On Tackling the Limits of Resolution in SAT Solving

Alexey Ignatiev[1,2], Antonio Morgado[1], and Joao Marques-Silva[1]

August 29, 2017

[1] LASIGE, FC, University of Lisbon, Portugal
[2] ISDCT SB RAS, Irkutsk, Russia

Definitions

CDCL SAT solvers use resolution:

CDCL SAT solvers use resolution:

$$\frac{x \vee A \qquad\qquad \neg x \vee B}{A \vee B}$$

CDCL SAT solvers use resolution:

$$\frac{x \vee A \qquad \neg x \vee B}{A \vee B}$$

Pigeonhole principle

$m + 1$ pigeons

by m holes

Pigeonhole principle

$m+1$ pigeons
by m holes

⇒

$\exists$ hole with
>1 pigeons

$m+1$ pigeons
by m holes

➡

$\exists$ hole with
>1 pigeons

CNF formulation:

$x_{ij} = true \Leftrightarrow$ pigeon i is at hole j

$m+1$ pigeons by m holes ⇒ $\exists$ hole with >1 pigeons

CNF formulation:

$x_{ij} = true \Leftrightarrow$ pigeon i is at hole j

$$\bigwedge_{i=1}^{m+1} \text{AtLeast1}(x_{i1}, \ldots, x_{im}) \;\wedge\; \bigwedge_{j=1}^{m} \text{AtMost1}(x_{1j}, \ldots, x_{m+1,j})$$

$m+1$ pigeons by m holes ⇒ $\exists$ hole with >1 pigeons

CNF formulation:

$x_{ij} = \mathit{true} \Leftrightarrow$ pigeon i is at hole j

$$\bigwedge_{i=1}^{m+1} \mathrm{AtLeast1}(x_{i1}, \ldots, x_{im}) \;\wedge\; \bigwedge_{j=1}^{m} \mathrm{AtMost1}(x_{1j}, \ldots, x_{m+1,j})$$

⇓

hard for resolution!

(A. Haken. The intractability of resolution. *TCS*, 39:297–308, 1985.)

Maximum satisfiability

$$\text{given } \mathcal{F} = \mathcal{H} \wedge \mathcal{S} \models \bot,$$

Maximum satisfiability

given $\mathcal{F} = \mathcal{H} \wedge \mathcal{S} \models \bot$,

satisfy $\mathcal{H}$ and maximize $\sum_{c \in \mathcal{S}} weight(c)$

Maximum satisfiability

$$\text{given } \mathcal{F} = \mathcal{H} \wedge \mathcal{S} \models \bot,$$

$$\text{satisfy } \mathcal{H} \text{ and maximize } \sum_{c \in \mathcal{S}} weight(c)$$

$$\begin{array}{lcccc} \mathcal{H} & = & (\neg x \vee \neg y, \top) & (\neg x \vee \neg z, \top) & (\neg y \vee \neg z, \top) \\ \mathcal{S} & = & (x, 10) & (y, 20) & (z, 40) \end{array}$$

Maximum satisfiability

$$\text{given } \mathcal{F} = \mathcal{H} \wedge \mathcal{S} \models \bot,$$

$$\text{satisfy } \mathcal{H} \text{ and maximize } \sum_{c \in \mathcal{S}} weight(c)$$

$$\begin{aligned}
\mathcal{H} &= (\neg x \vee \neg y, \top) \quad (\neg x \vee \neg z, \top) \quad (\neg y \vee \neg z, \top) \\
\mathcal{S} &= (x, 10) \quad (y, 20) \quad (z, 40)
\end{aligned}$$

Maximum satisfiability

$$\text{given } \mathcal{F} = \mathcal{H} \wedge \mathcal{S} \models \bot,$$

$$\text{satisfy } \mathcal{H} \text{ and maximize } \sum_{c \in \mathcal{S}} \mathit{weight}(c)$$

$$\begin{array}{lcccc} \mathcal{H} & = & (\neg x \vee \neg y, \top) & (\neg x \vee \neg z, \top) & (\neg y \vee \neg z, \top) \\ \mathcal{S} & = & (x, 10) & (y, 20) & (z, 40) \end{array}$$

Maximum satisfiability

$$\text{given } \mathcal{F} = \mathcal{H} \wedge \mathcal{S} \models \bot,$$

$$\text{satisfy } \mathcal{H} \text{ and maximize } \sum_{c \in \mathcal{S}} weight(c)$$

$$\begin{array}{lcccc} \mathcal{H} & = & (\neg x \vee \neg y, \top) & (\neg x \vee \neg z, \top) & (\neg y \vee \neg z, \top) \\ \mathcal{S} & = & (x, 10) & (y, 20) & (z, 40) \end{array}$$

Maximum satisfiability

given $\mathcal{F} = \mathcal{H} \wedge \mathcal{S} \models \bot$,

satisfy $\mathcal{H}$ and maximize $\sum_{c \in \mathcal{S}} weight(c)$

$$\begin{array}{lclll} \mathcal{H} & = & (\neg x \vee \neg y, \top) & (\neg x \vee \neg z, \top) & (\neg y \vee \neg z, \top) \\ \mathcal{S} & = & (x, 10) & (y, 20) & (z, 40) \end{array}$$

$$\Downarrow$$

$$\begin{array}{lclll} \mathcal{H} & = & (\neg x \vee \neg y, \top) & (\neg x \vee \neg z, \top) & (\neg y \vee \neg z, \top) \\ \mathcal{S} & = & (x, 10) & (y, 20) & (z, 40) \end{array}$$

MSU3 algorithm for MaxSAT

$$\mathcal{H} = (\neg x \vee \neg y, \top) \quad (\neg x \vee \neg z, \top) \quad (\neg y \vee \neg z, \top)$$

$$\mathcal{S} = (x, 1) \quad (y, 1) \quad (z, 1)$$

MSU3 algorithm for MaxSAT

$$\mathcal{H} = (\neg x \vee \neg y, \top) \quad (\neg x \vee \neg z, \top) \quad (\neg y \vee \neg z, \top)$$

$$\mathcal{S} = (x, 1) \quad (y, 1) \quad (z, 1)$$

MSU3 algorithm for MaxSAT

$$\mathcal{H} = (\neg x \lor \neg y, \top) \quad (\neg x \lor \neg z, \top) \quad (\neg y \lor \neg z, \top)$$

$$(r_1 + r_2 \leqslant 1, \top)$$

$$\mathcal{S} = \cancel{(x, 1)} \quad \cancel{(y, 1)} \quad (z, 1)$$

$$(x \lor r_1, 1) \quad (y \lor r_2, 1)$$

MSU3 algorithm for MaxSAT

$$\mathcal{H} = (\neg x \vee \neg y, \top) \quad (\neg x \vee \neg z, \top) \quad (\neg y \vee \neg z, \top)$$

$$(r_1 + r_2 \leqslant 1, \top)$$

$$\mathcal{S} = \cancel{(x, 1)} \quad \cancel{(y, 1)} \quad (z, 1)$$

$$(x \vee r_1, 1) \quad (y \vee r_2, 1)$$

$$\mathcal{H} = (\neg x \vee \neg y, \top) \quad (\neg x \vee \neg z, \top) \quad (\neg y \vee \neg z, \top)$$

