Certified symmetry breaking with VeriPB

Bart Bogaerts
(Thanks to several co-conspirators)
Vrije Universiteit Brussel

WHOOPS'24; 23/05/2023

ARTIFICIAL
INTELLIGENCE
RESEARCH GROUP

INTRODUCTION

- I will assume familiarity with notions such as *literals*, *formulas*, SAT Solving, *CDCL*, ...
- I also assume everyone is convinced of the benefits of proof logging

INTRODUCTION

- I will assume familiarity with notions such as *literals*, *formulas*, SAT Solving, *CDCL*, ...
- I also assume everyone is convinced of the benefits of proof logging
- I will focus on symmetry handling:
  A permutation σ of literals is a (syntactic) symmetry of a formula F if:
  - σ respects negation: $\overline{\sigma(x)} = \sigma(\overline{x})$
  - $F\upharpoonright_\sigma = F$
    ($F\upharpoonright_\sigma$ is replacing each x by $\sigma(x)$ in F)

INTRODUCTION

- I will assume familiarity with notions such as *literals*, *formulas*, SAT Solving, *CDCL*, ...
- I also assume everyone is convinced of the benefits of proof logging
- I will focus on symmetry handling:
  A permutation σ of literals is a (syntactic) symmetry of a formula F if:
  - σ respects negation: $\overline{\sigma(x)} = \sigma(\overline{x})$
  - $F\restriction_\sigma = F$
    ($F\restriction_\sigma$ is replacing each x by $\sigma(x)$ in F)

Example

Consider the formula F:

$$a \lor \overline{b} \lor x \lor y \qquad b \lor c \lor x \lor y$$

$$\overline{c} \lor \overline{d} \lor x \lor y \qquad d \lor \overline{a} \lor x \lor y$$

The permutation

$$(ab\overline{c}d)(xy)(\overline{a}\overline{b}c\overline{d})(\overline{x}\overline{y})$$

INTRODUCTION

- I will assume familiarity with notions such as *literals*, *formulas*, SAT Solving, *CDCL*, ...
- I also assume everyone is convinced of the benefits of proof logging
- I will focus on symmetry handling:
  A permutation σ of literals is a (syntactic) symmetry of a formula F if:
  - σ respects negation: $\overline{\sigma(x)} = \sigma(\overline{x})$
  - $F\restriction_\sigma = F$
    ($F\restriction_\sigma$ is replacing each x by $\sigma(x)$ in F)

Example

Consider the formula F:

$$a \vee \overline{b} \vee x \vee y \qquad b \vee c \vee x \vee y$$
$$\overline{c} \vee \overline{d} \vee x \vee y \qquad d \vee \overline{a} \vee x \vee y$$

The permutation

$$(ab\overline{c}d)(xy)(\overline{a}\overline{b}c\overline{d})(\overline{x}\overline{y})$$

is a symmetry of F since $F\restriction_\sigma$ is

$$b \vee c \vee y \vee x \qquad \overline{c} \vee \overline{d} \vee y \vee x$$
$$d \vee \overline{a} \vee y \vee x \qquad a \vee \overline{b} \vee y \vee x$$

INTRODUCTION

- I will assume familiarity with notions such as *literals*, *formulas*, SAT Solving, *CDCL*, ...
- I also assume everyone is convinced of the benefits of proof logging
- I will focus on symmetry handling:
  A permutation σ of literals is a (syntactic) symmetry of a formula F if:
  - σ respects negation: $\overline{\sigma(x)} = \sigma(\overline{x})$
  - $F\upharpoonright_\sigma = F$
    ($F\upharpoonright_\sigma$ is replacing each x by $\sigma(x)$ in F)
- Symmetric problems are often problematic for vanilla CDCL solvers (insert obligatory reference to PH principle here)

Example

Consider the formula F:

$$a \vee \overline{b} \vee x \vee y \qquad b \vee c \vee x \vee y$$
$$\overline{c} \vee \overline{d} \vee x \vee y \qquad d \vee \overline{a} \vee x \vee y$$

The permutation

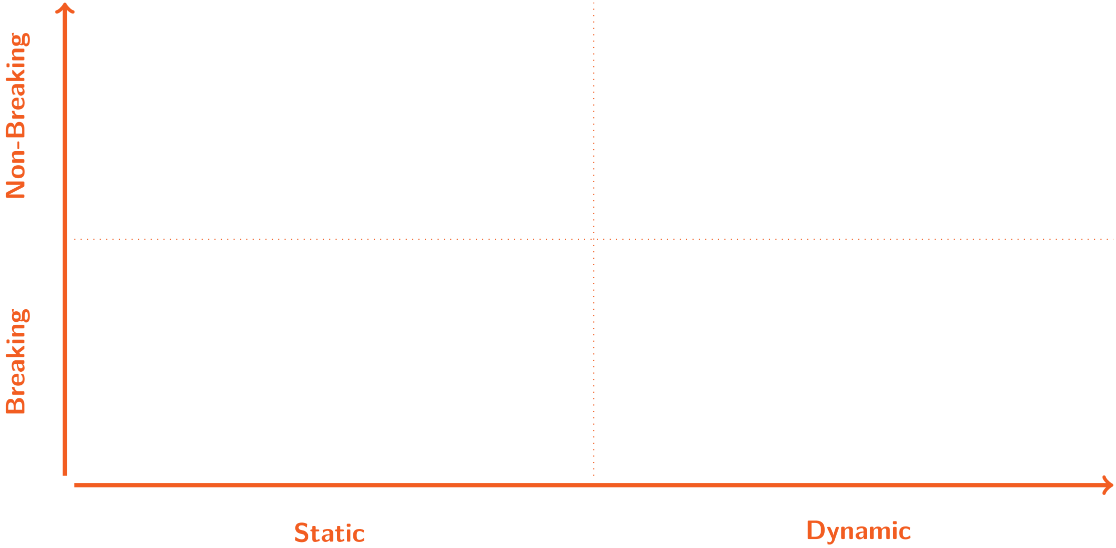
$$(ab\overline{c}d)(xy)(\overline{a}\overline{b}c\overline{d})(\overline{x}\overline{y})$$

is a symmetry of F since $F\upharpoonright_\sigma$ is

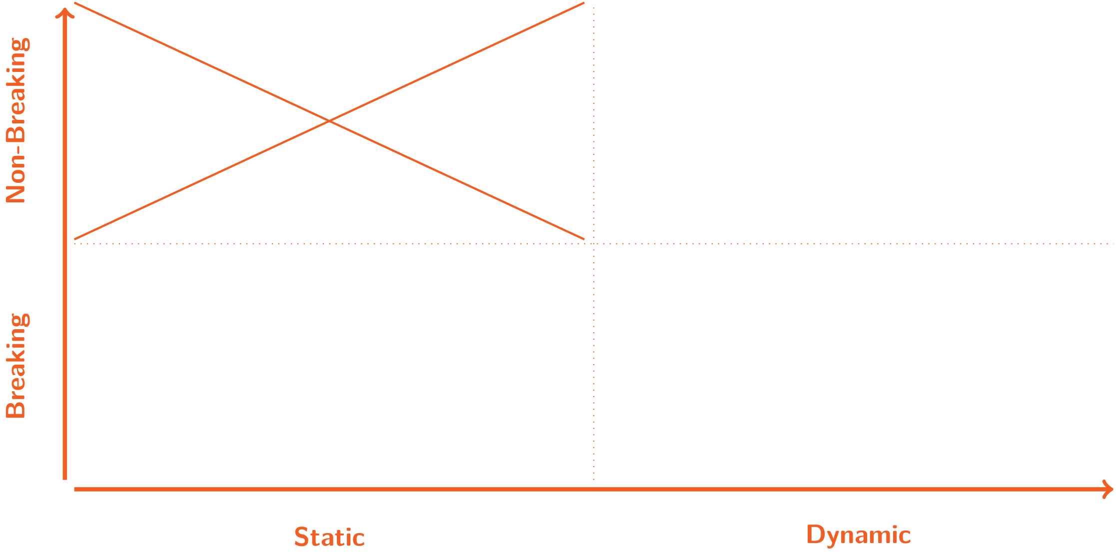
$$b \vee c \vee y \vee x \qquad \overline{c} \vee \overline{d} \vee y \vee x$$
$$d \vee \overline{a} \vee y \vee x \qquad a \vee \overline{b} \vee y \vee x$$

OUTLINE OF THIS TALK

SYMMETRY HANDLING TECHNIQUES



SYMMETRY HANDLING TECHNIQUES



SYMMETRY HANDLING TECHNIQUES

Non-Breaking

Add lex-leader constraint
for symmetries of F:

set of clauses B such that
$\alpha \models B$ iff $\alpha \preceq_{lex} \alpha \circ \sigma$

Breaking

Global symmetry breaking
SHATTER [ASM06]
BREAKID [DBBD16]

Static **Dynamic**

SYMMETRY HANDLING TECHNIQUES

	Static	Dynamic
Non-Breaking	Add lex-leader constraint for symmetries of subformulas of F:	
Breaking	Global symmetry breaking SHATTER [ASM06] BREAKID [DBBD16]	Local symmetry breaking [BS07]

SYMMETRY HANDLING TECHNIQUES

Non-Breaking

For "simple symmetries",
instead of branching on variables,
on the number of variables that are true

Breaking

Global symmetry breaking
SHATTER [ASM06]
BREAKID [DBBD16]

Local symmetry breaking [BS07]
Asymmetric branching SYMCHAFF [Sab09]

Static **Dynamic**

SYMMETRY HANDLING TECHNIQUES

Non-Breaking

Add lex-leader constraint
for symmetries of F

when these clauses would propagate

Breaking

Global symmetry breaking
SHATTER [ASM06]
BREAKID [DBBD16]

Local symmetry breaking [BS07]
Asymmetric branching SYMCHAFF [Sab09]
Effective symmetry breaking [MBCK18]

Static | Dynamic

SYMMETRY HANDLING TECHNIQUES

Non-Breaking

Propagator for $\preceq_{lex}$-minimality

Breaking

Global symmetry breaking
SHATTER [ASM06]
BREAKID [DBBD16]

Local symmetry breaking [BS07]
Asymmetric branching SYMCHAFF [Sab09]
Effective symmetry breaking [MBCK18]
SAT modulo symmetries [KS21]
SAT modulo CAS [BKG19]

Static **Dynamic**

SYMMETRY HANDLING TECHNIQUES

Non-Breaking

When SAT solver learns c,
also learn $c{\upharpoonright}_\sigma$ (if this seems "interesting")

Symmetric Learning [HKM+05]
[SHvM09, BNOS10, DBD+12, DBB17]

Breaking

Global symmetry breaking
SHATTER [ASM06]
BREAKID [DBBD16]

