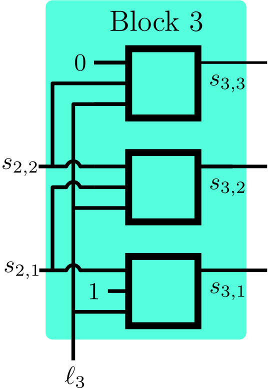
Proof Logging Tseitin Variables $s_{i,j}$

- Introduction of fresh variables $s_{i,j}$ as reification

$$s_{i,j} \Leftrightarrow \sum_{k=1}^{i-1} s_{i-1,k} + \ell_i \geq j$$

- For block 3 we have:

$$\begin{aligned}
\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 &\geq 1 \quad (s_{3,1} \Rightarrow s_{2,1} + s_{2,2} + \ell_3 \geq 1) \\
3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 &\geq 3 \quad (s_{3,1} \Leftarrow s_{2,1} + s_{2,2} + \ell_3 \geq 1) \\
2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 &\geq 2 \quad (s_{3,2} \Rightarrow s_{2,1} + s_{2,2} + \ell_3 \geq 2) \\
2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 &\geq 2 \quad (s_{3,2} \Leftarrow s_{2,1} + s_{2,2} + \ell_3 \geq 2) \\
3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 &\geq 3 \quad (s_{3,3} \Rightarrow s_{2,1} + s_{2,2} + \ell_3 \geq 3)
\end{aligned}$$



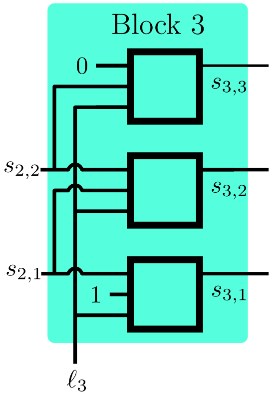
Proof Logging Tseitin Variables $s_{i,j}$

- Introduction of fresh variables $s_{i,j}$ as reification

$$s_{i,j} \Leftrightarrow \sum_{k=1}^{i-1} s_{i-1,k} + \ell_i \geq j$$

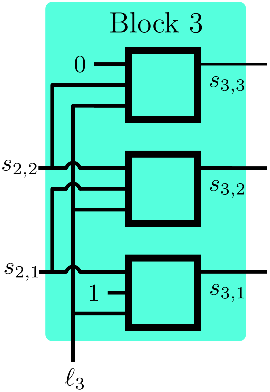
- For block 3 we have:

$$\begin{aligned}
\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 &\geq 1 \quad (s_{3,1} \Rightarrow s_{2,1} + s_{2,2} + \ell_3 \geq 1)\\
3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 &\geq 3 \quad (s_{3,1} \Leftarrow s_{2,1} + s_{2,2} + \ell_3 \geq 1)\\
2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 &\geq 2 \quad (s_{3,2} \Rightarrow s_{2,1} + s_{2,2} + \ell_3 \geq 2)\\
2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 &\geq 2 \quad (s_{3,2} \Leftarrow s_{2,1} + s_{2,2} + \ell_3 \geq 2)\\
3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 &\geq 3 \quad (s_{3,3} \Rightarrow s_{2,1} + s_{2,2} + \ell_3 \geq 3)\\
s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 &\geq 1 \quad (s_{3,3} \Leftarrow s_{2,1} + s_{2,2} + \ell_3 \geq 3)
\end{aligned}$$



Deriving Pseudo-Boolean Specification for Block 3

$$\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3$$
$$2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 \qquad 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2$$
$$3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 \qquad s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1$$

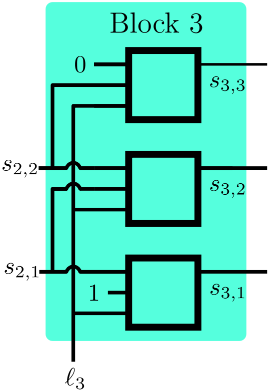


Deriving Pseudo-Boolean Specification for Block 3

$$\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3$$
$$2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 \qquad 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2$$
$$3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 \qquad s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1$$

We want to derive:

$$s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$

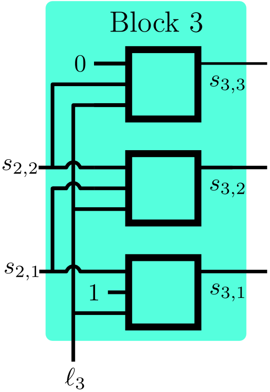


Deriving Pseudo-Boolean Specification for Block 3

$$\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3$$
$$2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 \qquad 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2$$
$$3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 \qquad s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1$$

We want to derive:

$$s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$
$$s_{3,1} \geq s_{3,2}$$
$$s_{3,2} \geq s_{3,3}$$



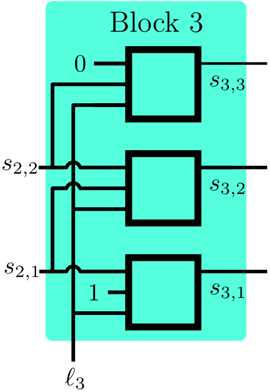
Deriving Pseudo-Boolean Specification for Block 3

$$\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3$$
$$2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 \qquad 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2$$
$$3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 \qquad s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1$$

We want to derive:

$$s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$
$$s_{3,1} \geq s_{3,2}$$
$$s_{3,2} \geq s_{3,3}$$

- Can be derived using standard Cutting Planes derivations



Derivation of Pseudo-Boolean Specification

$$
\begin{aligned}
\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 &\geq 1 & \qquad 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 &\geq 3 \\
2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 &\geq 2 & \qquad 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 &\geq 2 \\
3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 &\geq 3 & \qquad s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 &\geq 1
\end{aligned}
$$

$$s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$

Derivation of Pseudo-Boolean Specification

$$
\begin{array}{rr}
\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 & 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3 \\
2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 & 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2 \\
3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 & s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1
\end{array}
$$

$$
s_{3,1} + s_{3,2} + s_{3,3} \leq s_{2,1} + s_{2,2} + \ell_3
$$

Derivation of Pseudo-Boolean Specification

$$\begin{array}{ccc} \downarrow & \bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 & 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3 \\ & 2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 & 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2 \\ & 3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 & s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1 \end{array}$$

$$s_{3,1} + s_{3,2} + s_{3,3} \leq s_{2,1} + s_{2,2} + \ell_3$$

$$\text{Addition}\ \frac{\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2}{\bar{s}_{3,1} + 2\bar{s}_{3,2} + 2s_{2,1} + 2s_{2,2} + 2\ell_3 \geq 3}$$