$$\sout{(r_1 + r_2 \leqslant 1, \top)} \quad (r_1 + r_2 + r_3 \leqslant 2, \top)$$

$$\mathcal{S} = \sout{(x, 1)} \quad \sout{(y, 1)} \quad \sout{(z, 1)}$$

$$(x \vee r_1, 1) \quad (y \vee r_2, 1)$$

$$(z \vee r_3, 1)$$

MSU3 algorithm for MaxSAT

$$\mathcal{H} = (\neg x \vee \neg y, \top) \quad (\neg x \vee \neg z, \top) \quad (\neg y \vee \neg z, \top)$$

$$\cancel{(r_1 + r_2 \leqslant 1, \top)} \quad (r_1 + r_2 + r_3 \leqslant 2, \top)$$

$$\mathcal{S} = \cancel{(x, 1)} \quad \cancel{(y, 1)} \quad \cancel{(z, 1)}$$

$$(x \vee r_1, 1) \quad (y \vee r_2, 1)$$

$$(z \vee r_3, 1)$$

MSU3 algorithm for MaxSAT

$$\mathcal{H} = (\neg x \vee \neg y, \top) \quad (\neg x \vee \neg z, \top) \quad (\neg y \vee \neg z, \top)$$

$$\cancel{(r_1 + r_2 \leqslant 1, \top)} \quad (r_1 + r_2 + r_3 \leqslant 2, \top)$$

$$\mathcal{S} = \cancel{(x, 1)} \quad \cancel{(y, 1)} \quad \cancel{(z, 1)}$$

$$(x \vee r_1, 1) \quad (y \vee r_2, 1)$$

$$(z \vee r_3, 1)$$

$$cost = 2$$

Approach

Approach

input: $\mathcal{F}$

$\texttt{HEnc}(\mathcal{F}) = \langle \mathcal{H}, \mathcal{S} \rangle \leftarrow \texttt{DualRailEncode}(\mathcal{F})$

$\text{cost} \leftarrow \texttt{ApplyMaxSAT}(\texttt{HEnc}(\mathcal{F}))$

Approach

input: $\mathcal{F}$

$\texttt{HEnc}(\mathcal{F}) = \langle \mathcal{H}, \mathcal{S} \rangle \leftarrow \texttt{DualRailEncode}(\mathcal{F})$

cost $\leftarrow \texttt{ApplyMaxSAT}(\texttt{HEnc}(\mathcal{F}))$

if cost $\leqslant |\text{var}(\mathcal{F})|$**:**

    return *true*

else:

    return *false*

Dual-rail encoding

$\forall x_i \in \mathrm{var}(\mathcal{F})$

Dual-rail encoding

$$\forall x_i \in \text{var}(\mathcal{F}) \quad \Rightarrow \quad \begin{cases} (p_i, 1) \\ (n_i, 1) \\ (\neg p_i \lor \neg n_i, \top) \end{cases}$$

Dual-rail encoding

$$\forall x_i \in \mathrm{var}(\mathcal{F}) \quad \Rightarrow \quad \begin{cases} (p_i, 1) \\ (n_i, 1) \\ (\neg p_i \vee \neg n_i, \top) \end{cases}$$

$$\forall c_i \in \mathcal{F},$$
$$c_i = (l_{i1} \vee \ldots \vee l_{ik_i})$$

Dual-rail encoding

$$\forall x_i \in \mathrm{var}(\mathcal{F}) \quad \Rightarrow \quad \begin{cases} (p_i, 1) \\ (n_i, 1) \\ (\neg p_i \vee \neg n_i, \top) \end{cases}$$

$$\forall c_i \in \mathcal{F},\ c_i = (l_{i1} \vee \ldots \vee l_{ik_i}) \quad \Rightarrow \quad \begin{cases} (\neg y_{i1} \vee \ldots \vee \neg y_{ik_i}, \top) \text{ s.t.} \\ y_{ij} \leftarrow p_{ij} \text{ if } l_{ij} = \neg x_{ij} \\ y_{ij} \leftarrow n_{ij} \text{ if } l_{ij} = x_{ij} \end{cases}$$

Dual-rail encoding

$\forall x_i \in \text{var}(\mathcal{F})$ ➡ $\begin{cases} (p_i, 1) \\ (n_i, 1) \\ (\neg p_i \vee \neg n_i, \top) \end{cases}$

$\forall c_i \in \mathcal{F},$
$c_i = (l_{i1} \vee \ldots \vee l_{ik_i})$ ➡ $\begin{cases} (\neg y_{i1} \vee \ldots \vee \neg y_{ik_i}, \top) \text{ s.t.} \\ y_{ij} \leftarrow p_{ij} \text{ if } l_{ij} = \neg x_{ij} \\ y_{ij} \leftarrow n_{ij} \text{ if } l_{ij} = x_{ij} \end{cases}$

⇩

Horn MaxSAT formula!

Dual-rail encoding (example)

$$\mathcal{F} \qquad (\neg x_1 \vee \neg x_2) \quad (x_2)$$

Dual-rail encoding (example)

$\mathcal{F}$ $(\neg x_1 \vee \neg x_2)$ (x_2)

$\Downarrow$

$\mathcal{S}$ $(p_1, 1)$ $(n_1, 1)$ $(p_2, 1)$ $(n_2, 1)$

$\mathcal{P}$ $(\neg p_1 \vee \neg n_1, \top)$ $(\neg p_2 \vee \neg n_2, \top)$

Dual-rail encoding (example)

$\mathcal{F}$ $\quad (\neg x_1 \vee \neg x_2) \quad (x_2)$

$\Downarrow$

$\mathcal{S}$ $\quad (p_1, 1) \quad (n_1, 1) \quad (p_2, 1) \quad (n_2, 1)$

$\mathcal{P}$ $\quad (\neg p_1 \vee \neg n_1, \top) \quad (\neg p_2 \vee \neg n_2, \top)$

$\mathcal{H}$ $\quad (\neg p_1 \vee \neg p_2, \top) \quad (\neg n_2, \top)$

Dual-rail encoding (example)

$\mathcal{F}$ $(\neg x_1 \vee \neg x_2)$ (x_2)

$\mathcal{S}$ $(p_1, 1)$ $(n_1, 1)$ $(p_2, 1)$ $(n_2, 1)$

$\mathcal{P}$ $(\neg p_1 \vee \neg n_1, \top)$ $(\neg p_2 \vee \neg n_2, \top)$

$\mathcal{H}$ $(\neg p_1 \vee \neg p_2, \top)$ $(\neg n_2, \top)$

Dual-rail encoding (example)

$$\mathcal{F} \qquad (\neg x_1 \vee \neg x_2) \quad (x_2)$$

$$\Downarrow$$

$$\mathcal{S} \qquad (p_1, 1) \quad (n_1, 1) \quad (p_2, 1) \quad (n_2, 1)$$

$$\mathcal{P} \qquad (\neg p_1 \vee \neg n_1, \top) \quad (\neg p_2 \vee \neg n_2, \top)$$

$$\mathcal{H} \qquad (\neg p_1 \vee \neg p_2, \top) \quad (\neg n_2, \top)$$

Dual-rail encoding (example)

$$\mathcal{F} \qquad (\neg x_1 \vee \neg x_2) \quad (x_2)$$

$$\Downarrow$$

$$\mathcal{S} \qquad (p_1, 1) \quad (n_1, 1) \quad (p_2, 1) \quad (n_2, 1)$$

$$\mathcal{P} \qquad (\neg p_1 \vee \neg n_1, \top) \quad (\neg p_2 \vee \neg n_2, \top)$$

$$\mathcal{H} \qquad (\neg p_1 \vee \neg p_2, \top) \quad (\neg n_2, \top)$$

$$\text{cost} = 2$$

($\mathcal{F}$ is satisfiable)