Local symmetry breaking [BS07]
Asymmetric branching SYMCHAFF [Sab09]
Effective symmetry breaking [MBCK18]
SAT modulo symmetries [KS21]
SAT modulo CAS [BKG19]

Static | **Dynamic**

SYMMETRY HANDLING TECHNIQUES

	Static	Dynamic
Non-Breaking	Hybrid combination of Effective symmetry breaking predicates (first) and symmetric learning (for symmetries not broken completely)	Symmetric Learning [HKM$^+$05] [SHvM09, BNOS10, DBD$^+$12, DBB17]
		ESBP+SP [MBK19]
Breaking	Global symmetry breaking SHATTER [ASM06] BREAKID [DBBD16]	Local symmetry breaking [BS07] Asymmetric branching SYMCHAFF [Sab09] Effective symmetry breaking [MBCK18] SAT modulo symmetries [KS21] SAT modulo CAS [BKG19]

CERTIFIED SYMMETRY HANDLING

Static Symmetry breaking

- DRAT proof logging for limited cases only [HHW15]

CERTIFIED SYMMETRY HANDLING

Static Symmetry breaking

- DRAT proof logging for limited cases only [HHW15]
- VeriPB proof logging for general case [BGMN22]

CERTIFIED SYMMETRY HANDLING

Static Symmetry breaking

- DRAT proof logging for limited cases only [HHW15]
- VeriPB proof logging for general case [BGMN22]
- Also appears to be applicable to dynamic symmetry *breaking*

CERTIFIED SYMMETRY HANDLING

Static Symmetry breaking

- DRAT proof logging for limited cases only [HHW15]
- VeriPB proof logging for general case [BGMN22]
- Also appears to be applicable to dynamic symmetry *breaking*

Symmetric learning

- Recently proposed proof logging [TD20]
  1. Special-purpose, specific approach
  2. Requires adding explicit concept of symmetries
  3. Not compatible with preprocessing techniques

  Better to keep proof system super-simple(?)

THE VERIPB PROOF SYSTEM

A proof system for **pseudo-Boolean optimization problems**

- Reasons with general pseudo-Boolean constraints
- Builds on cutting planes
- Extends this with strengthening rules (natural generalizations of RAT/PR)

THE VERIPB PROOF SYSTEM

A proof system for **pseudo-Boolean optimization problems**

- Reasons with general pseudo-Boolean constraints
- Builds on cutting planes
- Extends this with strengthening rules (natural generalizations of RAT/PR)

Details about the proof checker, see Stephan Gocht's PhD thesis [Goc22]

PSEUDO-BOOLEAN CONSTRAINTS

Pseudo-Boolean constraints are 0-1 integer linear constraints

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

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

PSEUDO-BOOLEAN REASONING: CUTTING PLANES [CCT87]

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

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

Division $$\frac{\sum_i c a_i \ell_i \geq A}{\sum_i a_i \ell_i \geq \lceil A/c \rceil} \quad [c \in \mathbb{N}^+]$$

STRENGTHENING RULES

When is it allowed to derive a new constraint? If it is (clear that it is) implied?

STRENGTHENING RULES

When is it allowed to derive a new constraint? If it is (clear that it is) implied?

Sometimes weaker criterion needed — recall that to get variable a encoding

$$a \leftrightarrow (3x + 2y + z + w \geq 3)$$

we introduced pseudo-Boolean constraints

$$3\overline{a} + 3x + 2y + z + w \geq 3 \qquad 5a + 3\overline{x} + 2\overline{y} + \overline{z} + \overline{w} \geq 5$$

Cutting planes method inherently cannot certify such constraints — they are not implied!

STRENGTHENING RULES

When is it allowed to derive a new constraint? If it is (clear that it is) implied?

Sometimes weaker criterion needed — recall that to get variable a encoding

$$a \leftrightarrow (3x + 2y + z + w \geq 3)$$

we introduced pseudo-Boolean constraints

$$3\overline{a} + 3x + 2y + z + w \geq 3 \qquad 5a + 3\overline{x} + 2\overline{y} + \overline{z} + \overline{w} \geq 5$$

Cutting planes method inherently cannot certify such constraints — they are not implied!

Wish to allow without-loss-of-generality arguments that can derive non-implied constraints

REDUNDANCE-BASED STRENGTHENING

C is redundant with respect to F if F and $F \wedge C$ are equisatisfiable

Adding redundant constraints should be OK

REDUNDANCE-BASED STRENGTHENING

C is redundant with respect to F if F and $F \wedge C$ are equisatisfiable

Adding redundant constraints should be OK

Redundance-based strengthening [BT19, GN21]

C is redundant with respect to F iff there is a substitution ω (mapping variables to truth values or literals), called a witness, for which

$$F \wedge \neg C \models (F \wedge C)\restriction_{\omega}$$

REDUNDANCE-BASED STRENGTHENING

Fact

α satisfies $\phi\restriction_\omega$ iff $\alpha \circ \omega$ satisfies ϕ

C is redundant with respect to F if F and $F \wedge C$ are equisatisfiable

Adding redundant constraints should be OK

Redundance-based strengthening [BT19, GN21]

C is redundant with respect to F iff there is a substitution ω (mapping variables to truth values or literals), called a witness, for which

$$F \wedge \neg C \models (F \wedge C)\restriction_\omega$$

REDUNDANCE-BASED STRENGTHENING

Fact

α satisfies $\phi\restriction_\omega$ iff $\alpha \circ \omega$ satisfies ϕ

C is redundant with respect to F if F and $F \wedge C$ are equisatisfiable

Adding redundant constraints should be OK

Redundance-based strengthening [BT19, GN21]

C is redundant with respect to F iff there is a substitution ω (mapping variables to truth values or literals), called a witness, for which

$$F \wedge \neg C \models (F \wedge C)\restriction_\omega$$

Proof sketch for interesting direction: If α satisfies F but falsifies C, then $\alpha \circ \omega$ satisfies $F \wedge C$

REDUNDANCE-BASED STRENGTHENING

C is redundant with respect to F if F and $F \wedge C$ are equisatisfiable

Adding redundant constraints should be OK

Fact

α satisfies $\phi\restriction_\omega$ iff $\alpha \circ \omega$ satisfies ϕ

Redundance-based strengthening [BT19, GN21]

C is redundant with respect to F iff there is a substitution ω (mapping variables to truth values or literals), called a witness, for which

$$F \wedge \neg C \models (F \wedge C)\restriction_\omega$$

Proof sketch for interesting direction: If α satisfies F but falsifies C, then $\alpha \circ \omega$ satisfies $F \wedge C$

Witness ω should be specified, and implication should be efficiently verifiable, which is the case for constraints in $(F \wedge C)\restriction_\omega$ that are, e.g.,

- Reverse unit propagation (RUP) constraints w.r.t. $F \wedge \neg C$
- Obviously implied by a single constraint among $F \wedge \neg C$

DERIVING $A \leftrightarrow (3X + 2Y + Z + W \geq 3)$ USING THE REDUNDANCE RULE

Want to derive

$$3\overline{a} + 3x + 2y + z + w \geq 3 \qquad 5a + 3\overline{x} + 2\overline{y} + \overline{z} + \overline{w} \geq 5$$

using condition $F \land \neg C \models (F \land C)\restriction_\omega$

DERIVING $A \leftrightarrow (3X + 2Y + Z + W \geq 3)$ USING THE REDUNDANCE RULE

Want to derive

$$3\overline{a} + 3x + 2y + z + w \geq 3 \qquad 5a + 3\overline{x} + 2\overline{y} + \overline{z} + \overline{w} \geq 5$$

using condition $F \wedge \neg C \models (F \wedge C)\restriction_\omega$

1. $F \wedge \neg(3\overline{a} + 3x + 2y + z + w \geq 3) \models (F \wedge (3\overline{a} + 3x + 2y + z + w \geq 3))\restriction_\omega$
   Choose $\omega = \{a \mapsto 0\}$ — F untouched; new constraint satisfied

DERIVING $A \leftrightarrow (3X + 2Y + Z + W \geq 3)$ USING THE REDUNDANCE RULE

Want to derive

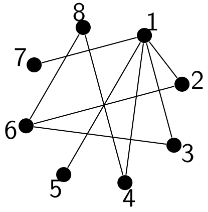
$$3\overline{a} + 3x + 2y + z + w \geq 3 \qquad 5a + 3\overline{x} + 2\overline{y} + \overline{z} + \overline{w} \geq 5$$

using condition $F \wedge \neg C \models (F \wedge C)\restriction_\omega$

1. $F \wedge \neg(3\overline{a} + 3x + 2y + z + w \geq 3) \models (F \wedge (3\overline{a} + 3x + 2y + z + w \geq 3))\restriction_\omega$
   Choose $\omega = \{a \mapsto 0\}$ — F untouched; new constraint satisfied
2. $F \wedge (3\overline{a} + 3x + 2y + z + w \geq 3) \wedge \neg(5a + 3\overline{x} + 2\overline{y} + \overline{z} + \overline{w} \geq 5) \models$
   $(F \wedge (3\overline{a} + 3x + 2y + z + w \geq 3) \wedge (5a + 3\overline{x} + 2\overline{y} + \overline{z} + \overline{w} \geq 5))\restriction_\omega$
   Choose $\omega = \{a \mapsto 1\}$ — F untouched; new constraint satisfied
   $\neg(5a + 3\overline{x} + 2\overline{y} + \overline{z} + \overline{w} \geq 5)$ forces $3\overline{x} + 2\overline{y} + \overline{z} + \overline{w} \leq 4$
   This is the same constraint as $3x + 2y + z + w \geq 3$
   And VeriPB can automatically detect this

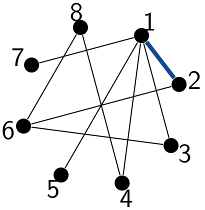
TOY EXAMPLE OF REDUNDANCE RULE

Is the following graph 2-colourable (bipartite)?



TOY EXAMPLE OF REDUNDANCE RULE

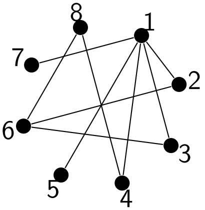
Is the following graph 2-colourable (bipartite)?



$$b_1 + b_2 \geq 1$$
$$\bar{b}_1 + \bar{b}_2 \geq 1$$

TOY EXAMPLE OF REDUNDANCE RULE

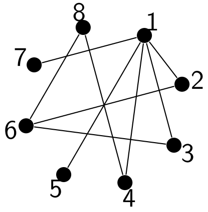
Is the following graph 2-colourable (bipartite)?



$$b_1 + b_2 \geq 1$$
$$\bar{b}_1 + \bar{b}_2 \geq 1$$
$$b_1 + b_3 \geq 1$$
$$\bar{b}_1 + \bar{b}_3 \geq 1$$
$$\ldots$$

TOY EXAMPLE OF REDUNDANCE RULE

Is the following graph 2-colourable (bipartite)?