Derivation of Pseudo-Boolean Specification

$$\begin{array}{ll} \bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 & 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3 \\ 2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 & 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2 \\ 3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 & s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1 \end{array}$$

$$s_{3,1} + s_{3,2} + s_{3,3} \leq s_{2,1} + s_{2,2} + \ell_3$$

$$\text{Division by 2}\ \dfrac{\text{Addition}\ \dfrac{\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2}{\bar{s}_{3,1} + 2\bar{s}_{3,2} + 2s_{2,1} + 2s_{2,2} + 2\ell_3 \geq 3}}{\bar{s}_{3,1} + \bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2}$$

Derivation of Pseudo-Boolean Specification

$$\begin{array}{ll} \bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 & 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3 \\ 2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 & 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2 \\ 3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 & s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1 \end{array}$$

$$s_{3,1} + s_{3,2} + s_{3,3} \leq s_{2,1} + s_{2,2} + \ell_3$$

$$\dfrac{\dfrac{\dfrac{\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2}{\bar{s}_{3,1} + 2\bar{s}_{3,2} + 2s_{2,1} + 2s_{2,2} + 2\ell_3 \geq 3}\text{Addition}}{\bar{s}_{3,1} + \bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2}\text{Division by 2}}{2\bar{s}_{3,1} + 2\bar{s}_{3,2} + 2s_{2,1} + 2s_{2,2} + 2\ell_3 \geq 4}\text{Multiplication by 2}$$

Derivation of Pseudo-Boolean Specification

$$\downarrow \quad \begin{array}{rr} \bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 & 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3 \\ 2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 & 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2 \\ 3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 & s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1 \end{array}$$

$$s_{3,1} + s_{3,2} + s_{3,3} \leq s_{2,1} + s_{2,2} + \ell_3$$

$$\cfrac{\cfrac{\cfrac{\cfrac{\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2}{\bar{s}_{3,1} + 2\bar{s}_{3,2} + 2s_{2,1} + 2s_{2,2} + 2\ell_3 \geq 3}\text{Addition}}{\bar{s}_{3,1} + \bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2}\text{Division by 2}}{2\bar{s}_{3,1} + 2\bar{s}_{3,2} + 2s_{2,1} + 2s_{2,2} + 2\ell_3 \geq 4}\text{Multiplication by 2} \qquad 3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3}{2\bar{s}_{3,1} + 2\bar{s}_{3,2} + 3\bar{s}_{3,3} + 3s_{2,1} + 3s_{2,2} + 3\ell_3 \geq 7}\text{Addition}$$

Derivation of Pseudo-Boolean Specification

$$\begin{array}{ll}
\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 & 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3 \\
2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 & 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2 \\
3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 & s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1
\end{array}$$

$$s_{3,1} + s_{3,2} + s_{3,3} \leq s_{2,1} + s_{2,2} + \ell_3$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2}{\bar{s}_{3,1} + 2\bar{s}_{3,2} + 2s_{2,1} + 2s_{2,2} + 2\ell_3 \geq 3}\text{Addition}}{\bar{s}_{3,1} + \bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2}\text{Division by 2}}{2\bar{s}_{3,1} + 2\bar{s}_{3,2} + 2s_{2,1} + 2s_{2,2} + 2\ell_3 \geq 4}\text{Multiplication by 2} \qquad 3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3}{\dfrac{2\bar{s}_{3,1} + 2\bar{s}_{3,2} + 3\bar{s}_{3,3} + 3s_{2,1} + 3s_{2,2} + 3\ell_3 \geq 7}{\bar{s}_{3,1} + \bar{s}_{3,2} + \bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3}\text{Division by 3}}\text{Addition}$$

Derivation of Pseudo-Boolean Specification

$$\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3$$

$$2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 \qquad 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2$$

$$3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 \qquad s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1$$

$$s_{3,1} + s_{3,2} + s_{3,3} \geq s_{2,1} + s_{2,2} + \ell_3$$

$$\dfrac{\dfrac{\dfrac{\dfrac{s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1 \qquad 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2}{s_{3,3} + 2s_{3,2} + 2\bar{s}_{2,1} + 2\bar{s}_{2,2} + 2\bar{\ell}_3 \geq 3}\text{Addition}}{s_{3,3} + s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2}\text{Division by 2}}{2s_{3,3} + 2s_{3,2} + 2\bar{s}_{2,1} + 2\bar{s}_{2,2} + 2\bar{\ell}_3 \geq 4}\text{Multiplication by 2} \qquad 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3}{\dfrac{2s_{3,3} + 2s_{3,2} + 3s_{3,1} + 3\bar{s}_{2,1} + 3\bar{s}_{2,2} + 3\bar{\ell}_3 \geq 7}{s_{3,1} + s_{3,2} + s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3}\text{Division by 3}}\text{Addition}$$

Derivation of Pseudo-Boolean Specification

$$\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3$$

$$2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 \qquad 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2$$

$$3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 \qquad s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1$$

$$s_{3,1} \geq s_{3,2}$$

Derivation of Pseudo-Boolean Specification

$$\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3$$

$$2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 \qquad 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2$$

$$3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 \qquad s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1$$

$$s_{3,1} \geq s_{3,2}$$

$$\text{Addition}\ \frac{3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3 \qquad 2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2}{3s_{3,1} + 2\bar{s}_{3,2} \geq 2}$$

Derivation of Pseudo-Boolean Specification

$$\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3$$

$$2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 \qquad 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2$$

$$3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 \qquad s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1$$

$$s_{3,1} \geq s_{3,2}$$

$$\text{Addition}\ \dfrac{3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3 \qquad 2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2}{\text{Divison by 3}\ \dfrac{3s_{3,1} + 2\bar{s}_{3,2} \geq 2}{s_{3,1} + \bar{s}_{3,2} \geq 1}}$$

Derivation of Pseudo-Boolean Specification

$$\bar{s}_{3,1} + s_{2,1} + s_{2,2} + \ell_3 \geq 1 \qquad 3s_{3,1} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 3$$

$$2\bar{s}_{3,2} + s_{2,1} + s_{2,2} + \ell_3 \geq 2 \qquad 2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2$$