Dual-rail encoding (example)

$\mathcal{F} \quad (x_1) \quad (\neg x_1 \vee \neg x_2) \quad (x_2)$

⬇

$\mathcal{S} \quad (p_1, 1) \quad (n_1, 1) \quad (p_2, 1) \quad (n_2, 1)$

$\mathcal{P} \quad (\neg p_1 \vee \neg n_1, \top) \quad (\neg p_2 \vee \neg n_2, \top)$

$\mathcal{H} \quad (\neg p_1 \vee \neg p_2, \top) \quad (\neg n_2, \top)$

Dual-rail encoding (example)

$$\mathcal{F} \quad (x_1) \quad (\neg x_1 \vee \neg x_2) \quad (x_2)$$

⬇

$$\mathcal{S} \quad (p_1, 1) \quad (n_1, 1) \quad (p_2, 1) \quad (n_2, 1)$$

$$\mathcal{P} \quad (\neg p_1 \vee \neg n_1, \top) \quad (\neg p_2 \vee \neg n_2, \top)$$

$$\mathcal{H} \quad (\neg n_1, \top) \quad (\neg p_1 \vee \neg p_2, \top) \quad (\neg n_2, \top)$$

Dual-rail encoding (example)

$\mathcal{F} \quad (x_1) \quad (\neg x_1 \vee \neg x_2) \quad (x_2)$

⬇

$\mathcal{S} \quad (p_1, 1) \quad (n_1, 1) \quad (p_2, 1) \quad (n_2, 1)$

$\mathcal{P} \quad (\neg p_1 \vee \neg n_1, \top) \quad (\neg p_2 \vee \neg n_2, \top)$

$\mathcal{H} \quad (\neg n_1, \top) \quad (\neg p_1 \vee \neg p_2, \top) \quad (\neg n_2, \top)$

$\text{cost} = 3$

($\mathcal{F}$ is unsatisfiable)

Dual-rail encoding PHP

$$x_{ij},\quad \left.\begin{matrix} 1 \leqslant i \leqslant m+1 \\ 1 \leqslant j \leqslant m \end{matrix}\right\} m \cdot (m+1) \text{ vars} \quad \Rightarrow \quad n_{ij} \text{ and } p_{ij}$$

Dual-rail encoding PHP

$x_{ij}, \quad \left.\begin{matrix} 1 \leqslant i \leqslant m+1 \\ 1 \leqslant j \leqslant m \end{matrix}\right\} m \cdot (m+1)$ vars ➡ n_{ij} and p_{ij}

$\mathcal{P}$ $\{(\neg p_{ij} \vee \neg n_{ij}, \top) \mid 1 \leqslant i \leqslant m+1, 1 \leqslant j \leqslant m\}$

Dual-rail encoding PHP

x_{ij}, $\left.\begin{matrix} 1 \leqslant i \leqslant m+1 \\ 1 \leqslant j \leqslant m \end{matrix}\right\}$ $m \cdot (m+1)$ vars ➡ n_{ij} and p_{ij}

$\mathcal{P}$ $\{(\neg p_{ij} \vee \neg n_{ij}, \top) \mid 1 \leqslant i \leqslant m+1, 1 \leqslant j \leqslant m\}$

$\mathcal{L}_i$ $\text{AtLeast1}(x_{i1}, \ldots, x_{im}) = (x_{i1} \vee \ldots \vee x_{im})$ ➡ $(\neg n_{i1} \vee \ldots \vee \neg n_{im}, \top)$

Dual-rail encoding PHP

x_{ij}, $\left.\begin{array}{c} 1 \leqslant i \leqslant m+1 \\ 1 \leqslant j \leqslant m \end{array}\right\} m \cdot (m+1)$ vars ➡ n_{ij} and p_{ij}

$\mathcal{P}$ $\{(\neg p_{ij} \vee \neg n_{ij}, \top) \mid 1 \leqslant i \leqslant m+1, 1 \leqslant j \leqslant m\}$

$\mathcal{L}_i$ $\text{AtLeast1}(x_{i1}, \ldots, x_{im}) = (x_{i1} \vee \ldots \vee x_{im})$ ➡ $(\neg n_{i1} \vee \ldots \vee \neg n_{im}, \top)$

$\mathcal{M}_j$ $\text{AtMost1}(x_{1j}, \ldots, x_{m+1,j}) = \{(\neg x_{kj} \vee \neg x_{lj}) \mid 1 \leqslant k < l \leqslant m+1\}$ ➡ $\{(\neg p_{kj} \vee \neg p_{lj}, \top) \mid 1 \leqslant k < l \leqslant m+1\}$

Dual-rail encoding PHP

$x_{ij},\quad \left.\begin{array}{c} 1 \leqslant i \leqslant m+1 \\ 1 \leqslant j \leqslant m \end{array}\right\} m \cdot (m+1)$ vars ➡ n_{ij} and p_{ij}

$\mathcal{P}$ $\quad \{(\neg p_{ij} \vee \neg n_{ij}, \top) \mid 1 \leqslant i \leqslant m+1, 1 \leqslant j \leqslant m\}$

$\mathcal{L}_i$ $\quad \text{AtLeast1}(x_{i1}, \ldots, x_{im}) = (x_{i1} \vee \ldots \vee x_{im})$ ➡ $(\neg n_{i1} \vee \ldots \vee \neg n_{im}, \top)$

$\mathcal{M}_j$ $\quad \text{AtMost1}(x_{1j}, \ldots, x_{m+1,j}) = \{(\neg x_{kj} \vee \neg x_{lj}) \mid 1 \leqslant k < l \leqslant m+1\}$ ➡ $\{(\neg p_{kj} \vee \neg p_{lj}, \top) \mid 1 \leqslant k < l \leqslant m+1\}$

$\mathcal{S}$ $\quad \{(p_{ij}, 1), (n_{ij}, 1) \mid 1 \leqslant i \leqslant m+1, 1 \leqslant j \leqslant m\}$

Dual-rail encoding PHP

x_{ij}, $\left.\begin{matrix} 1 \leqslant i \leqslant m+1 \\ 1 \leqslant j \leqslant m \end{matrix}\right\}$ $m \cdot (m+1)$ vars ➡ n_{ij} and p_{ij}

$\mathcal{P}$ $\{(\neg p_{ij} \vee \neg n_{ij}, \top) \mid 1 \leqslant i \leqslant m+1, 1 \leqslant j \leqslant m\}$

$\mathcal{L}_i$ $\text{AtLeast1}(x_{i1}, \ldots, x_{im}) = (x_{i1} \vee \ldots \vee x_{im})$ ➡ $(\neg n_{i1} \vee \ldots \vee \neg n_{im}, \top)$