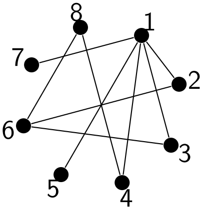


$$b_1 + b_2 \geq 1$$
$$\bar{b}_1 + \bar{b}_2 \geq 1$$
$$b_1 + b_3 \geq 1$$
$$\bar{b}_1 + \bar{b}_3 \geq 1$$
$$\dots$$

Without loss of generality, node 1 is blue (otherwise we can swap the two colours)

TOY EXAMPLE OF REDUNDANCE RULE

Is the following graph 2-colourable (bipartite)?



$$b_1 + b_2 \geq 1$$
$$\bar{b}_1 + \bar{b}_2 \geq 1$$
$$b_1 + b_3 \geq 1$$
$$\bar{b}_1 + \bar{b}_3 \geq 1$$
$$\ldots$$

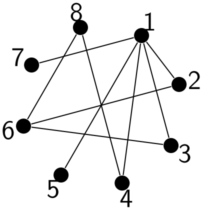
Without loss of generality, node 1 is blue (otherwise we can swap the two colours)

Derive $b_1 \geq 1$ with redundance; witness $\omega : b_i \mapsto \bar{b}_i$

$$F = F\restriction_\omega \quad \text{and} \quad \neg(b_1 \geq 1) = (b_1 \geq 1)\restriction_\omega$$

TOY EXAMPLE OF REDUNDANCE RULE

Is the following graph 2-colourable (bipartite)?



$b_1 + b_2 \geq 1$

$\overline{b}_1 + \overline{b}_2 \geq 1$

$b_1 + b_3 \geq 1$

$\overline{b}_1 + \overline{b}_3 \geq 1$

$\ldots$

Without loss of generality, node 1 is blue (otherwise we can swap the two colours)

Derive $b_1 \geq 1$ with redundance; witness $\omega : b_i \mapsto \overline{b}_i$

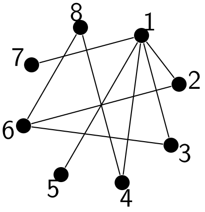
$$F = F\restriction_\omega \quad \text{and} \quad \neg(b_1 \geq 1) = (b_1 \geq 1)\restriction_\omega$$

```
pseudo-Boolean proof version 2.0
f 18
red 1 b1  >= 1 ; b1 -> ~b1 b2 -> ~b2
↪ b3 -> ~b3 b4 -> ~b4 b5 -> ~b5
↪ b6 -> ~b6 b7 -> ~b7 b8 -> ~b8
rup >= 1;
output NONE
conclusion UNSAT
end pseudo-Boolean proof
```

- Start the proof

TOY EXAMPLE OF REDUNDANCE RULE

Is the following graph 2-colourable (bipartite)?



$$\begin{aligned} b_1 + b_2 &\geq 1 \\ \overline{b}_1 + \overline{b}_2 &\geq 1 \\ b_1 + b_3 &\geq 1 \\ \overline{b}_1 + \overline{b}_3 &\geq 1 \\ &\ldots \end{aligned}$$

Without loss of generality, node 1 is blue (otherwise we can swap the two colours)

Derive $b_1 \geq 1$ with redundance; witness $\omega : b_i \mapsto \overline{b}_i$

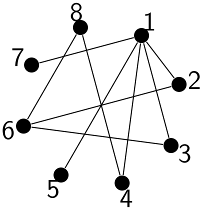
$$F = F\restriction_\omega \quad \text{and} \quad \neg(b_1 \geq 1) = (b_1 \geq 1)\restriction_\omega$$

```
pseudo-Boolean proof version 2.0
f 18
red 1 b1  >= 1 ; b1 -> ~b1 b2 -> ~b2
↪ b3 -> ~b3 b4 -> ~b4 b5 -> ~b5
↪ b6 -> ~b6 b7 -> ~b7 b8 -> ~b8
rup >= 1;
output NONE
conclusion UNSAT
end pseudo-Boolean proof
```

- Start the proof
- Derive $b_1 \geq 1$ with redundance rule

TOY EXAMPLE OF REDUNDANCE RULE

Is the following graph 2-colourable (bipartite)?



$$b_1 + b_2 \geq 1$$
$$\overline{b}_1 + \overline{b}_2 \geq 1$$
$$b_1 + b_3 \geq 1$$
$$\overline{b}_1 + \overline{b}_3 \geq 1$$
$$\ldots$$

Without loss of generality, node 1 is blue (otherwise we can swap the two colours)

Derive $b_1 \geq 1$ with redundance; witness $\omega : b_i \mapsto \overline{b}_i$

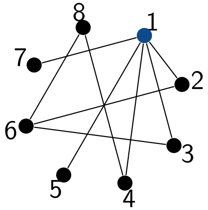
$$F = F\restriction_\omega \quad \text{and} \quad \neg(b_1 \geq 1) = (b_1 \geq 1)\restriction_\omega$$

```
pseudo-Boolean proof version 2.0
f 18
red 1 b1  >= 1 ; b1 -> ~b1 b2 -> ~b2
↪ b3 -> ~b3 b4 -> ~b4 b5 -> ~b5
↪ b6 -> ~b6 b7 -> ~b7 b8 -> ~b8
rup >= 1;
output NONE
conclusion UNSAT
end pseudo-Boolean proof
```

- Start the proof
- Derive $b_1 \geq 1$ with redundance rule
- Derive contradiction by RUP

TOY EXAMPLE OF REDUNDANCE RULE

Is the following graph 2-colourable (bipartite)?



$$b_1 + b_2 \geq 1$$
$$\overline{b}_1 + \overline{b}_2 \geq 1$$
$$b_1 + b_3 \geq 1$$
$$\overline{b}_1 + \overline{b}_3 \geq 1$$
$$\ldots$$

Without loss of generality, node 1 is blue (otherwise we can swap the two colours)

Derive $b_1 \geq 1$ with redundance; witness $\omega : b_i \mapsto \overline{b}_i$

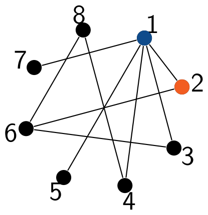
$$F = F\restriction_\omega \quad \text{and} \quad \neg(b_1 \geq 1) = (b_1 \geq 1)\restriction_\omega$$

```
pseudo-Boolean proof version 2.0
f 18
red 1 b1  >= 1 ; b1 -> ~b1 b2 -> ~b2
↪ b3 -> ~b3 b4 -> ~b4 b5 -> ~b5
↪ b6 -> ~b6 b7 -> ~b7 b8 -> ~b8
rup >= 1;
output NONE
conclusion UNSAT
end pseudo-Boolean proof
```

- Start the proof
- Derive $b_1 \geq 1$ with redundance rule
- Derive contradiction by RUP

TOY EXAMPLE OF REDUNDANCE RULE

Is the following graph 2-colourable (bipartite)?



$$b_1 + b_2 \geq 1$$
$$\overline{b}_1 + \overline{b}_2 \geq 1$$
$$b_1 + b_3 \geq 1$$
$$\overline{b}_1 + \overline{b}_3 \geq 1$$
$$\ldots$$

Without loss of generality, node 1 is blue (otherwise we can swap the two colours)

Derive $b_1 \geq 1$ with redundance; witness $\omega : b_i \mapsto \overline{b}_i$

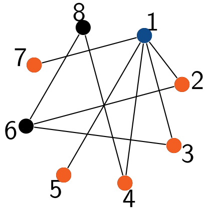
$$F = F{\restriction_\omega} \quad \text{and} \quad \neg(b_1 \geq 1) = (b_1 \geq 1){\restriction_\omega}$$

```
pseudo-Boolean proof version 2.0
f 18
red 1 b1  >= 1 ; b1 -> ~b1 b2 -> ~b2
↪ b3 -> ~b3 b4 -> ~b4 b5 -> ~b5
↪ b6 -> ~b6 b7 -> ~b7 b8 -> ~b8
rup >= 1;
output NONE
conclusion UNSAT
end pseudo-Boolean proof
```

- Start the proof
- Derive $b_1 \geq 1$ with redundance rule
- Derive contradiction by RUP

TOY EXAMPLE OF REDUNDANCE RULE

Is the following graph 2-colourable (bipartite)?



$b_1 + b_2 \geq 1$

$\bar{b}_1 + \bar{b}_2 \geq 1$

$b_1 + b_3 \geq 1$

$\bar{b}_1 + \bar{b}_3 \geq 1$

$\dots$

Without loss of generality, node 1 is blue (otherwise we can swap the two colours)

Derive $b_1 \geq 1$ with redundance; witness $\omega : b_i \mapsto \bar{b}_i$

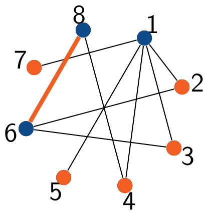
$$F = F\restriction_\omega \quad \text{and} \quad \neg(b_1 \geq 1) = (b_1 \geq 1)\restriction_\omega$$

```
pseudo-Boolean proof version 2.0
f 18
red 1 b1  >= 1 ; b1 -> ~b1 b2 -> ~b2
↪ b3 -> ~b3 b4 -> ~b4 b5 -> ~b5
↪ b6 -> ~b6 b7 -> ~b7 b8 -> ~b8
rup >= 1;
output NONE
conclusion UNSAT
end pseudo-Boolean proof
```

- Start the proof
- Derive $b_1 \geq 1$ with redundance rule
- Derive contradiction by RUP

TOY EXAMPLE OF REDUNDANCE RULE

Is the following graph 2-colourable (bipartite)?



$$b_1 + b_2 \geq 1$$
$$\overline{b}_1 + \overline{b}_2 \geq 1$$
$$b_1 + b_3 \geq 1$$
$$\overline{b}_1 + \overline{b}_3 \geq 1$$
$$\ldots$$

Without loss of generality, node 1 is blue (otherwise we can swap the two colours)

Derive $b_1 \geq 1$ with redundance; witness $\omega : b_i \mapsto \overline{b}_i$

$$F = F\restriction_\omega \quad \text{and} \quad \neg(b_1 \geq 1) = (b_1 \geq 1)\restriction_\omega$$

```
pseudo-Boolean proof version 2.0
f 18
red 1 b1  >= 1 ; b1 -> ~b1 b2 -> ~b2
↪ b3 -> ~b3 b4 -> ~b4 b5 -> ~b5
↪ b6 -> ~b6 b7 -> ~b7 b8 -> ~b8
rup >= 1;
output NONE
conclusion UNSAT
end pseudo-Boolean proof
```

- Start the proof
- Derive $b_1 \geq 1$ with redundance rule
- Derive contradiction by RUP

REDUNDANCE AND DOMINANCE RULES FOR OPTIMISATION

Redundance-based strengthening, optimisation version

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models (F \wedge C)\restriction_\omega \ \wedge \ f\restriction_\omega \leq f$$

REDUNDANCE AND DOMINANCE RULES FOR OPTIMISATION

Redundance-based strengthening, optimisation version

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models (F \wedge C)\restriction_\omega \wedge f\restriction_\omega \leq f$$

Can be more aggressive if witness ω strictly improves solution

REDUNDANCE AND DOMINANCE RULES FOR OPTIMISATION

Redundance-based strengthening, optimisation version

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models (F \wedge C)\restriction_\omega \;\wedge\; f\restriction_\omega \leq f$$

Can be more aggressive if witness ω strictly improves solution

Dominance-based strengthening (simplified)

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models F\restriction_\omega \;\wedge\; f\restriction_\omega < f$$

REDUNDANCE AND DOMINANCE RULES FOR OPTIMISATION

Redundance-based strengthening, optimisation version

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models (F \wedge C)\restriction_\omega \wedge f\restriction_\omega \leq f$$

Can be more aggressive if witness ω strictly improves solution

Dominance-based strengthening (simplified)

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models F\restriction_\omega \wedge f\restriction_\omega < f$$

- Applying ω should strictly decrease f
- If so, don't need to show that $C\restriction_\omega$ holds!