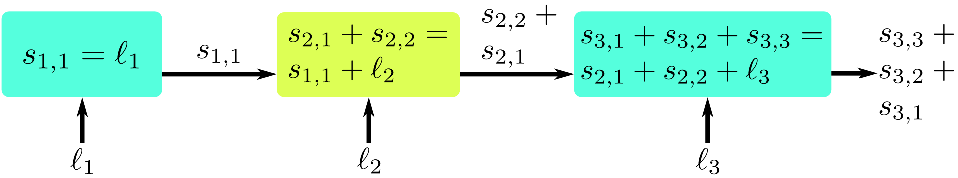
$$3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3 \qquad s_{3,3} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 1$$

$$s_{3,2} \geq s_{3,3}$$

$$\text{Divison by 3}\ \dfrac{\text{Addition}\ \dfrac{2s_{3,2} + \bar{s}_{2,1} + \bar{s}_{2,2} + \bar{\ell}_3 \geq 2 \qquad 3\bar{s}_{3,3} + s_{2,1} + s_{2,2} + \ell_3 \geq 3}{2s_{3,2} + 3\bar{s}_{3,3} \geq 2}}{s_{3,2} + \bar{s}_{3,3} \geq 1}$$

Telescoping Preservation Equality

$$s_{1,1} = \ell_1$$
$$s_{2,1} + s_{2,2} = s_{1,1} + \ell_2$$
$$s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$

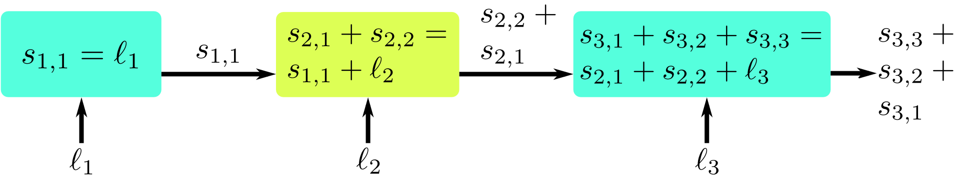


Telescoping Preservation Equality

$$\boxed{\begin{gathered} s_{1,1} = \ell_1 \\ s_{2,1} + s_{2,2} = s_{1,1} + \ell_2 \\ s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3 \end{gathered}}$$

- Sum preservation equality of blocks:

$$s_{1,1} = \ell_1$$
$$s_{2,1} + s_{2,2} = s_{1,1} + \ell_2$$

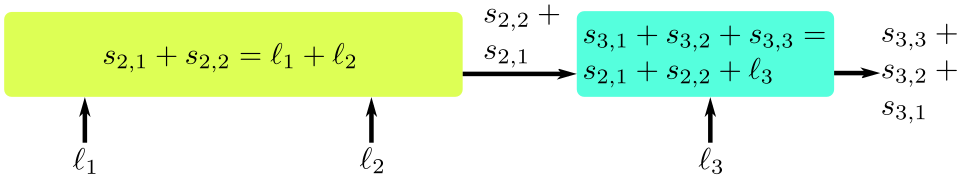


Telescoping Preservation Equality

$$
\boxed{\begin{aligned} s_{1,1} &= \ell_1 \\ s_{2,1} + s_{2,2} &= s_{1,1} + \ell_2 \\ s_{3,1} + s_{3,2} + s_{3,3} &= s_{2,1} + s_{2,2} + \ell_3 \end{aligned}}
$$

- Sum preservation equality of blocks:

$$
\begin{aligned} s_{1,1} &= \ell_1 \\ s_{2,1} + s_{2,2} &= s_{1,1} + \ell_2 \\ \Rightarrow s_{2,1} + s_{2,2} &= \ell_1 + \ell_2 \end{aligned}
$$

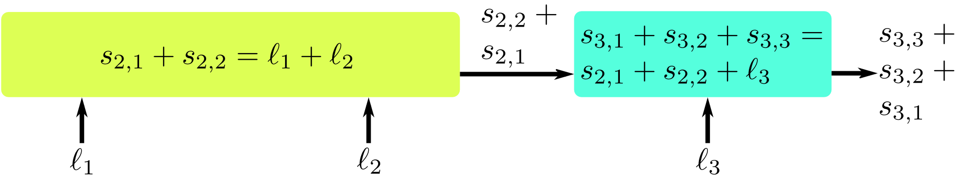


Telescoping Preservation Equality

$$\begin{aligned} s_{1,1} &= \ell_1 \\ s_{2,1} + s_{2,2} &= s_{1,1} + \ell_2 \\ s_{3,1} + s_{3,2} + s_{3,3} &= s_{2,1} + s_{2,2} + \ell_3 \end{aligned}$$

- Sum preservation equality of blocks:

$$\begin{aligned} s_{1,1} &= \ell_1 \\ s_{2,1} + s_{2,2} &= s_{1,1} + \ell_2 \\ \Rightarrow s_{2,1} + s_{2,2} &= \ell_1 + \ell_2 \\ s_{3,1} + s_{3,2} + s_{3,3} &= s_{2,1} + s_{2,2} + \ell_3 \end{aligned}$$

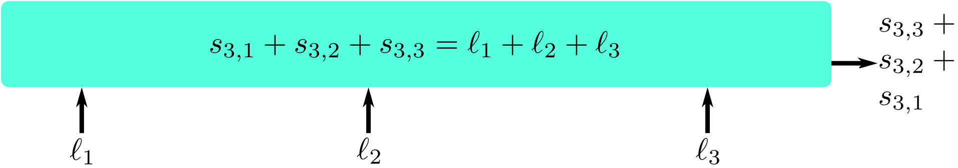


Telescoping Preservation Equality

$$\boxed{\begin{aligned} s_{1,1} &= \ell_1 \\ s_{2,1} + s_{2,2} &= s_{1,1} + \ell_2 \\ s_{3,1} + s_{3,2} + s_{3,3} &= s_{2,1} + s_{2,2} + \ell_3 \end{aligned}}$$

- Sum preservation equality of blocks:

$$\begin{aligned} s_{1,1} &= \ell_1 \\ s_{2,1} + s_{2,2} &= s_{1,1} + \ell_2 \\ \Rightarrow s_{2,1} + s_{2,2} &= \ell_1 + \ell_2 \\ s_{3,1} + s_{3,2} + s_{3,3} &= s_{2,1} + s_{2,2} + \ell_3 \\ \Rightarrow s_{3,1} + s_{3,2} + s_{3,3} &= \ell_1 + \ell_2 + \ell_3 \end{aligned}$$