$\mathcal{M}_j$ $\text{AtMost1}(x_{1j}, \ldots, x_{m+1,j}) = \{(\neg x_{kj} \vee \neg x_{lj}) \mid 1 \leqslant k < l \leqslant m+1\}$ ➡ $\{(\neg p_{kj} \vee \neg p_{lj}, \top) \mid 1 \leqslant k < l \leqslant m+1\}$

$\mathcal{S}$ $\{(p_{ij}, 1), (n_{ij}, 1) \mid 1 \leqslant i \leqslant m+1, 1 \leqslant j \leqslant m\}$

$$\text{HEnc}(\text{PHP}_m^{m+1}) \triangleq \langle \mathcal{H}, \mathcal{S} \rangle = \left\langle \bigwedge_{i=1}^{m+1} \mathcal{L}_i \wedge \bigwedge_{j=1}^{m} \mathcal{M}_j \wedge \mathcal{P}, \mathcal{S} \right\rangle$$

DRE+MaxSAT for PHP in polynomial time

1. assume MSU3 algorithm
    - analyze ***disjoint*** sets separately

DRE+MaxSAT for PHP in polynomial time

1. assume MSU3 algorithm
   - analyze ***disjoint*** sets separately
2. relate soft clauses with each $\mathcal{L}_i$ and $\mathcal{M}_j$
   - each constraint disjoint from the others — but not from $\mathcal{P}$

DRE+MaxSAT for PHP in polynomial time

1. assume MSU3 algorithm
   - analyze ***disjoint*** sets separately
2. relate soft clauses with each $\mathcal{L}_i$ and $\mathcal{M}_j$
   - each constraint disjoint from the others — but not from $\mathcal{P}$
3. derive large enough lower bound on # of falsified clauses:

DRE+MaxSAT for PHP in polynomial time

1. assume MSU3 algorithm
   - analyze ***disjoint*** sets separately
2. relate soft clauses with each $\mathcal{L}_i$ and $\mathcal{M}_j$
   - each constraint disjoint from the others — but not from $\mathcal{P}$
3. derive large enough lower bound on # of falsified clauses:

Constr. type	# falsified cls	# constr	in total
$\mathcal{L}_i$	1	$i = 1, \ldots, m+1$	$m+1$
$\mathcal{M}_j$	m	$j = 1, \ldots, m$	$m \cdot m$
			$m \cdot (m+1) + 1$

DRE+MaxSAT for PHP in polynomial time

1. assume MSU3 algorithm
   - analyze ***disjoint*** sets separately
2. relate soft clauses with each $\mathcal{L}_i$ and $\mathcal{M}_j$
   - each constraint disjoint from the others — but not from $\mathcal{P}$
3. derive large enough lower bound on # of falsified clauses:

Constr. type	# falsified cls	# constr	in total
$\mathcal{L}_i$	1	$i = 1, \ldots, m+1$	$m+1$
$\mathcal{M}_j$	m	$j = 1, \ldots, m$	$m \cdot m$
			$m \cdot (m+1) + 1$

4. each lower bound increase — by unit propagation

DRE+MaxSAT for PHP in polynomial time

Constraint	Hard clause(s)	Soft clause(s)	Relaxed clauses	Updated AtMost$k$ Constraints	LB increase
$\mathcal{L}_i$	$(\neg n_{i1} \vee \ldots \vee \neg n_{im})$	$(n_{i1}), \ldots, (n_{im})$	$(r_{il} \vee n_{il})$, $1 \leqslant l \leqslant m$	$\sum_{l=1}^{m} r_{il} \leqslant 1$	1
$\mathcal{M}_j$	$(\neg p_{1j} \vee \neg p_{2j})$	$(p_{1j}), (p_{2j})$	$(r_{1j} \vee p_{1j})$, $(r_{2j} \vee p_{2j})$	$\sum_{l=1}^{2} r_{lj} \leqslant 1$	1
	$(\neg p_{1j} \vee \neg p_{3j})$, $(\neg p_{2j} \vee \neg p_{3j})$, $(r_{1j} \vee p_{1j})$, $(r_{2j} \vee p_{2j})$, $\sum_{l=1}^{2} r_{lj} \leqslant 1$	(p_{3j})	$(r_{3j} \vee p_{3j})$	$\sum_{l=1}^{3} r_{lj} \leqslant 2$	1
	$\cdots$ ($m - 3$ times)				
	$(\neg p_{1j} \vee \neg p_{m+1j}), \ldots,$ $(\neg p_{mj} \vee \neg p_{m+1j})$, $(r_{1j} \vee p_{1j}), \ldots,$ $(r_{mj} \vee p_{mj})$, $\sum_{l=1}^{m} r_{lj} \leqslant m - 1$	(p_{m+1j})	$(r_{m+1j} \vee p_{m+1j})$	$\sum_{l=1}^{m+1} r_{lj} \leqslant m$	1

DRE+MaxSAT for PHP in polynomial time

Constraint	Hard clause(s)	Soft clause(s)	Relaxed clauses	Updated AtMost$k$ Constraints	LB increase
$\mathcal{L}_i$	$(\neg n_{i1} \vee \ldots \vee \neg n_{im})$	$(n_{i1}), \ldots, (n_{im})$	$(r_{il} \vee n_{il})$, $1 \leqslant l \leqslant m$	$\sum_{l=1}^{m} r_{il} \leqslant 1$	1

DRE+MaxSAT for PHP in polynomial time

Constraint	Hard clause(s)	Soft clause(s)	Relaxed clauses	Updated AtMost$k$ Constraints	LB increase
$\mathcal{L}_i$	$(\neg n_{i1} \vee \ldots \vee \neg n_{im})$	$(n_{i1}), \ldots, (n_{im})$	$(r_{il} \vee n_{il})$, $1 \leqslant l \leqslant m$	$\sum_{l=1}^{m} r_{il} \leqslant 1$	1
$\mathcal{M}_j$	$(\neg p_{1j} \vee \neg p_{2j})$	$(p_{1j}), (p_{2j})$	$(r_{1j} \vee p_{1j})$, $(r_{2j} \vee p_{2j})$	$\sum_{l=1}^{2} r_{lj} \leqslant 1$	1

DRE+MaxSAT for PHP in polynomial time

Constraint	Hard clause(s)	Soft clause(s)	Relaxed clauses	Updated AtMost$k$ Constraints	LB increase
$\mathcal{L}_i$	$(\neg n_{i1} \vee \ldots \vee \neg n_{im})$	$(n_{i1}), \ldots, (n_{im})$	$(r_{il} \vee n_{il})$, $1 \leqslant l \leqslant m$	$\sum_{l=1}^{m} r_{il} \leqslant 1$	1
	$(\neg p_{1j} \vee \neg p_{2j})$	$(p_{1j}), (p_{2j})$	$(r_{1j} \vee p_{1j})$, $(r_{2j} \vee p_{2j})$	$\sum_{l=1}^{2} r_{lj} \leqslant 1$	1
$\mathcal{M}_j$	$(\neg p_{1j} \vee \neg p_{3j})$, $(\neg p_{2j} \vee \neg p_{3j})$, $(r_{1j} \vee p_{1j})$, $(r_{2j} \vee p_{2j})$, $\sum_{l=1}^{2} r_{lj} \leqslant 1$	(p_{3j})	$(r_{3j} \vee p_{3j})$	$\sum_{l=1}^{3} r_{lj} \leqslant 2$	1