SOUNDNESS OF DOMINANCE RULE

Dominance-based strengthening (simplified)

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models F\restriction_\omega \wedge f\restriction_\omega < f$$

Why is this sound?

SOUNDNESS OF DOMINANCE RULE

Dominance-based strengthening (simplified)

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models F\restriction_\omega \ \wedge \ f\restriction_\omega < f$$

Why is this sound?

1. Suppose α satisfies F but falsifies C (i.e., satisfies $\neg C$)

SOUNDNESS OF DOMINANCE RULE

Dominance-based strengthening (simplified)

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models F\restriction_\omega \ \wedge \ f\restriction_\omega < f$$

Why is this sound?

1. Suppose α satisfies F but falsifies C (i.e., satisfies $\neg C$)
2. Then $\alpha \circ \omega$ satisfies F and $f(\alpha \circ \omega) < f(\alpha)$

SOUNDNESS OF DOMINANCE RULE

Dominance-based strengthening (simplified)

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models F\restriction_\omega \wedge f\restriction_\omega < f$$

Why is this sound?

1. Suppose α satisfies F but falsifies C (i.e., satisfies $\neg C$)
2. Then $\alpha \circ \omega$ satisfies F and $f(\alpha \circ \omega) < f(\alpha)$
3. If $\alpha \circ \omega$ satisfies C, we're done

SOUNDNESS OF DOMINANCE RULE

Dominance-based strengthening (simplified)

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models F\restriction_\omega \wedge f\restriction_\omega < f$$

Why is this sound?

1. Suppose α satisfies F but falsifies C (i.e., satisfies $\neg C$)
2. Then $\alpha \circ \omega$ satisfies F and $f(\alpha \circ \omega) < f(\alpha)$
3. If $\alpha \circ \omega$ satisfies C, we're done
4. Otherwise $(\alpha \circ \omega) \circ \omega$ satisfies F and $f\big((\alpha \circ \omega) \circ \omega\big) < f\big(\alpha \circ \omega\big)$

SOUNDNESS OF DOMINANCE RULE

Dominance-based strengthening (simplified)

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models F\restriction_\omega \ \wedge \ f\restriction_\omega < f$$

Why is this sound?

1. Suppose α satisfies F but falsifies C (i.e., satisfies $\neg C$)
2. Then $\alpha \circ \omega$ satisfies F and $f(\alpha \circ \omega) < f(\alpha)$
3. If $\alpha \circ \omega$ satisfies C, we're done
4. Otherwise $(\alpha \circ \omega) \circ \omega$ satisfies F and $f((\alpha \circ \omega) \circ \omega) < f(\alpha \circ \omega)$
5. If $(\alpha \circ \omega) \circ \omega$ satisfies C, we're done

SOUNDNESS OF DOMINANCE RULE

Dominance-based strengthening (simplified)

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models F\restriction_\omega \wedge f\restriction_\omega < f$$

Why is this sound?

1. Suppose α satisfies F but falsifies C (i.e., satisfies $\neg C$)
2. Then $\alpha \circ \omega$ satisfies F and $f(\alpha \circ \omega) < f(\alpha)$
3. If $\alpha \circ \omega$ satisfies C, we're done
4. Otherwise $(\alpha \circ \omega) \circ \omega$ satisfies F and $f\big((\alpha \circ \omega) \circ \omega\big) < f\big(\alpha \circ \omega\big)$
5. If $(\alpha \circ \omega) \circ \omega$ satisfies C, we're done
6. Otherwise $\big((\alpha \circ \omega) \circ \omega\big) \circ \omega$ satisfies F and $f\big(((\alpha \circ \omega) \circ \omega) \circ \omega\big) < f\big((\alpha \circ \omega) \circ \omega\big)$

SOUNDNESS OF DOMINANCE RULE

Dominance-based strengthening (simplified)

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models F\restriction_\omega \wedge f\restriction_\omega < f$$

Why is this sound?

1. Suppose α satisfies F but falsifies C (i.e., satisfies $\neg C$)
2. Then $\alpha \circ \omega$ satisfies F and $f(\alpha \circ \omega) < f(\alpha)$
3. If $\alpha \circ \omega$ satisfies C, we're done
4. Otherwise $(\alpha \circ \omega) \circ \omega$ satisfies F and $f\big((\alpha \circ \omega) \circ \omega\big) < f\big(\alpha \circ \omega\big)$
5. If $(\alpha \circ \omega) \circ \omega$ satisfies C, we're done
6. Otherwise $\big((\alpha \circ \omega) \circ \omega\big) \circ \omega$ satisfies F and $f\big(((\alpha \circ \omega) \circ \omega) \circ \omega\big) < f\big((\alpha \circ \omega) \circ \omega\big)$
7. ...

SOUNDNESS OF DOMINANCE RULE

Dominance-based strengthening (simplified)

Add constraint C to formula F if exists witness substitution ω s.t.

$$F \wedge \neg C \models F\restriction_\omega \wedge f\restriction_\omega < f$$

Why is this sound?

1. Suppose α satisfies F but falsifies C (i.e., satisfies $\neg C$)
2. Then $\alpha \circ \omega$ satisfies F and $f(\alpha \circ \omega) < f(\alpha)$
3. If $\alpha \circ \omega$ satisfies C, we're done
4. Otherwise $(\alpha \circ \omega) \circ \omega$ satisfies F and $f\big((\alpha \circ \omega) \circ \omega\big) < f\big(\alpha \circ \omega\big)$
5. If $(\alpha \circ \omega) \circ \omega$ satisfies C, we're done
6. Otherwise $\big((\alpha \circ \omega) \circ \omega\big) \circ \omega$ satisfies F and $f\big(((\alpha \circ \omega) \circ \omega) \circ \omega\big) < f\big((\alpha \circ \omega) \circ \omega\big)$
7. ...
8. Can't go on forever, so finally reach α' satisfying $F \wedge C$

STRENGTH OF DOMINANCE RULE

Dominance-based strengthening (stronger, still simplified)

If $C_1, C_2, \ldots, C_{m-1}$ have been derived from F (maybe using dominance), then can derive C_m if exists witness substitution ω s.t.

$$F \wedge \bigwedge_{i=1}^{m-1} C_i \wedge \neg C_m \models F\restriction_\omega \wedge f\restriction_\omega < f$$

Only consider F — no need to show that any $C_i\restriction_\omega$ implied!

STRENGTH OF DOMINANCE RULE

Dominance-based strengthening (stronger, still simplified)

If $C_1, C_2, \ldots, C_{m-1}$ have been derived from F (maybe using dominance), then can derive C_m if exists witness substitution ω s.t.

$$F \wedge \bigwedge_{i=1}^{m-1} C_i \wedge \neg C_m \models F\restriction_\omega \wedge f\restriction_\omega < f$$

Only consider F — no need to show that any $C_i\restriction_\omega$ implied!

Now why is *this* sound?

- Same inductive proof as before, but nested

STRENGTH OF DOMINANCE RULE

Dominance-based strengthening (stronger, still simplified)

If $C_1, C_2, \ldots, C_{m-1}$ have been derived from F (maybe using dominance), then can derive C_m if exists witness substitution ω s.t.

$$F \wedge \bigwedge_{i=1}^{m-1} C_i \wedge \neg C_m \models F\restriction_\omega \wedge f\restriction_\omega < f$$

Only consider F — no need to show that any $C_i\restriction_\omega$ implied!

Now why is *this* sound?

- Same inductive proof as before, but nested
- Or pick solution α minimizing f and argue by contradiction

STRENGTH OF DOMINANCE RULE

Dominance-based strengthening (stronger, still simplified)

If $C_1, C_2, \ldots, C_{m-1}$ have been derived from F (maybe using dominance), then can derive C_m if exists witness substitution ω s.t.

$$F \land \bigwedge_{i=1}^{m-1} C_i \land \neg C_m \models F\restriction_\omega \land f\restriction_\omega < f$$

Only consider F — no need to show that any $C_i\restriction_\omega$ implied!

Now why is *this* sound?