Enforcing Bound on Output Variables

$$s_{3,1} + s_{3,2} + s_{3,3} = \ell_1 + \ell_2 + \ell_3$$

We want a bound on the output variables:

$\ell_1 + \ell_2 + \ell_3 \geq 2$	pseudo-Boolean input
$s_{3,1} + s_{3,2} + s_{3,3} = \ell_1 + \ell_2 + \ell_3$	just derived

Enforcing Bound on Output Variables

$$s_{3,1} + s_{3,2} + s_{3,3} = \ell_1 + \ell_2 + \ell_3$$

We want a bound on the output variables:

$\ell_1 + \ell_2 + \ell_3 \geq 2$	pseudo-Boolean input
$s_{3,1} + s_{3,2} + s_{3,3} = \ell_1 + \ell_2 + \ell_3$	just derived
$\Rightarrow s_{3,1} + s_{3,2} + s_{3,3} \geq 2$	telescoping again

Deriving the CNF Translation

We now have pseudo-Boolean constraints:

$$s_{1,1} = \ell_1 \qquad s_{2,1} + s_{2,2} = s_{1,1} + \ell_2 \qquad s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$

$$s_{2,1} \geq s_{2,2} \qquad s_{3,1} \geq s_{3,2} \qquad s_{3,2} \geq s_{3,3} \qquad s_{3,1} + s_{3,2} + s_{3,3} \geq 2$$

Deriving the CNF Translation

We now have pseudo-Boolean constraints:

$$s_{1,1} = \ell_1 \qquad s_{2,1} + s_{2,2} = s_{1,1} + \ell_2 \qquad s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$
$$s_{2,1} \geq s_{2,2} \qquad s_{3,1} \geq s_{3,2} \qquad s_{3,2} \geq s_{3,3} \qquad s_{3,1} + s_{3,2} + s_{3,3} \geq 2$$

But we want clauses:

$$\begin{array}{rrrr}
\bar{\ell}_1 \vee s_{1,1} & \bar{\ell}_2 \vee \bar{s}_{1,1} \vee s_{2,2} & \ell_3 \vee s_{2,1} \vee \bar{s}_{3,1} & \bar{\ell}_3 \vee \bar{s}_{2,2} \vee s_{3,3} \\
\ell_1 \vee \bar{s}_{1,1} & \ell_2 \vee \bar{s}_{2,2} & \bar{\ell}_3 \vee \bar{s}_{2,1} \vee s_{3,2} & \ell_3 \vee \bar{s}_{3,3} \\
\bar{\ell}_2 \vee s_{2,1} & s_{1,1} \vee \bar{s}_{2,2} & \bar{s}_{2,2} \vee s_{3,2} & s_{2,2} \vee \bar{s}_{3,3} \\
\bar{s}_{1,1} \vee s_{2,1} & \bar{\ell}_3 \vee s_{3,1} & \ell_3 \vee s_{2,2} \vee \bar{s}_{3,2} & s_{3,2} \\
\ell_2 \vee s_{1,1} \vee \bar{s}_{2,1} & \bar{s}_{2,1} \vee s_{3,1} & s_{2,1} \vee \bar{s}_{3,2} &
\end{array}$$

Deriving the CNF Translation

We now have pseudo-Boolean constraints:

$$s_{1,1} = \ell_1 \qquad s_{2,1} + s_{2,2} = s_{1,1} + \ell_2 \qquad s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$
$$s_{2,1} \geq s_{2,2} \qquad s_{3,1} \geq s_{3,2} \qquad s_{3,2} \geq s_{3,3} \qquad s_{3,1} + s_{3,2} + s_{3,3} \geq 2$$

But we want clauses:

$\bar{\ell}_1 \vee s_{1,1}$	$\bar{\ell}_2 \vee \bar{s}_{1,1} \vee s_{2,2}$	$\ell_3 \vee s_{2,1} \vee \bar{s}_{3,1}$	$\bar{\ell}_3 \vee \bar{s}_{2,2} \vee s_{3,3}$
$\ell_1 \vee \bar{s}_{1,1}$	$\ell_2 \vee \bar{s}_{2,2}$	$\bar{\ell}_3 \vee \bar{s}_{2,1} \vee s_{3,2}$	$\ell_3 \vee \bar{s}_{3,3}$
$\bar{\ell}_2 \vee s_{2,1}$	$s_{1,1} \vee \bar{s}_{2,2}$	$\bar{s}_{2,2} \vee s_{3,2}$	$s_{2,2} \vee \bar{s}_{3,3}$
$\bar{s}_{1,1} \vee s_{2,1}$	$\bar{\ell}_3 \vee s_{3,1}$	$\ell_3 \vee s_{2,2} \vee \bar{s}_{3,2}$	$s_{3,2}$
$\ell_2 \vee s_{1,1} \vee \bar{s}_{2,1}$	$\bar{s}_{2,1} \vee s_{3,1}$	$s_{2,1} \vee \bar{s}_{3,2}$	

- Clauses follow from PB specification by reverse unit propagation [GN03, Van08b]

Deriving the CNF Translation

We now have pseudo-Boolean constraints:

$$s_{1,1} = \ell_1 \qquad s_{2,1} + s_{2,2} = s_{1,1} + \ell_2 \qquad s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$

$$s_{2,1} \geq s_{2,2} \qquad s_{3,1} \geq s_{3,2} \qquad s_{3,2} \geq s_{3,3} \qquad s_{3,1} + s_{3,2} + s_{3,3} \geq 2$$

But we want clauses:

$\bar{\ell}_1 \vee s_{1,1}$	$\bar{\ell}_2 \vee \bar{s}_{1,1} \vee s_{2,2}$	$\ell_3 \vee s_{2,1} \vee \bar{s}_{3,1}$	$\bar{\ell}_3 \vee \bar{s}_{2,2} \vee s_{3,3}$
$\ell_1 \vee \bar{s}_{1,1}$	$\ell_2 \vee \bar{s}_{2,2}$	$\bar{\ell}_3 \vee \bar{s}_{2,1} \vee s_{3,2}$	$\ell_3 \vee \bar{s}_{3,3}$
$\bar{\ell}_2 \vee s_{2,1}$	$s_{1,1} \vee \bar{s}_{2,2}$	$\bar{s}_{2,2} \vee s_{3,2}$	$s_{2,2} \vee \bar{s}_{3,3}$
$\bar{s}_{1,1} \vee s_{2,1}$	$\bar{\ell}_3 \vee s_{3,1}$	$\ell_3 \vee s_{2,2} \vee \bar{s}_{3,2}$	$s_{3,2}$
$\ell_2 \vee s_{1,1} \vee \bar{s}_{2,1}$	$\bar{s}_{2,1} \vee s_{3,1}$	$s_{2,1} \vee \bar{s}_{3,2}$	