DRE+MaxSAT for PHP in polynomial time

Constraint	Hard clause(s)	Soft clause(s)	Relaxed clauses	Updated AtMost$k$ Constraints	LB increase
$\mathcal{L}_i$	$(\neg n_{i1} \vee \ldots \vee \neg n_{im})$	$(n_{i1}), \ldots, (n_{im})$	$(r_{il} \vee n_{il})$, $1 \leqslant l \leqslant m$	$\sum_{l=1}^{m} r_{il} \leqslant 1$	1
	$(\neg p_{1j} \vee \neg p_{2j})$	$(p_{1j}), (p_{2j})$	$(r_{1j} \vee p_{1j})$, $(r_{2j} \vee p_{2j})$	$\sum_{l=1}^{2} r_{lj} \leqslant 1$	1
$\mathcal{M}_j$	$(\neg p_{1j} \vee \neg p_{3j})$, $(\neg p_{2j} \vee \neg p_{3j})$, $(r_{1j} \vee p_{1j})$, $(r_{2j} \vee p_{2j})$, $\sum_{l=1}^{2} r_{lj} \leqslant 1$	(p_{3j})	$(r_{3j} \vee p_{3j})$	$\sum_{l=1}^{3} r_{lj} \leqslant 2$	1
		$\cdots$ ($m - 3$ times)			

DRE+MaxSAT for PHP in polynomial time

Constraint	Hard clause(s)	Soft clause(s)	Relaxed clauses	Updated AtMostk Constraints	LB increase
$\mathcal{L}_i$	$(\neg n_{i1} \vee \ldots \vee \neg n_{im})$	$(n_{i1}), \ldots, (n_{im})$	$(r_{il} \vee n_{il})$, $1 \leqslant l \leqslant m$	$\sum_{l=1}^{m} r_{il} \leqslant 1$	1
$\mathcal{M}_j$	$(\neg p_{1j} \vee \neg p_{2j})$	$(p_{1j}), (p_{2j})$	$(r_{1j} \vee p_{1j})$, $(r_{2j} \vee p_{2j})$	$\sum_{l=1}^{2} r_{lj} \leqslant 1$	1
	$(\neg p_{1j} \vee \neg p_{3j})$, $(\neg p_{2j} \vee \neg p_{3j})$, $(r_{1j} \vee p_{1j})$, $(r_{2j} \vee p_{2j})$, $\sum_{l=1}^{2} r_{lj} \leqslant 1$	(p_{3j})	$(r_{3j} \vee p_{3j})$	$\sum_{l=1}^{3} r_{lj} \leqslant 2$	1
	$\cdots$ $(m-3$ times$)$				
	$(\neg p_{1j} \vee \neg p_{m+1j}), \ldots,$ $(\neg p_{mj} \vee \neg p_{m+1j})$, $(r_{1j} \vee p_{1j}), \ldots,$ $(r_{mj} \vee p_{mj})$, $\sum_{l=1}^{m} r_{lj} \leqslant m-1$	(p_{m+1j})	$(r_{m+1j} \vee p_{m+1j})$	$\sum_{l=1}^{m+1} r_{lj} \leqslant m$	1

DRE+MaxSAT for PHP — unit propagation steps in $\mathcal{M}_j$

Clauses	Unit Propagation
(p_{k+1j})	$p_{k+1j} = 1$
$(\neg p_{1j} \vee \neg p_{k+1j}), \ldots, (\neg p_{kj} \vee \neg p_{k+1j})$	$p_{1j} = \ldots = p_{kj} = 0$
$(r_{1j} \vee p_{1j}), \ldots, (r_{kj} \vee p_{kj})$	$r_{1j} = \ldots = r_{kj} = 1$
$\sum_{l=1}^{k} r_{lj} \leqslant k-1$	$\left(\sum_{l=1}^{k} r_{lj} \leqslant k-1\right) \vdash_1 \bot$

DRE+MaxSAT for PHP — unit propagation steps in $\mathcal{M}_j$

Clauses	Unit Propagation
(p_{k+1j})	$p_{k+1j} = 1$
$(\neg p_{1j} \vee \neg p_{k+1j}), \ldots, (\neg p_{kj} \vee \neg p_{k+1j})$	$p_{1j} = \ldots = p_{kj} = 0$
$(r_{1j} \vee p_{1j}), \ldots, (r_{kj} \vee p_{kj})$	$r_{1j} = \ldots = r_{kj} = 1$
$\sum_{l=1}^{k} r_{lj} \leqslant k - 1$	$\left(\sum_{l=1}^{k} r_{lj} \leqslant k - 1\right) \vdash_1 \bot$

DRE+MaxSAT for PHP — unit propagation steps in $\mathcal{M}_j$

Clauses	Unit Propagation
(p_{k+1j})	$p_{k+1j} = 1$
$(\neg p_{1j} \vee \neg p_{k+1j}), \ldots, (\neg p_{kj} \vee \neg p_{k+1j})$	$p_{1j} = \ldots = p_{kj} = 0$
$(r_{1j} \vee p_{1j}), \ldots, (r_{kj} \vee p_{kj})$	$r_{1j} = \ldots = r_{kj} = 1$
$\sum_{l=1}^{k} r_{lj} \leqslant k-1$	$\left(\sum_{l=1}^{k} r_{lj} \leqslant k-1\right) \vdash_1 \bot$

DRE+MaxSAT for PHP — unit propagation steps in $\mathcal{M}_j$

Clauses	Unit Propagation
(p_{k+1j})	$p_{k+1j} = 1$
$(\neg p_{1j} \vee \neg p_{k+1j}), \ldots, (\neg p_{kj} \vee \neg p_{k+1j})$	$p_{1j} = \ldots = p_{kj} = 0$
$(r_{1j} \vee p_{1j}), \ldots, (r_{kj} \vee p_{kj})$	$r_{1j} = \ldots = r_{kj} = 1$
$\sum_{l=1}^{k} r_{lj} \leqslant k - 1$	$\left(\sum_{l=1}^{k} r_{lj} \leqslant k - 1\right) \vdash_1 \bot$

DRE+MaxSAT for PHP — unit propagation steps in $\mathcal{M}_j$

Clauses	Unit Propagation
$(p_{k+1\,j})$	$p_{k+1\,j} = 1$
$(\neg p_{1j} \vee \neg p_{k+1\,j}), \ldots, (\neg p_{kj} \vee \neg p_{k+1\,j})$	$p_{1j} = \ldots = p_{kj} = 0$
$(r_{1j} \vee p_{1j}), \ldots, (r_{kj} \vee p_{kj})$	$r_{1j} = \ldots = r_{kj} = 1$
$\sum_{l=1}^{k} r_{lj} \leqslant k-1$	$\left(\sum_{l=1}^{k} r_{lj} \leqslant k-1\right) \vdash_1 \bot$