- Same inductive proof as before, but nested
- Or pick solution α minimizing f and argue by contradiction

Further extensions:

- Define dominance rule w.r.t. order independent of objective
- Switch between different orders in same proof
- See [BGMN23] for details

STRATEGY FOR SYMMETRY BREAKING IN SAT SOLVING

1. Pretend to solve optimisation problem minimizing $f \doteq \sum_{i=1}^{n} 2^{n-i} \cdot x_i$ (search for lexicographically smallest assignment satisfying formula)

STRATEGY FOR SYMMETRY BREAKING IN SAT SOLVING

1. Pretend to solve optimisation problem minimizing $f \doteq \sum_{i=1}^{n} 2^{n-i} \cdot x_i$ (search for lexicographically smallest assignment satisfying formula)
2. Derive (for proof log only) pseudo-Boolean version of lex-leader constraint

$$C_\sigma \quad \doteq \quad f \leq f\restriction_\sigma \quad \doteq \quad \sum_{i=1}^{n} 2^{n-i} \cdot (\sigma(x_i) - x_i) \geq 0$$

STRATEGY FOR SYMMETRY BREAKING IN SAT SOLVING

1. Pretend to solve optimisation problem minimizing $f \doteq \sum_{i=1}^{n} 2^{n-i} \cdot x_i$
   (search for lexicographically smallest assignment satisfying formula)

2. Derive (for proof log only) pseudo-Boolean version of lex-leader constraint

$$C_\sigma \quad \doteq \quad f \leq f\restriction_\sigma \quad \doteq \quad \sum_{i=1}^{n} 2^{n-i} \cdot (\sigma(x_i) - x_i) \geq 0$$

3. Derive CNF encoding of lex-leader constraint used by SAT solver from pseudo-Boolean constraint
   (in same spirit as [GMNO22])

$$y_0 \qquad\qquad \overline{y}_j \vee \overline{\sigma(x_j)} \vee x_j$$

$$\overline{y}_{j-1} \vee \overline{x}_j \vee \sigma(x_j) \qquad\qquad y_j \vee \overline{y}_{j-1} \vee \overline{x}_j$$

$$\overline{y}_j \vee y_{j-1} \qquad\qquad y_j \vee \overline{y}_{j-1} \vee \sigma(x_j)$$

STRATEGY FOR SYMMETRY BREAKING IN SAT SOLVING

1. Pretend to solve optimisation problem minimizing $f \doteq \sum_{i=1}^{n} 2^{n-i} \cdot x_i$ (search for lexicographically smallest assignment satisfying formula)
2. Derive (for proof log only) pseudo-Boolean version of lex-leader constraint

$$C_\sigma \quad \doteq \quad f \leq f\upharpoonright_\sigma \quad \doteq \quad \sum_{i=1}^{n} 2^{n-i} \cdot (\sigma(x_i) - x_i) \geq 0$$

3. Derive CNF encoding of lex-leader constraint used by SAT solver from pseudo-Boolean constraint (in same spirit as [GMNO22])

$$y_0 \geq 1 \qquad \overline{y}_j + \overline{\sigma(x_j)} + x_j \geq 1$$

$$\overline{y}_{j-1} + \overline{x}_j + \sigma(x_j) \geq 1 \qquad y_j + \overline{y}_{j-1} + \overline{x}_j \geq 1$$

$$\overline{y}_j + y_{j-1} \geq 1 \qquad y_j + \overline{y}_{j-1} + \sigma(x_j) \geq 1$$

SYMMETRY BREAKING: EXAMPLE

Example: Pigeonhole principle (PHP) formula

- Variables p_{ij} $(1 \leq i \leq 4, 1 \leq j \leq 3)$ true iff pigeon i in hole j
- Focus on pigeon symmetries — notation:
  - $\sigma_{(12)}$ swaps pigeons 1 and 2

SYMMETRY BREAKING: EXAMPLE

Example: Pigeonhole principle (PHP) formula

- Variables p_{ij} $(1 \leq i \leq 4, 1 \leq j \leq 3)$ true iff pigeon i in hole j
- Focus on pigeon symmetries — notation:
  - $\sigma_{(12)}$ swaps pigeons 1 and 2
    Formally: $\sigma_{(12)}(p_{1j}) = p_{2j}$ and $\sigma_{(12)}(p_{2j}) = p_{1j}$ for all j
  - $\sigma_{(1234)}$ shifts all pigeons

SYMMETRY BREAKING: EXAMPLE

Example: Pigeonhole principle (PHP) formula

- Variables p_{ij} $(1 \leq i \leq 4, 1 \leq j \leq 3)$ true iff pigeon i in hole j
- Focus on pigeon symmetries — notation:
  - $\sigma_{(12)}$ swaps pigeons 1 and 2
    Formally: $\sigma_{(12)}(p_{1j}) = p_{2j}$ and $\sigma_{(12)}(p_{2j}) = p_{1j}$ for all j
  - $\sigma_{(1234)}$ shifts all pigeons

Order: "Pick smallest hole for pigeon 1, then smallest for pigeon 2, ..."

$$f \doteq 2^{11} \cdot p_{13} + 2^{10} \cdot p_{12} + 2^{9} \cdot p_{11} + 2^{8} \cdot p_{23} + \cdots + 1 \cdot p_{41}$$

BREAKING A SINGLE SIMPLE SYMMETRY (EXAMPLE)

- F is a formula expressing PHP constraints with $F\restriction_{\sigma_{(12)}} = F$
- Add constraint C_{12} breaking $\sigma_{(12)}$ — should be satisfied by α iff α "at least as good" as $\sigma_{(12)}(\alpha)$

BREAKING A SINGLE SIMPLE SYMMETRY (EXAMPLE)

- F is a formula expressing PHP constraints with $F\restriction_{\sigma_{(12)}} = F$
- Add constraint C_{12} breaking $\sigma_{(12)}$ — should be satisfied by α iff α "at least as good" as $\sigma_{(12)}(\alpha)$

$$C_{12} \doteq f \leq f\restriction_{\sigma_{(12)}}$$

BREAKING A SINGLE SIMPLE SYMMETRY (EXAMPLE)

- F is a formula expressing PHP constraints with $F\restriction_{\sigma_{(12)}} = F$
- Add constraint C_{12} breaking $\sigma_{(12)}$ — should be satisfied by α iff α "at least as good" as $\sigma_{(12)}(\alpha)$

$$
\begin{aligned}
C_{12} &\doteq f \leq f\restriction_{\sigma_{(12)}} \\
&\doteq \sum_{i=1}^{n} 2^{n-i} \cdot \big(\sigma_{(12)}(x_i) - x_i\big) \geq 0
\end{aligned}
$$

BREAKING A SINGLE SIMPLE SYMMETRY (EXAMPLE)

- F is a formula expressing PHP constraints with $F\restriction_{\sigma_{(12)}} = F$
- Add constraint C_{12} breaking $\sigma_{(12)}$ — should be satisfied by α iff α "at least as good" as $\sigma_{(12)}(\alpha)$

$$
\begin{aligned}
C_{12} &\doteq f \leq f\restriction_{\sigma_{(12)}} \\
&\doteq \textstyle\sum_{i=1}^{n} 2^{n-i} \cdot \left(\sigma_{(12)}(x_i) - x_i\right) \geq 0 \\
&\doteq \left(2^{11} - 2^{8}\right)(p_{23} - p_{13}) + \left(2^{10} - 2^{7}\right)(p_{22} - p_{12}) + \left(2^{9} - 2^{6}\right)(p_{21} - p_{11}) \geq 0
\end{aligned}
$$

"Pigeon 1 in smaller hole than pigeon 2"

BREAKING A SINGLE SIMPLE SYMMETRY (EXAMPLE)

- F is a formula expressing PHP constraints with $F\restriction_{\sigma_{(12)}} = F$
- Add constraint C_{12} breaking $\sigma_{(12)}$ — should be satisfied by α iff α "at least as good" as $\sigma_{(12)}(\alpha)$

$$\begin{aligned} C_{12} &\doteq f \leq f\restriction_{\sigma_{(12)}} \\ &\doteq \sum_{i=1}^{n} 2^{n-i} \cdot \left(\sigma_{(12)}(x_i) - x_i\right) \geq 0 \\ &\doteq \left(2^{11} - 2^{8}\right)(p_{23} - p_{13}) + \left(2^{10} - 2^{7}\right)(p_{22} - p_{12}) + \left(2^{9} - 2^{6}\right)(p_{21} - p_{11}) \geq 0 \end{aligned}$$

"Pigeon 1 in smaller hole than pigeon 2"

- Can use redundance rule (the symmetry is the witness):

$$F \wedge \neg C_{12} \models F\restriction_{\sigma_{(12)}} \wedge C_{12}\restriction_{\sigma_{(12)}} \wedge f\restriction_{\sigma_{(12)}} \leq f$$

$$F \wedge \neg(f \leq f\restriction_{\sigma_{(12)}}) \models F\restriction_{\sigma_{(12)}} \wedge (f \leq f\restriction_{\sigma_{(12)}})\restriction_{\sigma_{(12)}} \wedge f\restriction_{\sigma_{(12)}} \leq f$$

BREAKING A SINGLE SIMPLE SYMMETRY (EXAMPLE)

- F is a formula expressing PHP constraints with $F\restriction_{\sigma_{(12)}} = F$
- Add constraint C_{12} breaking $\sigma_{(12)}$ — should be satisfied by α iff α "at least as good" as $\sigma_{(12)}(\alpha)$

$$\begin{aligned} C_{12} &\doteq f \leq f\restriction_{\sigma_{(12)}} \\ &\doteq \textstyle\sum_{i=1}^{n} 2^{n-i} \cdot \left(\sigma_{(12)}(x_i) - x_i\right) \geq 0 \\ &\doteq \left(2^{11} - 2^{8}\right)(p_{23} - p_{13}) + \left(2^{10} - 2^{7}\right)(p_{22} - p_{12}) + \left(2^{9} - 2^{6}\right)(p_{21} - p_{11}) \geq 0 \end{aligned}$$

"Pigeon 1 in smaller hole than pigeon 2"

- Can use redundance rule (the symmetry is the witness):

$$F \wedge \neg C_{12} \models F\restriction_{\sigma_{(12)}} \wedge C_{12}\restriction_{\sigma_{(12)}} \wedge f\restriction_{\sigma_{(12)}} \leq f$$

$$F \wedge f > f\restriction_{\sigma_{(12)}} \models F\restriction_{\sigma_{(12)}} \wedge f\restriction_{\sigma_{(12)}} \leq f \wedge f\restriction_{\sigma_{(12)}} \leq f$$

BREAKING A SINGLE SIMPLE SYMMETRY (EXAMPLE)

- F is a formula expressing PHP constraints with $F\restriction_{\sigma_{(12)}} = F$
- Add constraint C_{12} breaking $\sigma_{(12)}$ — should be satisfied by α iff α "at least as good" as $\sigma_{(12)}(\alpha)$

$$
\begin{aligned}
C_{12} &\doteq f \leq f\restriction_{\sigma_{(12)}} \\
&\doteq \sum_{i=1}^{n} 2^{n-i} \cdot \left(\sigma_{(12)}(x_i) - x_i\right) \geq 0 \\
&\doteq \left(2^{11} - 2^{8}\right)(p_{23} - p_{13}) + \left(2^{10} - 2^{7}\right)(p_{22} - p_{12}) + \left(2^{9} - 2^{6}\right)(p_{21} - p_{11}) \geq 0
\end{aligned}
$$

"Pigeon 1 in smaller hole than pigeon 2"

- Can use redundance rule (the symmetry is the witness):

$$
\begin{aligned}
F \wedge \neg C_{12} &\models F\restriction_{\sigma_{(12)}} \wedge C_{12}\restriction_{\sigma_{(12)}} \wedge f\restriction_{\sigma_{(12)}} \leq f \\
F \wedge f > f\restriction_{\sigma_{(12)}} &\models F\restriction_{\sigma_{(12)}} \wedge f\restriction_{\sigma_{(12)}} \leq f \wedge f\restriction_{\sigma_{(12)}} \leq f
\end{aligned}
$$

Similar to DRAT symmetry breaking [HHW15]