$s_{3,2} \to 0$

Deriving the CNF Translation

We now have pseudo-Boolean constraints:

$$s_{1,1} = \ell_1 \qquad s_{2,1} + s_{2,2} = s_{1,1} + \ell_2 \qquad s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$

$$s_{2,1} \geq s_{2,2} \qquad s_{3,1} \geq s_{3,2} \qquad s_{3,2} \geq s_{3,3} \qquad s_{3,1} + s_{3,2} + s_{3,3} \geq 2$$

But we want clauses:

$\bar{\ell}_1 \vee s_{1,1}$	$\bar{\ell}_2 \vee \bar{s}_{1,1} \vee s_{2,2}$	$\ell_3 \vee s_{2,1} \vee \bar{s}_{3,1}$	$\bar{\ell}_3 \vee \bar{s}_{2,2} \vee s_{3,3}$
$\ell_1 \vee \bar{s}_{1,1}$	$\ell_2 \vee \bar{s}_{2,2}$	$\bar{\ell}_3 \vee \bar{s}_{2,1} \vee s_{3,2}$	$\ell_3 \vee \bar{s}_{3,3}$
$\bar{\ell}_2 \vee s_{2,1}$	$s_{1,1} \vee \bar{s}_{2,2}$	$\bar{s}_{2,2} \vee s_{3,2}$	$s_{2,2} \vee \bar{s}_{3,3}$
$\bar{s}_{1,1} \vee s_{2,1}$	$\bar{\ell}_3 \vee s_{3,1}$	$\ell_3 \vee s_{2,2} \vee \bar{s}_{3,2}$	$s_{3,2}$
$\ell_2 \vee s_{1,1} \vee \bar{s}_{2,1}$	$\bar{s}_{2,1} \vee s_{3,1}$	$s_{2,1} \vee \bar{s}_{3,2}$	

$s_{3,2} \to 0 \quad s_{3,3} \to 0$

Deriving the CNF Translation

We now have pseudo-Boolean constraints:

$$s_{1,1} = \ell_1 \qquad s_{2,1} + s_{2,2} = s_{1,1} + \ell_2 \qquad s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$
$$s_{2,1} \geq s_{2,2} \qquad s_{3,1} \geq s_{3,2} \qquad s_{3,2} \geq s_{3,3} \qquad s_{3,1} + s_{3,2} + s_{3,3} \geq 2$$

But we want clauses:

$\bar{\ell}_1 \vee s_{1,1}$	$\bar{\ell}_2 \vee \bar{s}_{1,1} \vee s_{2,2}$	$\ell_3 \vee s_{2,1} \vee \bar{s}_{3,1}$	$\bar{\ell}_3 \vee \bar{s}_{2,2} \vee s_{3,3}$
$\ell_1 \vee \bar{s}_{1,1}$	$\ell_2 \vee \bar{s}_{2,2}$	$\bar{\ell}_3 \vee \bar{s}_{2,1} \vee s_{3,2}$	$\ell_3 \vee \bar{s}_{3,3}$
$\bar{\ell}_2 \vee s_{2,1}$	$s_{1,1} \vee \bar{s}_{2,2}$	$\bar{s}_{2,2} \vee s_{3,2}$	$s_{2,2} \vee \bar{s}_{3,3}$
$\bar{s}_{1,1} \vee s_{2,1}$	$\bar{\ell}_3 \vee s_{3,1}$	$\ell_3 \vee s_{2,2} \vee \bar{s}_{3,2}$	$s_{3,2}$
$\ell_2 \vee s_{1,1} \vee \bar{s}_{2,1}$	$\bar{s}_{2,1} \vee s_{3,1}$	$s_{2,1} \vee \bar{s}_{3,2}$	

$s_{3,2} \to 0 \quad s_{3,3} \to 0$

Deriving the CNF Translation

We now have pseudo-Boolean constraints:

$$s_{1,1} = \ell_1 \qquad s_{2,1} + s_{2,2} = s_{1,1} + \ell_2 \qquad s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$

$$s_{2,1} \geq s_{2,2} \qquad s_{3,1} \geq s_{3,2} \qquad s_{3,2} \geq s_{3,3} \qquad s_{3,1} + s_{3,2} + s_{3,3} \geq 2$$

But we want clauses:

$\bar{\ell}_1 \vee s_{1,1}$	$\bar{\ell}_2 \vee \bar{s}_{1,1} \vee s_{2,2}$	$\ell_3 \vee s_{2,1} \vee \bar{s}_{3,1}$	$\bar{\ell}_3 \vee \bar{s}_{2,2} \vee s_{3,3}$
$\ell_1 \vee \bar{s}_{1,1}$	$\ell_2 \vee \bar{s}_{2,2}$	$\bar{\ell}_3 \vee \bar{s}_{2,1} \vee s_{3,2}$	$\ell_3 \vee \bar{s}_{3,3}$
$\bar{\ell}_2 \vee s_{2,1}$	$s_{1,1} \vee \bar{s}_{2,2}$	$\bar{s}_{2,2} \vee s_{3,2}$	$s_{2,2} \vee \bar{s}_{3,3}$
$\bar{s}_{1,1} \vee s_{2,1}$	$\bar{\ell}_3 \vee s_{3,1}$	$\ell_3 \vee s_{2,2} \vee \bar{s}_{3,2}$	$s_{3,2}$
$\ell_2 \vee s_{1,1} \vee \bar{s}_{2,1}$	$\bar{s}_{2,1} \vee s_{3,1}$	$s_{2,1} \vee \bar{s}_{3,2}$	

$\ell_3 \to 0 \quad s_{2,2} \to 0 \quad s_{3,2} \to 1$

Deriving the CNF Translation

We now have pseudo-Boolean constraints:

$$s_{1,1} = \ell_1 \qquad s_{2,1} + s_{2,2} = s_{1,1} + \ell_2 \qquad s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$

$$s_{2,1} \geq s_{2,2} \qquad s_{3,1} \geq s_{3,2} \qquad s_{3,2} \geq s_{3,3} \qquad s_{3,1} + s_{3,2} + s_{3,3} \geq 2$$