DRE+MaxSAT for PHP — unit propagation steps in $\mathcal{M}_j$

Clauses	Unit Propagation
(p_{k+1j})	$p_{k+1j} = 1$
$(\neg p_{1j} \vee \neg p_{k+1j}), \ldots, (\neg p_{kj} \vee \neg p_{k+1j})$	$p_{1j} = \ldots = p_{kj} = 0$
$(r_{1j} \vee p_{1j}), \ldots, (r_{kj} \vee p_{kj})$	$r_{1j} = \ldots = r_{kj} = 1$
$\sum_{l=1}^{k} r_{lj} \leqslant k-1$	$\left(\sum_{l=1}^{k} r_{lj} \leqslant k-1\right) \vdash_1 \bot$

DRE+MaxSAT for PHP — unit propagation steps in $\mathcal{M}_j$

Clauses	Unit Propagation
(p_{k+1j})	$p_{k+1j} = 1$
$(\neg p_{1j} \vee \neg p_{k+1j}), \ldots, (\neg p_{kj} \vee \neg p_{k+1j})$	$p_{1j} = \ldots = p_{kj} = 0$
$(r_{1j} \vee p_{1j}), \ldots, (r_{kj} \vee p_{kj})$	$r_{1j} = \ldots = r_{kj} = 1$
$\sum_{l=1}^{k} r_{lj} \leqslant k - 1$	$\left(\sum_{l=1}^{k} r_{lj} \leqslant k - 1\right) \vdash_1 \bot$

DRE+MaxSAT for PHP — unit propagation steps in $\mathcal{M}_j$

Clauses	Unit Propagation
(p_{k+1j})	$p_{k+1j} = 1$
$(\neg p_{1j} \vee \neg p_{k+1j}), \ldots, (\neg p_{kj} \vee \neg p_{k+1j})$	$p_{1j} = \ldots = p_{kj} = 0$
$(r_{1j} \vee p_{1j}), \ldots, (r_{kj} \vee p_{kj})$	$r_{1j} = \ldots = r_{kj} = 1$
$\sum_{l=1}^{k} r_{lj} \leqslant k - 1$	$\left(\sum_{l=1}^{k} r_{lj} \leqslant k - 1\right) \vdash_1 \bot$

DRE+MaxSAT for PHP — unit propagation steps in $\mathcal{M}_j$

Clauses	Unit Propagation
(p_{k+1j})	$p_{k+1j} = 1$
$(\neg p_{1j} \vee \neg p_{k+1j}), \ldots, (\neg p_{kj} \vee \neg p_{k+1j})$	$p_{1j} = \ldots = p_{kj} = 0$
$(r_{1j} \vee p_{1j}), \ldots, (r_{kj} \vee p_{kj})$	$r_{1j} = \ldots = r_{kj} = 1$
$\sum_{l=1}^{k} r_{lj} \leqslant k - 1$	$\left(\sum_{l=1}^{k} r_{lj} \leqslant k - 1\right) \vdash_1 \bot$

DRE+MaxSAT for PHP — unit propagation steps in $\mathcal{M}_j$

Clauses	Unit Propagation
(p_{k+1j})	$p_{k+1j} = 1$
$(\neg p_{1j} \vee \neg p_{k+1j}), \ldots, (\neg p_{kj} \vee \neg p_{k+1j})$	$p_{1j} = \ldots = p_{kj} = 0$
$(r_{1j} \vee p_{1j}), \ldots, (r_{kj} \vee p_{kj})$	$r_{1j} = \ldots = r_{kj} = 1$
$\sum_{l=1}^{k} r_{lj} \leqslant k-1$	$\left(\sum_{l=1}^{k} r_{lj} \leqslant k-1\right) \vdash_1 \bot$

- **Key points**:
  - for each $\mathcal{L}_i$, UP raises LB by 1

DRE+MaxSAT for PHP — unit propagation steps in $\mathcal{M}_j$

Clauses	Unit Propagation
(p_{k+1j})	$p_{k+1j} = 1$
$(\neg p_{1j} \vee \neg p_{k+1j}), \ldots, (\neg p_{kj} \vee \neg p_{k+1j})$	$p_{1j} = \ldots = p_{kj} = 0$
$(r_{1j} \vee p_{1j}), \ldots, (r_{kj} \vee p_{kj})$	$r_{1j} = \ldots = r_{kj} = 1$
$\sum_{l=1}^{k} r_{lj} \leqslant k - 1$	$\left(\sum_{l=1}^{k} r_{lj} \leqslant k - 1\right) \vdash_1 \bot$

- **Key points**:
  - for each $\mathcal{L}_i$, UP raises LB by 1
  - for each $\mathcal{M}_j$, UP raises LB by m

DRE+MaxSAT for PHP — unit propagation steps in $\mathcal{M}_j$

Clauses	Unit Propagation
(p_{k+1j})	$p_{k+1j} = 1$
$(\neg p_{1j} \vee \neg p_{k+1j}), \ldots, (\neg p_{kj} \vee \neg p_{k+1j})$	$p_{1j} = \ldots = p_{kj} = 0$
$(r_{1j} \vee p_{1j}), \ldots, (r_{kj} \vee p_{kj})$	$r_{1j} = \ldots = r_{kj} = 1$
$\sum_{l=1}^{k} r_{lj} \leqslant k-1$	$\left(\sum_{l=1}^{k} r_{lj} \leqslant k-1\right) \vdash_1 \bot$

- **Key points**:
  - for each $\mathcal{L}_i$, UP raises LB by 1
  - for each $\mathcal{M}_j$, UP raises LB by m
  - in total, UP raises LB by $m \cdot (m+1) + 1$

DRE+MaxSAT for PHP — unit propagation steps in $\mathcal{M}_j$

Clauses	Unit Propagation
(p_{k+1j})	$p_{k+1j} = 1$
$(\neg p_{1j} \vee \neg p_{k+1j}), \ldots, (\neg p_{kj} \vee \neg p_{k+1j})$	$p_{1j} = \ldots = p_{kj} = 0$
$(r_{1j} \vee p_{1j}), \ldots, (r_{kj} \vee p_{kj})$	$r_{1j} = \ldots = r_{kj} = 1$
$\sum_{l=1}^{k} r_{lj} \leqslant k-1$	$\left(\sum_{l=1}^{k} r_{lj} \leqslant k-1\right) \vdash_1 \bot$

- **Key points**:
  - for each $\mathcal{L}_i$, UP raises LB by 1
  - for each $\mathcal{M}_j$, UP raises LB by m
  - in total, UP raises LB by $m \cdot (m+1) + 1$
  - PHP_m^{m+1} is ***unsatisfiable***

Short MaxSAT proof for PHP

short DRE+MaxSAT-resolution proof

⬇

see the paper!