BREAKING MORE/OTHER SYMMETRIES

Problem

This idea does not generalize

- Breaking two symmetries

- Breaking complex symmetries

BREAKING MORE/OTHER SYMMETRIES

Problem

This idea does not generalize

- Breaking two symmetries

$$F \wedge C_{12} \wedge \neg C_{23} \not\models F\restriction_{\sigma_{(23)}} \wedge C_{12}\restriction_{\sigma_{(23)}} \wedge C_{23}\restriction_{\sigma_{(23)}} \wedge f\restriction_{\sigma_{(23)}} \leq f$$

Intuitively: applying $\sigma_{(23)}$ potentially falsifies C_{12}

- Breaking complex symmetries

BREAKING MORE/OTHER SYMMETRIES

Problem

This idea does not generalize

- Breaking two symmetries

$$F \wedge C_{12} \wedge \neg C_{23} \not\models F\restriction_{\sigma_{(23)}} \wedge C_{12}\restriction_{\sigma_{(23)}} \wedge C_{23}\restriction_{\sigma_{(23)}} \wedge f\restriction_{\sigma_{(23)}} \leq f$$

Intuitively: applying $\sigma_{(23)}$ potentially falsifies C_{12}
We might have to apply $\sigma_{(12)}$ again

- Breaking complex symmetries

BREAKING MORE/OTHER SYMMETRIES

Problem

This idea does not generalize

- Breaking two symmetries

$$F \wedge C_{12} \wedge \neg C_{23} \not\models F\restriction_{\sigma_{(23)}} \wedge C_{12}\restriction_{\sigma_{(23)}} \wedge C_{23}\restriction_{\sigma_{(23)}} \wedge f\restriction_{\sigma_{(23)}} \leq f$$

Intuitively: applying $\sigma_{(23)}$ potentially falsifies C_{12}
We might have to apply $\sigma_{(12)}$ again

- Breaking complex symmetries

$$F \wedge \neg C_{1234} \models F\restriction_{\sigma_{(1234)}} \wedge C_{1234}\restriction_{\sigma_{(1234)}} \wedge f\restriction_{\sigma_{(1234)}} \leq f$$

Intuitively, C_{1234} holds if shifting all the pigeons results in a worse assignment

BREAKING MORE/OTHER SYMMETRIES

Problem

This idea does not generalize

- Breaking two symmetries

$$F \wedge C_{12} \wedge \neg C_{23} \not\models F\restriction_{\sigma_{(23)}} \wedge C_{12}\restriction_{\sigma_{(23)}} \wedge C_{23}\restriction_{\sigma_{(23)}} \wedge f\restriction_{\sigma_{(23)}} \leq f$$

Intuitively: applying $\sigma_{(23)}$ potentially falsifies C_{12}
We might have to apply $\sigma_{(12)}$ again

- Breaking complex symmetries

$$F \wedge \neg C_{1234} \models F\restriction_{\sigma_{(1234)}} \wedge C_{1234}\restriction_{\sigma_{(1234)}} \wedge f\restriction_{\sigma_{(1234)}} \leq f$$

Intuitively, C_{1234} holds if shifting all the pigeons results in a worse assignment
Can satisfy this constraint by applying $\sigma_{(1234)}$ once, twice, or thrice

BREAKING SYMMETRIES WITH THE DOMINANCE RULE (1/2)

Definition

Given a symmetry σ, the (pseudo-Boolean) breaking constraint of σ is

$$C_\sigma \doteq f \leq f\restriction_\sigma$$

BREAKING SYMMETRIES WITH THE DOMINANCE RULE (1/2)

Definition

Given a symmetry σ, the (pseudo-Boolean) breaking constraint of σ is

$$C_\sigma \doteq f \leq f\restriction_\sigma$$

Theorem ([BGMN23])

C_σ can be derived from F using dominance with witness σ

$$F \wedge \neg C_\sigma \models F\restriction_\sigma \wedge f\restriction_\sigma < f$$

BREAKING SYMMETRIES WITH THE DOMINANCE RULE (2/2)

Breaking symmetries with the dominance rule

- Surprisingly simple

BREAKING SYMMETRIES WITH THE DOMINANCE RULE (2/2)

Breaking symmetries with the dominance rule

- Surprisingly simple
- Generalizes well

BREAKING SYMMETRIES WITH THE DOMINANCE RULE (2/2)

Breaking symmetries with the dominance rule

- Surprisingly simple
- Generalizes well
  - Works for arbitrary symmetries

BREAKING SYMMETRIES WITH THE DOMINANCE RULE (2/2)

Breaking symmetries with the dominance rule

- Surprisingly simple
- Generalizes well
  - Works for arbitrary symmetries
  - Works for multiple symmetries (can ignore previously derived symmetry breaking constraints)

$$F \land C_{12} \land \neg C_{23} \models F\restriction_{\sigma_{(23)}} \land f\restriction_{\sigma_{(23)}} < f$$

BREAKING SYMMETRIES WITH THE DOMINANCE RULE (2/2)

Breaking symmetries with the dominance rule

- Surprisingly simple
- Generalizes well
  - Works for arbitrary symmetries
  - Works for multiple symmetries (can ignore previously derived symmetry breaking constraints)

$$F \wedge C_{12} \wedge \neg C_{23} \models F\restriction_{\sigma_{(23)}} \wedge f\restriction_{\sigma_{(23)}} < f$$

Why does it work?

- Witness need not satisfy all derived constraints
- Sufficient to just produce “better” assignment

STRATEGY FOR SYMMETRY BREAKING IN SAT SOLVING

1. Pretend to solve optimisation problem minimizing $f \doteq \sum_{i=1}^{n} 2^{n-i} \cdot x_i$ (search for lexicographically smallest assignment satisfying formula)
2. Derive (for proof log only) pseudo-Boolean version of lex-leader constraint

$$C_\sigma \quad \doteq \quad f \leq f\restriction_\sigma \quad \doteq \quad \sum_{i=1}^{n} 2^{n-i} \cdot (\sigma(x_i) - x_i) \geq 0$$

3. Derive CNF encoding of lex-leader constraint used by SAT solver from pseudo-Boolean constraint (in same spirit as [GMNO22])

$$\begin{aligned} y_0 &\geq 1 & \overline{y}_j + \overline{\sigma(x_j)} + x_j &\geq 1 \\ \overline{y}_{j-1} + \overline{x}_j + \sigma(x_j) &\geq 1 & y_j + \overline{y}_{j-1} + \overline{x}_j &\geq 1 \\ \overline{y}_j + y_{j-1} &\geq 1 & y_j + \overline{y}_{j-1} + \sigma(x_j) &\geq 1 \end{aligned}$$

SYMMETRY BREAKING IN CNF

- In SAT symmetry breaking tools, symmetry is broken by adding clausal constraints

SYMMETRY BREAKING IN CNF

- In SAT symmetry breaking tools, symmetry is broken by adding clausal constraints
- Need to show how to derive this CNF encoding

SYMMETRY BREAKING IN CNF

- In SAT symmetry breaking tools, symmetry is broken by adding clausal constraints
- Need to show how to derive this CNF encoding
- We use the encoding of BREAKID [DBBD16]:

$$\begin{aligned} y_0 \\ \overline{y}_{j-1} + \overline{x}_j + \sigma(x_j) \\ \overline{y}_j + y_{j-1} \\ \overline{y}_j + \overline{\sigma(x_j)} + x_j \\ y_j + \overline{y}_{j-1} + \overline{x}_j \\ y_j + \overline{y}_{j-1} + \sigma(x_j) \end{aligned}$$

SYMMETRY BREAKING IN CNF

- In SAT symmetry breaking tools, symmetry is broken by adding clausal constraints
- Need to show how to derive this CNF encoding
- We use the encoding of BreakID [DBBD16]:

$$y_0$$
$$\overline{y}_{j-1} + \overline{x}_j + \sigma(x_j)$$
$$\overline{y}_j + y_{j-1}$$
$$\overline{y}_j + \overline{\sigma(x_j)} + x_j$$
$$y_j + \overline{y}_{j-1} + \overline{x}_j$$
$$y_j + \overline{y}_{j-1} + \sigma(x_j)$$

Define y_j true if x_k equals $\sigma(x_k)$ for all $k \leq j$

$$y_k \Leftrightarrow y_{k-1} \wedge (x_k \Leftrightarrow \sigma(x_k))$$

(derivable with redundance rule)

SYMMETRY BREAKING IN CNF

- In SAT symmetry breaking tools, symmetry is broken by adding clausal constraints
- Need to show how to derive this CNF encoding
- We use the encoding of BreakID [DBBD16]:

$$y_0$$

$$\overline{y}_{j-1} + \overline{x}_j + \sigma(x_j)$$

$$\overline{y}_j + y_{j-1}$$

$$\overline{y}_j + \overline{\sigma(x_j)} + x_j$$

$$y_j + \overline{y}_{j-1} + \overline{x}_j$$

$$y_j + \overline{y}_{j-1} + \sigma(x_j)$$

Define y_j true if x_k equals $\sigma(x_k)$ for all $k \leq j$

$$y_k \Leftrightarrow y_{k-1} \wedge (x_k \Leftrightarrow \sigma(x_k))$$

(derivable with redundance rule)

If y_{j-1} is true, x_j is at most $\sigma(x_j)$
(derivable from the PB breaking constraint)

BACK TO OUR PIGEONS — PRETEND OPTIMISATION PROBLEM

Start the proof and load input formula

```
pseudo-Boolean proof version 2.0
f 22
pre_order exp
  vars
    left  u1 u2 u3 u4 u5 u6 u7 u8 u9 u10 u11 u12
    right v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12
    aux
  end
  def
    -1 u12 1 v12 -2 u11 2 v11 [...] -1024 u2 1024 v2 -2048 u1 2048 v1 >= 0;
  end
  transitivity
    vars
      fresh_right w1 w2 w3 w4 w5 w6 w7 w8 w9 w10 w11 w12
    end
  proof
    proofgoal #1
      pol  1 2 + 3 +
      qed -1
    qed
  end
end
load_order exp p13 p12 p11 p23 p22 p21 p31 p32 p33 p41 p42 p43
```

BACK TO OUR PIGEONS — PRETEND OPTIMISATION PROBLEM

Start the proof and load input formula

1. Pretend to solve optimisation problem minimizing $f \doteq 2^{11} \cdot p_{13} + 2^{10} \cdot p_{12} + 2^9 \cdot p_{11} + 2^8 \cdot p_{23} + \cdots + 2 \cdot p_{42} + 1 \cdot p_{41}$

```
pseudo-Boolean proof version 2.0
f 22
pre_order exp
  vars
    left  u1 u2 u3 u4 u5 u6 u7 u8 u9 u10 u11 u12
    right v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12
    aux
  end
  def
    -1 u12 1 v12 -2 u11 2 v11 [...] -1024 u2 1024 v2 -2048 u1 2048 v1 >= 0;
  end
  transitivity
    vars
      fresh_right w1 w2 w3 w4 w5 w6 w7 w8 w9 w10 w11 w12
    end
  proof
    proofgoal #1
      pol  1 2 + 3 +
      qed -1
    qed
  end
end
load_order exp p13 p12 p11 p23 p22 p21 p31 p32 p33 p41 p42 p43
```