But we want clauses:

$\bar{\ell}_1 \vee s_{1,1}$	$\bar{\ell}_2 \vee \bar{s}_{1,1} \vee s_{2,2}$	$\ell_3 \vee s_{2,1} \vee \bar{s}_{3,1}$	$\bar{\ell}_3 \vee \bar{s}_{2,2} \vee s_{3,3}$
$\ell_1 \vee \bar{s}_{1,1}$	$\ell_2 \vee \bar{s}_{2,2}$	$\bar{\ell}_3 \vee \bar{s}_{2,1} \vee s_{3,2}$	$\ell_3 \vee \bar{s}_{3,3}$
$\bar{\ell}_2 \vee s_{2,1}$	$s_{1,1} \vee \bar{s}_{2,2}$	$\bar{s}_{2,2} \vee s_{3,2}$	$s_{2,2} \vee \bar{s}_{3,3}$
$\bar{s}_{1,1} \vee s_{2,1}$	$\bar{\ell}_3 \vee s_{3,1}$	$\ell_3 \vee s_{2,2} \vee \bar{s}_{3,2}$	$s_{3,2}$
$\ell_2 \vee s_{1,1} \vee \bar{s}_{2,1}$	$\bar{s}_{2,1} \vee s_{3,1}$	$s_{2,1} \vee \bar{s}_{3,2}$	

$\ell_3 \to 0 \quad s_{2,2} \to 0 \quad s_{3,2} \to 1 \quad s_{3,1} \to 1$

Deriving the CNF Translation

We now have pseudo-Boolean constraints:

$$s_{1,1} = \ell_1 \qquad s_{2,1} + s_{2,2} = s_{1,1} + \ell_2 \qquad s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$

$$s_{2,1} \geq s_{2,2} \qquad s_{3,1} \geq s_{3,2} \qquad s_{3,2} \geq s_{3,3} \qquad s_{3,1} + s_{3,2} + s_{3,3} \geq 2$$

But we want clauses:

$\bar{\ell}_1 \vee s_{1,1}$	$\bar{\ell}_2 \vee \bar{s}_{1,1} \vee s_{2,2}$	$\ell_3 \vee s_{2,1} \vee \bar{s}_{3,1}$	$\bar{\ell}_3 \vee \bar{s}_{2,2} \vee s_{3,3}$
$\ell_1 \vee \bar{s}_{1,1}$	$\ell_2 \vee \bar{s}_{2,2}$	$\bar{\ell}_3 \vee \bar{s}_{2,1} \vee s_{3,2}$	$\ell_3 \vee \bar{s}_{3,3}$
$\bar{\ell}_2 \vee s_{2,1}$	$s_{1,1} \vee \bar{s}_{2,2}$	$\bar{s}_{2,2} \vee s_{3,2}$	$s_{2,2} \vee \bar{s}_{3,3}$
$\bar{s}_{1,1} \vee s_{2,1}$	$\bar{\ell}_3 \vee s_{3,1}$	$\ell_3 \vee s_{2,2} \vee \bar{s}_{3,2}$	$s_{3,2}$
$\ell_2 \vee s_{1,1} \vee \bar{s}_{2,1}$	$\bar{s}_{2,1} \vee s_{3,1}$	$s_{2,1} \vee \bar{s}_{3,2}$	

$\ell_3 \to 0 \quad s_{2,2} \to 0 \quad s_{3,2} \to 1 \quad s_{3,1} \to 1$

Deriving the CNF Translation

We now have pseudo-Boolean constraints:

$$s_{1,1} = \ell_1 \qquad s_{2,1} + s_{2,2} = s_{1,1} + \ell_2 \qquad s_{3,1} + s_{3,2} + s_{3,3} = s_{2,1} + s_{2,2} + \ell_3$$
$$s_{2,1} \geq s_{2,2} \qquad s_{3,1} \geq s_{3,2} \qquad s_{3,2} \geq s_{3,3} \qquad s_{3,1} + s_{3,2} + s_{3,3} \geq 2$$

But we want clauses:

$$\begin{array}{rrrr}
\bar{\ell}_1 \lor s_{1,1} & \bar{\ell}_2 \lor \bar{s}_{1,1} \lor s_{2,2} & \ell_3 \lor s_{2,1} \lor \bar{s}_{3,1} & \bar{\ell}_3 \lor \bar{s}_{2,2} \lor s_{3,3} \\
\ell_1 \lor \bar{s}_{1,1} & \ell_2 \lor \bar{s}_{2,2} & \bar{\ell}_3 \lor \bar{s}_{2,1} \lor s_{3,2} & \ell_3 \lor \bar{s}_{3,3} \\
\bar{\ell}_2 \lor s_{2,1} & s_{1,1} \lor \bar{s}_{2,2} & \bar{s}_{2,2} \lor s_{3,2} & s_{2,2} \lor \bar{s}_{3,3} \\
\bar{s}_{1,1} \lor s_{2,1} & \bar{\ell}_3 \lor s_{3,1} & \ell_3 \lor s_{2,2} \lor \bar{s}_{3,2} & s_{3,2} \\
\ell_2 \lor s_{1,1} \lor \bar{s}_{2,1} & \bar{s}_{2,1} \lor s_{3,1} & s_{2,1} \lor \bar{s}_{3,2} &
\end{array}$$

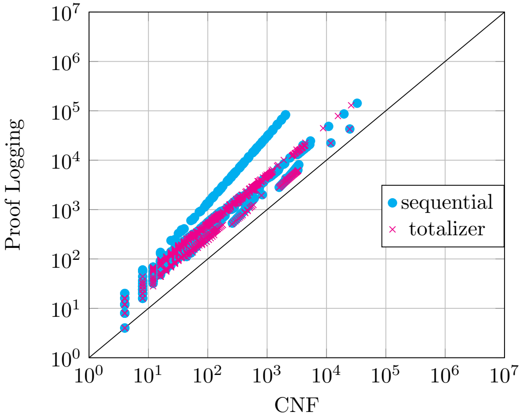
- Clauses follow from PB specification by reverse unit propagation [GN03, Van08b]
- See SAT'22 paper for detailed explanation [GMNO22]