Experimental results

Experimental evaluation

- approaches tested:
  1. **SAT** (MiniSat 2.2, *Glucose 3.0*)
  2. **SAT+** (*lingeling*, CryptoMiniSat)

Experimental evaluation

- approaches tested:
  1. **SAT** (MiniSat 2.2, *Glucose 3.0*)
  2. **SAT+** (*lingeling*, CryptoMiniSat)
  3. **IHS MaxSAT** (*MaxHS*, *LMHS*)

Experimental evaluation

- approaches tested:
  1. SAT (MiniSat 2.2, *Glucose 3.0*)
  2. SAT+ (*lingeling*, CryptoMiniSat)
  3. IHS MaxSAT (*MaxHS*, *LMHS*)
  4. CG MaxSAT (*MSCG*, *OpenWBO16*, WPM3)

Experimental evaluation

- approaches tested:
  1. **SAT** (MiniSat 2.2, *Glucose 3.0*)
  2. **SAT+** (*lingeling*, CryptoMiniSat)
  3. **IHS MaxSAT** (*MaxHS*, *LMHS*)
  4. **CG MaxSAT** (*MSCG*, *OpenWBO16*, WPM3)
  5. **MaxRes** (*Eva*)

Experimental evaluation

- approaches tested:
  1. SAT (MiniSat 2.2, *Glucose 3.0*)
  2. SAT+ (*lingeling*, CryptoMiniSat)
  3. IHS MaxSAT (*MaxHS*, *LMHS*)
  4. CG MaxSAT (*MSCG*, *OpenWBO16*, WPM3)
  5. MaxRes (*Eva*)
  6. MIP (*CPLEX*)

Experimental evaluation

- approaches tested:
    1. SAT (MiniSat 2.2, *Glucose 3.0*)
    2. SAT+ (*lingeling*, CryptoMiniSat)
    3. IHS MaxSAT (*MaxHS*, *LMHS*)
    4. CG MaxSAT (*MSCG*, *OpenWBO16*, WPM3)
    5. MaxRes (*Eva*)
    6. MIP (*CPLEX*)
    7. OPB (*cdcl-cuttingplanes*, Sat4j)

Experimental evaluation

- approaches tested:
  1. SAT (MiniSat 2.2, *Glucose 3.0*)
  2. SAT+ (*lingeling*, CryptoMiniSat)
  3. IHS MaxSAT (*MaxHS*, *LMHS*)
  4. CG MaxSAT (*MSCG*, *OpenWBO16*, WPM3)
  5. MaxRes (*Eva*)
  6. MIP (*CPLEX*)
  7. OPB (*cdcl-cuttingplanes*, Sat4j)
  8. BDD (*ZRes*)

Experimental evaluation

- approaches tested:
  1. **SAT** (MiniSat 2.2, *Glucose 3.0*)
  2. **SAT+** (*lingeling*, CryptoMiniSat)
  3. **IHS MaxSAT** (*MaxHS*, *LMHS*)
  4. **CG MaxSAT** (*MSCG*, *OpenWBO16*, WPM3)
  5. **MaxRes** (*Eva*)
  6. **MIP** (*CPLEX*)
  7. **OPB** (*cdcl-cuttingplanes*, Sat4j)
  8. **BDD** (*ZRes*)

- Machine configuration:
  - Intel Xeon E5-2630 2.60GHz with 64GByte RAM

Experimental evaluation

- approaches tested:
  1. **SAT** (MiniSat 2.2, *Glucose 3.0*)
  2. **SAT+** (*lingeling*, CryptoMiniSat)
  3. **IHS MaxSAT** (*MaxHS*, *LMHS*)
  4. **CG MaxSAT** (*MSCG*, *OpenWBO16*, WPM3)
  5. **MaxRes** (*Eva*)
  6. **MIP** (*CPLEX*)
  7. **OPB** (*cdcl-cuttingplanes*, Sat4j)
  8. **BDD** (*ZRes*)

- Machine configuration:
  - Intel Xeon E5-2630 2.60GHz with 64GByte RAM
  - running Ubuntu Linux

Experimental evaluation

- approaches tested:
    1. **SAT** (MiniSat 2.2, *Glucose 3.0*)
    2. **SAT+** (*lingeling*, CryptoMiniSat)
    3. **IHS MaxSAT** (*MaxHS*, *LMHS*)
    4. **CG MaxSAT** (*MSCG*, *OpenWBO16*, WPM3)
    5. **MaxRes** (*Eva*)
    6. **MIP** (*CPLEX*)
    7. **OPB** (*cdcl-cuttingplanes*, Sat4j)
    8. **BDD** (*ZRes*)
- Machine configuration:
    - Intel Xeon E5-2630 2.60GHz with 64GByte RAM
    - running Ubuntu Linux
    - 1800s timeout

Experimental evaluation

- approaches tested:
  1. **SAT** (MiniSat 2.2, *Glucose 3.0*)
  2. **SAT+** (*lingeling*, CryptoMiniSat)
  3. **IHS MaxSAT** (*MaxHS*, *LMHS*)
  4. **CG MaxSAT** (*MSCG*, *OpenWBO16*, WPM3)
  5. **MaxRes** (*Eva*)
  6. **MIP** (*CPLEX*)
  7. **OPB** (*cdcl-cuttingplanes*, Sat4j)
  8. **BDD** (*ZRes*)

- Machine configuration:
  - Intel Xeon E5-2630 2.60GHz with 64GByte RAM
  - running Ubuntu Linux
  - 1800s timeout
  - 10GByte memout

Experimental evaluation

- benchmarks:
  1. PHP (pigeonhole principle):
     - PHP_m^{m+1}, $m \in \{4, \ldots, 100\}$
     - pairwise — 46 instances
     - sequential counter — 46 instances

[1] P. Chatalic and L. Simon. Multiresolution for SAT checking. *International Journal on Artificial Intelligence Tools*, 10(4):451–481, 2001.

Experimental evaluation

- benchmarks:
  1. PHP (pigeonhole principle):
     - PHP_m^{m+1} , $m \in \{4, \ldots, 100\}$
     - pairwise — 46 instances
     - sequential counter — 46 instances
  2. URQ (Urquhart formulas[1]):
     - $\text{URQ}_{n,i}$, $n \in \{3, \ldots, 30\}$, $i \in \{1, 2, 3\}$
     - 84 instances

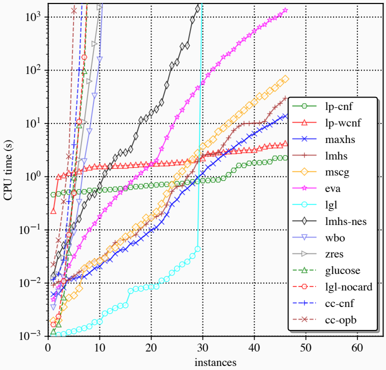
[1] P. Chatalic and L. Simon. Multiresolution for SAT checking. *International Journal on Artificial Intelligence Tools*, 10(4):451–481, 2001.

Experimental evaluation

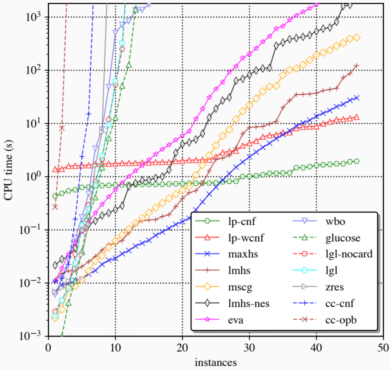
- benchmarks:
  1. PHP (pigeonhole principle):
     - PHP_m^{m+1}, $m \in \{4, \ldots, 100\}$
     - pairwise — 46 instances
     - sequential counter — 46 instances
  2. URQ (Urquhart formulas[1]):
     - $\mathrm{URQ}_{n,i}$, $n \in \{3, \ldots, 30\}$, $i \in \{1, 2, 3\}$
     - 84 instances
  3. COMB (combined):
     - $\mathrm{PHP}_m^{m+1} \vee \mathrm{URQ}_{n,i}$, $m \in \{7, 9, 11, 13\}$, $n \in \{3, \ldots, 10\}$, $i \in \{1, 2, 3\}$
     - 96 instances

[1] P. Chatalic and L. Simon. Multiresolution for SAT checking. *International Journal on Artificial Intelligence Tools*, 10(4):451–481, 2001.