BACK TO OUR PIGEONS — PRETEND OPTIMISATION PROBLEM

Start the proof and load input formula

1. Pretend to solve optimisation problem minimizing $f \doteq 2^{11} \cdot p_{13} + 2^{10} \cdot p_{12} + 2^{9} \cdot p_{11} + 2^{8} \cdot p_{23} + \cdots + 2 \cdot p_{42} + 1 \cdot p_{41}$

(Actually defining an order — see [BGMN23] for details)

```
pseudo-Boolean proof version 2.0
f 22
pre_order exp
  vars
    left  u1 u2 u3 u4 u5 u6 u7 u8 u9 u10 u11 u12
    right v1 v2 v3 v4 v5 v6 v7 v8 v9 v10 v11 v12
    aux
  end
  def
    -1 u12 1 v12 -2 u11 2 v11 [...] -1024 u2 1024 v2 -2048 u1 2048 v1 >= 0;
  end
  transitivity
    vars
      fresh_right w1 w2 w3 w4 w5 w6 w7 w8 w9 w10 w11 w12
    end
  proof
    proofgoal #1
      pol  1 2 + 3 +
      qed -1
    qed
  end
end
load_order exp p13 p12 p11 p23 p22 p21 p31 p32 p33 p41 p42 p43
```

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Pseudo-Boolean breaking constraint

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \dots \geq 0$$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Pseudo-Boolean breaking constraint

Use dominance with witness $\sigma = (p_{11} p_{21})(p_{12} p_{22})(p_{13} p_{23})$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
 proofgoal #2
   pol -1 -2 +
 qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Pseudo-Boolean breaking constraint

Use dominance with witness $\sigma = (p_{11}p_{21})(p_{12}p_{22})(p_{13}p_{23})$

$$F \wedge \neg C_{12} \models F\restriction_\omega \wedge \left(f\restriction_\omega < f\right)$$

VeriPB fills in all missing subproofs except for $\neg C_{12} \wedge C_{12} \models \bot$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by redundance with witness $\omega = \{y_0 \mapsto 1\}$

$$F \wedge \mathcal{D} \wedge \neg(y_0) \models (F \wedge \mathcal{D})\restriction_\omega \wedge (y_0)\restriction_\omega$$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by redundance with witness $\omega = \{y_0 \mapsto 1\}$

$$F \wedge \mathcal{D} \wedge \neg(y_0) \models (F \wedge \mathcal{D})\restriction_\omega \wedge (y_0)\restriction_\omega$$

$$F \wedge \mathcal{D} \wedge (\overline{y}_0) \models (F \wedge \mathcal{D}) \quad \wedge (1)$$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by RUP

$$F \wedge \mathcal{D} \wedge \neg(\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13}))$$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by RUP

$$F \wedge \mathcal{D} \wedge \neg(\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})) \models F \wedge \mathcal{D} \wedge (y_0) \wedge (p_{13}) \wedge (\overline{p}_{23})$$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by RUP

$$F \wedge \mathcal{D} \wedge \neg(\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})) \models F \wedge \mathcal{D} \wedge (y_0) \wedge (p_{13}) \wedge (\overline{p}_{23})$$

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by RUP

$$F \wedge \mathcal{D} \wedge \neg(\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})) \models F \wedge \mathcal{D} \wedge (y_0) \wedge (p_{13}) \wedge (\overline{p}_{23})$$

$$2^{11} \cdot (\quad -1 \quad) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by RUP

$$F \wedge \mathcal{D} \wedge \neg(\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})) \models F \wedge \mathcal{D} \wedge (y_0) \wedge (p_{13}) \wedge (\overline{p}_{23})$$

$$2^{11} \cdot (\quad -1 \quad) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

where $\sum_{i=1}^{10} 2^i < 2^{11}$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

$\overline{y}_1 + y_0$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by redundance with witness $\omega = \{y_1 \mapsto 0\}$

$$F \wedge \mathcal{D} \wedge \neg(\overline{y}_1 + y_0) \models (F \wedge \mathcal{D})\restriction_\omega \wedge (\overline{y}_1 + y_0)\restriction_\omega$$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

$\overline{y}_1 + y_0$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0         >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1   >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1   >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by redundance with witness $\omega = \{y_1 \mapsto 0\}$

$$F \wedge \mathcal{D} \wedge \neg(\overline{y}_1 + y_0) \models (F \wedge \mathcal{D})\restriction_\omega \wedge (\overline{y}_1 + y_0)\restriction_\omega$$

$$F \wedge \mathcal{D} \wedge (y_1 + \overline{y}_0 \geq 2) \models (F \wedge \mathcal{D}) \quad \wedge (1 + y_0)$$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

$\overline{y}_1 + y_0$

$\overline{y}_1 + \overline{\sigma(p_{13})} + p_{13}$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by redundance with witness $\omega = \{y_1 \mapsto 0\}$ (essentially same argument)

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

$\overline{y}_1 + y_0$

$\overline{y}_1 + \overline{\sigma(p_{13})} + p_{13}$

$y_1 + \overline{y}_0 + \overline{p}_{13}$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by redundance with witness $\omega = \{y_1 \mapsto 1\}$

$$F \wedge \mathcal{D} \wedge \neg(y_1 + \overline{y}_0 + \overline{p}_{13}) \models (F \wedge \mathcal{D})\restriction_\omega \wedge (y_1 + \overline{y}_0 + \overline{p}_{13})\restriction_\omega$$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

$\overline{y}_1 + y_0$

$\overline{y}_1 + \overline{\sigma(p_{13})} + p_{13}$

$y_1 + \overline{y}_0 + \overline{p}_{13}$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by redundance with witness $\omega = \{y_1 \mapsto 1\}$

$$F \wedge \mathcal{D} \wedge \neg(y_1 + \overline{y}_0 + \overline{p}_{13}) \models (F \wedge \mathcal{D})\restriction_\omega \wedge (y_1 + \overline{y}_0 + \overline{p}_{13})\restriction_\omega$$

$$F \wedge \mathcal{D} \wedge (\overline{y}_1 + y_0 + p_{13} \geq 3) \models \cdots \wedge \mathcal{D}\restriction_\omega \wedge \ldots$$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

$\overline{y}_1 + y_0$

$\overline{y}_1 + \overline{\sigma(p_{13})} + p_{13}$

$y_1 + \overline{y}_0 + \overline{p}_{13}$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by redundance with witness $\omega = \{y_1 \mapsto 1\}$

$$F \wedge \mathcal{D} \wedge \neg(y_1 + \overline{y}_0 + \overline{p}_{13}) \models (F \wedge \mathcal{D})\restriction_\omega \wedge (y_1 + \overline{y}_0 + \overline{p}_{13})\restriction_\omega$$

$$F \wedge \mathcal{D} \wedge (\overline{y}_1 + y_0 + p_{13} \geq 3) \models \cdots \wedge \mathcal{D}\restriction_\omega \wedge \ldots$$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

$\overline{y}_1 + y_0$

$\overline{y}_1 + \overline{\sigma(p_{13})} + p_{13}$

$y_1 + \overline{y}_0 + \overline{p}_{13}$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by redundance with witness $\omega = \{y_1 \mapsto 1\}$

$$F \wedge \mathcal{D} \wedge \neg(y_1 + \overline{y}_0 + \overline{p}_{13}) \models (F \wedge \mathcal{D})\restriction_\omega \wedge (y_1 + \overline{y}_0 + \overline{p}_{13})\restriction_\omega$$

$$F \wedge \mathcal{D} \wedge (\overline{y}_1 + y_0 + p_{13} \geq 3) \models \cdots \wedge \mathcal{D}\restriction_\omega \wedge \ldots$$

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

$\overline{y}_1 + y_0$

$\overline{y}_1 + \overline{\sigma(p_{13})} + p_{13}$

$y_1 + \overline{y}_0 + \overline{p}_{13}$

$y_1 + \overline{y}_0 + \sigma(p_{13})$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Derivable by redundance with witness $\omega = \{y_1 \mapsto 1\}$
(same argument)

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \dots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

$\overline{y}_1 + y_0$

$\overline{y}_1 + \overline{\sigma(p_{13})} + p_{13}$

$y_1 + \overline{y}_0 + \overline{p}_{13}$

$y_1 + \overline{y}_0 + \sigma(p_{13})$

$2^{11} \cdot \overline{y}_1 + 2^{10} \cdot (p_{22} - p_{12}) \dots$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Simplify the pseudo-Boolean breaking constraint and delete original constraint

BACK TO OUR PIGEONS — DERIVING THE CONSTRAINTS

Derived constraints ($\mathcal{D}$):

$$2^{11} \cdot (p_{23} - p_{13}) + 2^{10} \cdot (p_{22} - p_{12}) + \cdots \geq 0$$

y_0

$\overline{y}_0 + \overline{p}_{13} + \sigma(p_{13})$

$\overline{y}_1 + y_0$

$\overline{y}_1 + \overline{\sigma(p_{13})} + p_{13}$

$y_1 + \overline{y}_0 + \overline{p}_{13}$

$y_1 + \overline{y}_0 + \sigma(p_{13})$

$2^{11} \cdot \overline{y}_1 + 2^{10} \cdot (p_{22} - p_{12}) \ldots$

$\overline{y}_1 + \overline{p}_{12} + \sigma(p_{22})$

```
dom -64 p21 64 [...] -2048 p13 2048 p23  >= 0 ; p11 -> p21 [...] p23 -> p13 ; begin
  proofgoal #2
    pol -1 -2 +
  qed -1
end
red 1 y0  >= 1 ; y0 -> 1
rup 1 ~y0  1 ~p13 1 p23 >= 1 ;
red 1 ~y1  1  y0        >= 1 ; y1 -> 0
red 1 ~y1  1 ~p23 1 p13 >= 1 ; y1 -> 0
red 1  p23 1 ~y0  1 y1  >= 1 ; y1 -> 1
red 1 ~p13 1 ~y0  1 y1  >= 1 ; y1 -> 1
pol 26 32 2048 * +
del id 26
rup 1 ~y1  1 ~p12 1 p22 >= 1 ;
```

Continue in the same way for following y_i-variables

...

SAT MODULO SYMMETRIES

- In previous talks, I claimed “looks like this applies to SMS as well”

SAT MODULO SYMMETRIES

- In previous talks, I claimed “looks like this applies to SMS as well”
- However...

SAT MODULO SYMMETRIES

- In previous talks, I claimed “looks like this applies to SMS as well”
- However... SMS used to generates all mathematical objects (e.g., graphs) modulo isomorphisms
  Challenges:
  - How to certify enumeration?