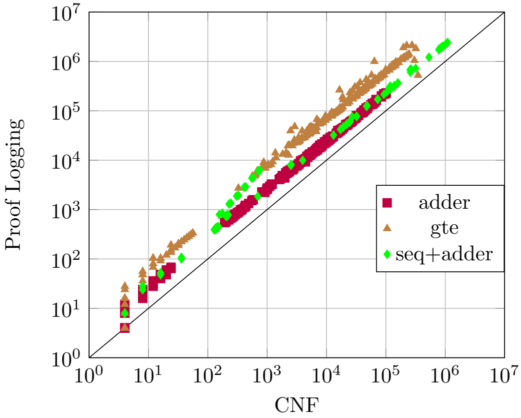
Experiments

- Implemented CNF translations with proofs of correctness[1] for
  - Sequential counter [Sin05]
  - Totalizer [BB03]
  - Generalized totalizer [JMM15]
  - Adder network [ES06]
- Proof verified by proof checker VeriPB[2]
- Benchmarks from PB 2016 Evaluation[3]
  - `SMALLINT` decision benchmarks without purely clausal formulas
  - 3 subclasses of benchmarks:
    - Only cardinality constraints (sequential counter, totalizer)
    - Only general PB constraints (generalized totalizer, adder network)
    - Mixed cardinality & general PB constraints (sequential counter + adder network)

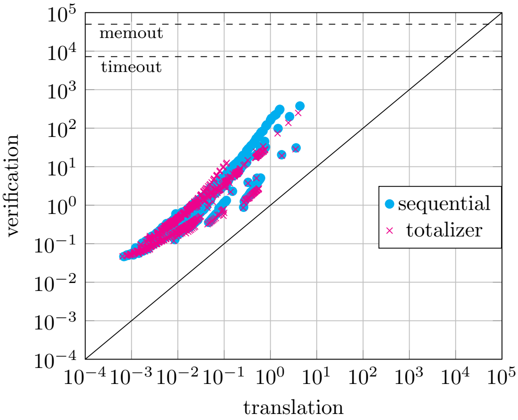
[1] `https://github.com/forge-lab/VeritasPBLib`
[2] `https://gitlab.com/MIAOresearch/software/VeriPB`
[3] `http://www.cril.univ-artois.fr/PB16/`

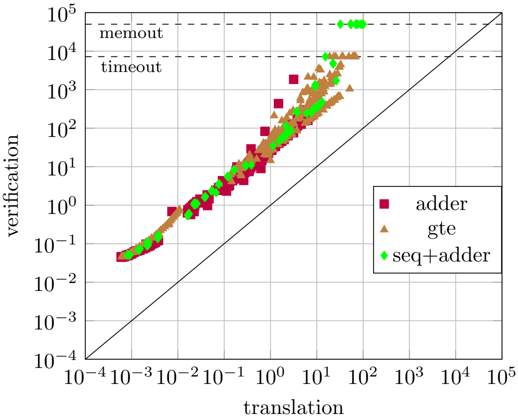
CNF Size vs Proof Size in KiB





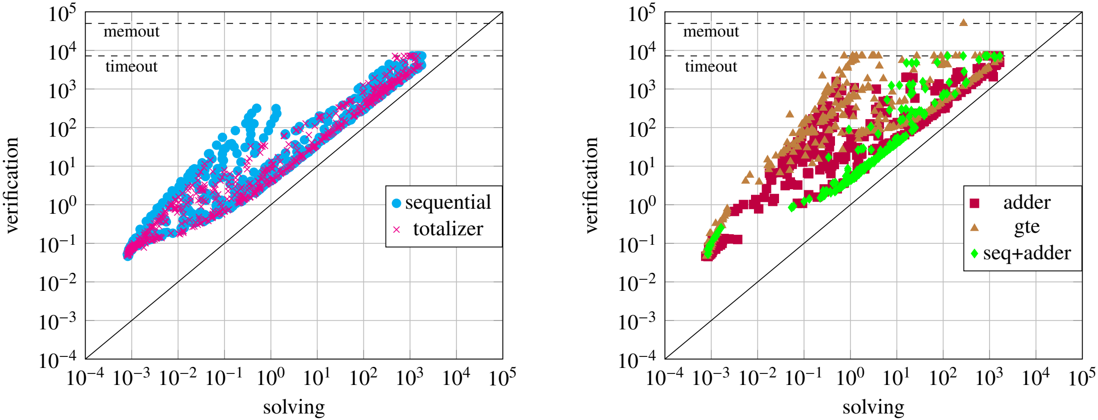
Translation Time vs Verification Time in Seconds





- Translation just generates clauses and proof
- Verification slower, as reasoning has to be performed

Solving Time vs Verification Time in Seconds



- Solved with fork of Kissat[4] syntactically modified to output pseudo-Boolean proofs

[4] `https://gitlab.com/MIAOresearch/kissat_fork`

Future Work

Equivalence:

- This work: Prove CNF translation is **implied** by pseudo-Boolean formula
- Stronger statement: Prove **equivalence** of CNF translation and PB formula

Future Work

Equivalence:

- This work: Prove CNF translation is **implied** by pseudo-Boolean formula
- Stronger statement: Prove **equivalence** of CNF translation and PB formula

Improving performance:

- Cutting Planes derivations instead of reverse unit propagations [VWB22]
- Backwards checking/trimming for verification (as in DRAT-trim [HHW13a])

Future Work

Equivalence:

- This work: Prove CNF translation is **implied** by pseudo-Boolean formula
- Stronger statement: Prove **equivalence** of CNF translation and PB formula

Improving performance:

- Cutting Planes derivations instead of reverse unit propagations [VWB22]
- Backwards checking/trimming for verification (as in DRAT-trim [HHW13a])

Extend proof logging further:

- Sorting networks like odd-even mergesort, bitonic sorter [Bat68]
- MaxSAT solving and pseudo-Boolean optimization

Conclusion

This work:

- General approach for certifying different PB-to-CNF translations
- End-to-end verification of SAT-based pseudo-Boolean solving

Conclusion

This work:

- General approach for certifying different PB-to-CNF translations
- End-to-end verification of SAT-based pseudo-Boolean solving

Pseudo-Boolean reasoning provides unified proof logging method for:

- SAT solving (including advanced techniques) [GN21, BGMN22]
- (Basic) constraint programming [EGMN20, GMN22]
- Subgraph problems [GMN20, GMM+20]
- **This work:** SAT-based pseudo-Boolean solving

Conclusion

This work:

- General approach for certifying different PB-to-CNF translations
- End-to-end verification of SAT-based pseudo-Boolean solving

Pseudo-Boolean reasoning provides unified proof logging method for:

- SAT solving (including advanced techniques) [GN21, BGMN22]
- (Basic) constraint programming [EGMN20, GMN22]
- Subgraph problems [GMN20, GMM+20]
- **This work:** SAT-based pseudo-Boolean solving
- **And soon(?):** MaxSAT solving and pseudo-Boolean optimization

Conclusion

This work:

- General approach for certifying different PB-to-CNF translations
- End-to-end verification of SAT-based pseudo-Boolean solving

Pseudo-Boolean reasoning provides unified proof logging method for:

- SAT solving (including advanced techniques) [GN21, BGMN22]
- (Basic) constraint programming [EGMN20, GMN22]
- Subgraph problems [GMN20, GMM+20]
- **This work:** SAT-based pseudo-Boolean solving
- **And soon(?):** MaxSAT solving and pseudo-Boolean optimization

Thank you for your attention!