Performance on pigeonhole formulas



(a) PHP-pw *(pairwise)*



(b) PHP-sc *(sequential counter)*

$\mathcal{P}$ clauses can be harmful

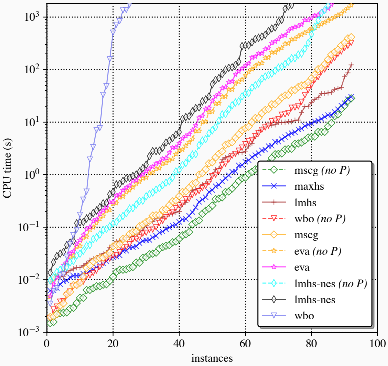
$(\neg p_i \vee \neg n_i, \top)$

$\mathcal{P}$ clauses can be harmful

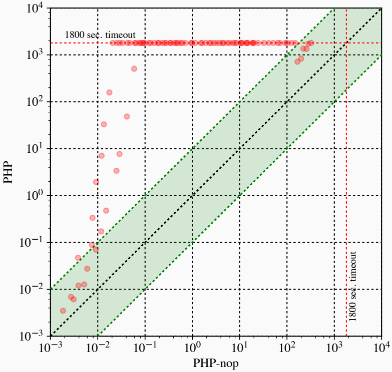
$(\neg p_i \vee \neg n_i, \top) \wedge (p_i, 1) \wedge (n_i, 1)$ — trivial core

$\mathcal{P}$ clauses can be harmful

$(\neg p_i \vee \neg n_i, \top) \wedge (p_i, 1) \wedge (n_i, 1)$ — trivial core

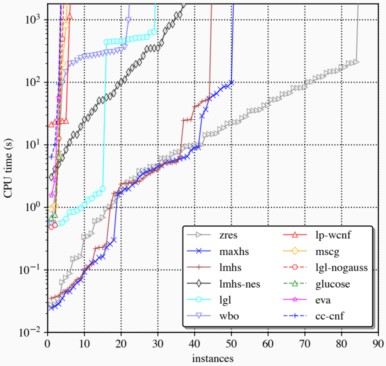


(a) cactus plot

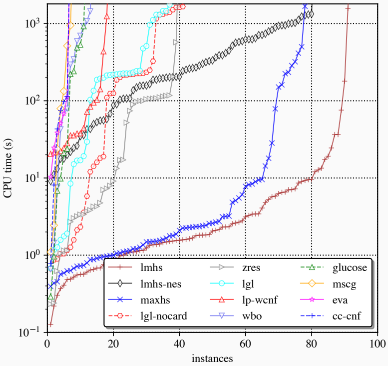


(b) *wbo* w/ and w/o $\mathcal{P}$ clauses

Performance on Urquhart and combined formulas



(a) URQ instances



(b) COMB instances

Overall performance

		glucose	lgl	lgl-no[2]	maxhs	lmhs	lmhs-nes	mscg	wbo	eva	lp-cnf	lp-wcnf	cc-cnf	cc-opb	zres
PHP-pw	(46)	7	29	7	***46***	***46***	29	***46***	10	***46***	***46***	***46***	6	5	10
PHP-sc	(46)	13	11	11	***46***	***46***	45	***46***	15	40	***46***	***46***	6	2	8
URQ	(84)	3	29	4	50	44	37	5	22	3	0	6	3	0	***84***
COMB	(96)	11	37	41	78	***91***	80	7	13	6	0	18	6	0	39
Total	(272)	34	106	63	220	***227***	191	104	60	95	92	116	21	7	141

[2] This represents *lgl-nogauss* for URQ and *lgl-nocard* for PHP-pw, PHP-sc, and COMB.

Summary and future work

Summary and future work

- solving SAT with DRE+MaxSAT

Summary and future work

- solving SAT with DRE+MaxSAT
  - dual-rail (Horn) encoding

Summary and future work

- solving SAT with DRE+MaxSAT
  - dual-rail (Horn) encoding
  - apply MaxSAT technology:
    - core-guided reasoning
    - MaxSAT resolution

Summary and future work

- solving SAT with DRE+MaxSAT
  - dual-rail (Horn) encoding
  - apply MaxSAT technology:
    - core-guided reasoning
    - MaxSAT resolution
  - refuting PHP in polynomial time
  - more examples (Urquhart and combined)

Summary and future work

- solving SAT with DRE+MaxSAT
    - dual-rail (Horn) encoding
    - apply MaxSAT technology:
        - core-guided reasoning
        - MaxSAT resolution
    - refuting PHP in polynomial time
    - more examples (Urquhart and combined)
- more easy examples?

Summary and future work

- solving SAT with DRE+MaxSAT
  - dual-rail (Horn) encoding
  - apply MaxSAT technology:
    - core-guided reasoning
    - MaxSAT resolution
  - refuting PHP in polynomial time
  - more examples (Urquhart and combined)

- more easy examples? any hard examples?

Summary and future work

- solving SAT with DRE+MaxSAT
  - dual-rail (Horn) encoding
  - apply MaxSAT technology:
    - core-guided reasoning
    - MaxSAT resolution
  - refuting PHP in polynomial time
  - more examples (Urquhart and combined)

- more easy examples? any hard examples?
- similar transformations for other hard examples?

Summary and future work

- solving SAT with DRE+MaxSAT
    - dual-rail (Horn) encoding
    - apply MaxSAT technology:
        - core-guided reasoning
        - MaxSAT resolution
    - refuting PHP in polynomial time
    - more examples (Urquhart and combined)

- more easy examples? any hard examples?
- similar transformations for other hard examples?
- integrating in a SAT solver?

Summary and future work

- solving SAT with DRE+MaxSAT
    - dual-rail (Horn) encoding
    - apply MaxSAT technology:
        - core-guided reasoning
        - MaxSAT resolution
    - refuting PHP in polynomial time
    - more examples (Urquhart and combined)

- more easy examples? any hard examples?
- similar transformations for other hard examples?
- integrating in a SAT solver?
- relation to known proof systems (TLR/GR/ER, CP, etc.)?

Summary and future work

- solving SAT with DRE+MaxSAT
  - dual-rail (Horn) encoding
  - apply MaxSAT technology:
    - core-guided reasoning
    - MaxSAT resolution
  - refuting PHP in polynomial time
  - more examples (Urquhart and combined)

- more easy examples? any hard examples?
- similar transformations for other hard examples?
- integrating in a SAT solver?
- relation to known proof systems (TLR/GR/ER, CP, etc.)?
- why is IHS so *good*?

Questions?