SAT MODULO SYMMETRIES

- In previous talks, I claimed "looks like this applies to SMS as well"
- However... SMS used to generates all mathematical objects (e.g., graphs) modulo isomorphisms
  Challenges:
  - How to certify enumeration?
  - Isomorphisms might not be symmetries of the SAT encoding

SAT MODULO SYMMETRIES

- In previous talks, I claimed “looks like this applies to SMS as well”
- However... SMS used to generates all mathematical objects (e.g., graphs) modulo isomorphisms
  Challenges:
  - How to certify enumeration?
  - Isomorphisms might not be symmetries of the SAT encoding
  - How to verify that symmetry breaking was complete?

CONCLUSION

- Variety of symmetry handling methods

CONCLUSION

- Variety of symmetry handling methods
- For static (and some dynamic) symmetry breaking, fully general symmetry breaking in VeriPB
  - **Challenge:** get this to work in (some extension of) DRAT

CONCLUSION

- Variety of symmetry handling methods
- For static (and some dynamic) symmetry breaking, fully general symmetry breaking in VeriPB
  Challenge: get this to work in (some extension of) DRAT
- Makes use heavily of dominance rule
  Challenge: analyze this rule (can extended Frege simulate it?)

CONCLUSION

- Variety of symmetry handling methods
- For static (and some dynamic) symmetry breaking, fully general symmetry breaking in VeriPB
  Challenge: get this to work in (some extension of) DRAT
- Makes use heavily of dominance rule
  Challenge: analyze this rule (can extended Frege simulate it?)
- Claim that this generalizes to dynamic symmetry breaking methods
  Challenge: Verify this for other dynamic symmetry breaking methods
  Challenge: What about SAT modulo symmetries?

CONCLUSION

- Variety of symmetry handling methods
- For static (and some dynamic) symmetry breaking, fully general symmetry breaking in VeriPB
  - **Challenge:** get this to work in (some extension of) DRAT
- Makes use heavily of dominance rule
  - **Challenge:** analyze this rule (can extended Frege simulate it?)
- Claim that this generalizes to dynamic symmetry breaking methods
  - **Challenge:** Verify this for other dynamic symmetry breaking methods
  - **Challenge:** What about SAT modulo symmetries?
- For symmetric learning, dedicated proof system has been developed
  - **Challenge:** develop certification in a formalism that doesn't know about symmetries
  - Proofs with lemmas?

ADVERTISEMENTS & POINTERS

- I'm moving & hiring postdocs @ KU Leuven. Join the proof logging revolution!

ADVERTISEMENTS & POINTERS

- I'm moving & hiring postdocs @ KU Leuven. Join the proof logging revolution!
- Veripb format and tools: `https://gitlab.com/MIAOresearch/software/VeriPB`
- Description of VeriPB checker [BMM+23] used in SAT 2023 competition (`https://satcompetition.github.io/2023/checkers.html`)
- Specific details on different proof logging techniques covered in research papers [EGMN20, GMN20, GMM+20, GN21, GMN22, GMNO22, VDB22, BBN+23, BGMN23, MM23]
- Lots of concrete example files at `https://gitlab.com/MIAOresearch/software/VeriPB`
- Examples from this talk and our tutorials `https://bartbogaerts.eu/talks/veripb-tutorial-series`

ADVERTISEMENTS & POINTERS

- I'm moving & hiring postdocs @ KU Leuven. Join the proof logging revolution!
- Veripb format and tools: `https://gitlab.com/MIAOresearch/software/VeriPB`
- Description of VeriPB checker [BMM$^+$23] used in SAT 2023 competition (`https://satcompetition.github.io/2023/checkers.html`)
- Specific details on different proof logging techniques covered in research papers [EGMN20, GMN20, GMM$^+$20, GN21, GMN22, GMNO22, VDB22, BBN$^+$23, BGMN23, MM23]
- Lots of concrete example files at `https://gitlab.com/MIAOresearch/software/VeriPB`
- Examples from this talk and our tutorials `https://bartbogaerts.eu/talks/veripb-tutorial-series`

Thank you for your attention!

REFERENCES

[ASM06] Fadi A. Aloul, Karem A. Sakallah, and Igor L. Markov. Efficient symmetry breaking for Boolean satisfiability. *IEEE Transactions on Computers*, 55(5):549–558, 2006.

[BBN+23] Jeremias Berg, Bart Bogaerts, Jakob Nordström, Andy Oertel, and Dieter Vandesande. Certified core-guided MaxSAT solving. In *Proceedings of the 29th International Conference on Automated Deduction (CADE-29)*, volume 14132 of *Lecture Notes in Computer Science*, pages 1–22. Springer, July 2023.

[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.

[BGMN23] Bart Bogaerts, Stephan Gocht, Ciaran McCreesh, and Jakob Nordström. Certified symmetry and dominance breaking for combinatorial optimisation. *Journal of Artificial Intelligence Research*, 77:1539–1589, August 2023. Preliminary version in *AAAI '22*.

[BKG19] Curtis Bright, Ilias S. Kotsireas, and Vijay Ganesh. SAT solvers and computer algebra systems: a powerful combination for mathematics. In Tima Pakfetrat, Guy-Vincent Jourdan, Kostas Kontogiannis, and Robert F. Enenkel, editors, *Proceedings of the 29th Annual International Conference on Computer Science and Software Engineering, CASCON 2019, Markham, Ontario, Canada, November 4-6, 2019*, pages 323–328. ACM, 2019.

[BMM+23] Bart Bogaerts, Ciaran McCreesh, Magnus O. Myreen, Jakob Nordström, Andy Oertel, and Yong Kiam Tan. Documentation of VeriPB and CakePB for the SAT competition 2023. Available at `https://satcompetition.github.io/2023/checkers.html`, March 2023.

REFERENCES

[BNOS10] Belaïd Benhamou, Tarek Nabhani, Richard Ostrowski, and Mohamed Réda Saïdi. Enhancing clause learning by symmetry in SAT solvers. In *Proceedings of the 2010 22Nd IEEE International Conference on Tools with Artificial Intelligence - Volume 01*, ICTAI '10, pages 329–335, Washington, DC, USA, 2010. IEEE Computer Society.

[BS07] Belaïd Benhamou and Mohamed Réda Saïdi. Dynamic detection and elimination of local symmetry in CSPs. 2007.

[BT19] Samuel R. Buss and Neil Thapen. DRAT proofs, propagation redundancy, and extended resolution. In *Proceedings of the 22nd International Conference on Theory and Applications of Satisfiability Testing (SAT '19)*, volume 11628 of *Lecture Notes in Computer Science*, pages 71–89. Springer, July 2019.

[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.

[DBB17] Jo Devriendt, Bart Bogaerts, and Maurice Bruynooghe. Symmetric explanation learning: Effective dynamic symmetry handling for SAT. In *Proceedings of the 20th International Conference on Theory and Applications of Satisfiability Testing (SAT '17)*, volume 10491 of *Lecture Notes in Computer Science*, pages 83–100. Springer, August 2017.

[DBBD16] Jo Devriendt, Bart Bogaerts, Maurice Bruynooghe, and Marc Denecker. Improved static symmetry breaking for SAT. In *Proceedings of the 19th International Conference on Theory and Applications of Satisfiability Testing (SAT '16)*, volume 9710 of *Lecture Notes in Computer Science*, pages 104–122. Springer, July 2016.

[DBD+12] Jo Devriendt, Bart Bogaerts, Broes De Cat, Marc Denecker, and Christopher Mears. Symmetry propagation: Improved dynamic symmetry breaking in SAT. In *IEEE 24th International Conference on Tools with Artificial Intelligence, ICTAI 2012, Athens, Greece, November 7-9, 2012*, pages 49–56. IEEE Computer Society, 2012.

REFERENCES

[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.

[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.

[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.

[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.

[Goc22] Stephan Gocht. Certifying correctness for combinatorial algorithms by using pseudo-boolean reasoning, 2022.

REFERENCES

[HHW15] Marijn J. H. Heule, Warren A. Hunt Jr., and Nathan Wetzler. Expressing symmetry breaking in DRAT proofs. In *Proceedings of the 25th International Conference on Automated Deduction (CADE-25)*, volume 9195 of *Lecture Notes in Computer Science*, pages 591–606. Springer, August 2015.

[HKM+05] Marijn Heule, Alexander Keur, Hans Van Maaren, Coen Stevens, and Mark Voortman. CNF symmetry breaking options in conflict driven SAT solving, 2005.

[KS21] Markus Kirchweger and Stefan Szeider. SAT modulo symmetries for graph generation. In Laurent D. Michel, editor, *27th International Conference on Principles and Practice of Constraint Programming, CP 2021, Montpellier, France (Virtual Conference), October 25-29, 2021*, volume 210 of *LIPIcs*, pages 34:1–34:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[MBCK18] Hakan Metin, Souheib Baarir, Maximilien Colange, and Fabrice Kordon. Cdclsym: Introducing effective symmetry breaking in SAT solving. In *TACAS 2018, Proceedings, Part I*, pages 99–114, 2018.

[MBK19] Hakan Metin, Souheib Baarir, and Fabrice Kordon. Composing symmetry propagation and effective symmetry breaking for SAT solving. In *NASA Formal Methods, Proceedings*, pages 316–332, 2019.

[MM23] Matthew McIlree and Ciaran McCreesh. Proof logging for smart extensional constraints. In *Proceedings of the 29th International Conference on Principles and Practice of Constraint Programming (CP '23)*, volume 280 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 26:1–26:17, August 2023.

[Sab09] Ashish Sabharwal. SymChaff: Exploiting symmetry in a structure-aware satisfiability solver. *Constraints*, 14(4):478–505, 2009.

REFERENCES

[SHvM09] Bas Schaafsma, Marijn Heule, and Hans van Maaren. Dynamic symmetry breaking by simulating Zykov contraction. In Oliver Kullmann, editor, *Theory and Applications of Satisfiability Testing - SAT 2009, 12th International Conference, SAT 2009, Swansea, UK, June 30 - July 3, 2009. Proceedings*, volume 5584 of *Lecture Notes in Computer Science*, pages 223–236. Springer, 2009.

[TD20] Rodrigue Konan Tchinda and Clémentin Tayou Djamégni. On certifying the UNSAT result of dynamic symmetry-handling-based SAT solvers. *Constraints*, 25(3–4):251–279, December 2020.

[VDB22] Dieter Vandesande, Wolf De Wulf, and Bart Bogaerts. QMaxSATpb: A certified MaxSAT solver. In *Proceedings of the 16th International Conference on Logic Programming and Non-monotonic Reasoning (LPNMR '22)*, volume 13416 of *Lecture Notes in Computer Science*, pages 429–442. Springer, September 2022.

Funded/Co-funded by the European Union (ERC, CertiFOX, 101122653). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.