References I

[AGJ+18] Özgür Akgün, Ian P. Gent, Christopher Jefferson, Ian Miguel, and Peter Nightingale.
Metamorphic testing of constraint solvers.
In *Proceedings of the 24th International Conference on Principles and Practice of Constraint Programming (CP '18)*, volume 11008 of *Lecture Notes in Computer Science*, pages 727–736. Springer, August 2018.

[AW13] Tobias Achterberg and Roland Wunderling.
Mixed integer programming: Analyzing 12 years of progress.
In Michael Jünger and Gerhard Reinelt, editors, *Facets of Combinatorial Optimization*, pages 449–481. Springer, 2013.

[Bat68] Kenneth E. Batcher.
Sorting networks and their applications.
In *Proceedings of the Spring Joint Computer Conference of the American Federation of Information Processing Societies (AFIPS '68)*, volume 32, pages 307–314, April 1968.

References II

[BB03] Olivier Bailleux and Yacine Boufkhad.
Efficient CNF encoding of Boolean cardinality constraints.
In *Proceedings of the 9th International Conference on Principles and Practice of Constraint Programming (CP '03)*, volume 2833 of *Lecture Notes in Computer Science*, pages 108–122. Springer, September 2003.

[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.

[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.

[BLB10] Robert Brummayer, Florian Lonsing, and Armin Biere.
Automated testing and debugging of SAT and QBF solvers.
In *Proceedings of the 13th International Conference on Theory and Applications of Satisfiability Testing (SAT '10)*, volume 6175 of *Lecture Notes in Computer Science*, pages 44–57. Springer, July 2010.

References III

[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.

[CKSW13] William Cook, Thorsten Koch, Daniel E. Steffy, and Kati Wolter.
A hybrid branch-and-bound approach for exact rational mixed-integer programming.
Mathematical Programming Computation, 5(3):305–344, September 2013.

[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.

[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.

References IV

[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.

[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.

[GMN20] 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.

References V

[GMN22] Stephan Gocht, Ciaran McCreesh, and Jakob Nordström.
An auditable constraint programming solver.
In *Proceedings of the 28th International Conference on Principles and Practice of Constraint Programming (CP '22)*, volume 235 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 25:1–25:18, August 2022.

[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.

[GN03] Evgueni Goldberg and Yakov Novikov.
Verification of proofs of unsatisfiability for CNF formulas.
In *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '03)*, pages 886–891, March 2003.

References VI

[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.

[GS19] Graeme Gange and Peter Stuckey.
Certifying optimality in constraint programming.
Presentation at KTH Royal Institute of Technology. Slides available at `https://www.kth.se/polopoly_fs/1.879851.1550484700!/CertifiedCP.pdf`, February 2019.

[GSD19] Xavier Gillard, Pierre Schaus, and Yves Deville.
SolverCheck: Declarative testing of constraints.
In *Proceedings of the 25th International Conference on Principles and Practice of Constraint Programming (CP '19)*, volume 11802 of *Lecture Notes in Computer Science*, pages 565–582. Springer, October 2019.

[HHW13a] Marijn J. H. Heule, Warren A. Hunt Jr., and Nathan Wetzler.
Trimming while checking clausal proofs.
In *Proceedings of the 13th International Conference on Formal Methods in Computer-Aided Design (FMCAD '13)*, pages 181–188, October 2013.

References VII

[HHW13b] Marijn J. H. Heule, Warren A. Hunt Jr., and Nathan Wetzler.
Verifying refutations with extended resolution.
In *Proceedings of the 24th International Conference on Automated Deduction (CADE-24)*, volume 7898 of *Lecture Notes in Computer Science*, pages 345–359. Springer, June 2013.

[JMM15] Saurabh Joshi, Ruben Martins, and Vasco M. Manquinho.
Generalized totalizer encoding for pseudo-Boolean constraints.
In *Proceedings of the 21st International Conference on Principles and Practice of Constraint Programming (CP '15)*, volume 9255 of *Lecture Notes in Computer Science*, pages 200–209. Springer, August-September 2015.

[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 VIII

[RvBW06] Francesca Rossi, Peter van Beek, and Toby Walsh, editors.
Handbook of Constraint Programming, volume 2 of *Foundations of Artificial Intelligence*.
Elsevier, 2006.

[Sin05] Carsten Sinz.
Towards an optimal CNF encoding of Boolean cardinality constraints.
In *Proceedings of the 11th International Conference on Principles and Practice of Constraint Programming (CP '05)*, volume 3709 of *Lecture Notes in Computer Science*, pages 827–831. Springer, October 2005.

[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.

[Van08a] Allen Van Gelder.
Verifying RUP proofs of propositional unsatisfiability.
In *10th International Symposium on Artificial Intelligence and Mathematics (ISAIM '08)*, 2008. Available at `http://isaim2008.unl.edu/index.php?page=proceedings`.

References IX

[Van08b] Allen Van Gelder.
Verifying RUP proofs of propositional unsatisfiability.
In *10th International Symposium on Artificial Intelligence and Mathematics (ISAIM)*, 2008.
http://isaim2008.unl.edu/index.php?page=proceedings.

[VWB22] Dieter Vandesande, Wolf De Wulf, and Bart Bogaerts.
QMaxSATpb: A certified MaxSAT solver.
In *Logic Programming and Non-Monotonic Reasoning*, 2022.
To appear.

[WHH14] Nathan Wetzler, Marijn J. H. Heule, and Warren A. Hunt Jr.
DRAT-trim: Efficient checking and trimming using expressive clausal proofs.
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 422–429. Springer, July 2014.