25 years of SAT
Maximum Satisfiability for Real-World Optimization

Jeremias Berg

HIIT, Department of Computer Science, University of Helsinki, Finland

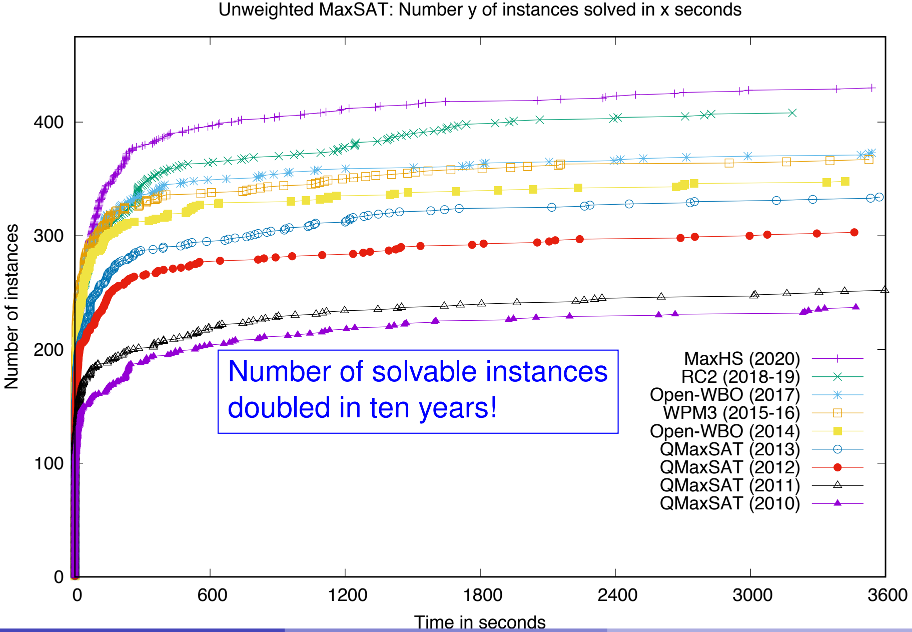
August 4
SAT@FLoC 2022
Haifa

Motivation: MaxSAT

- Competitive optimization paradigm.
  - planning, scheduling, data analysis, machine learning, knowledge representation and reasoning, verification, ...
- State-of-the-art complete solvers (for industrial instances) build on the success of CDCL SAT solvers.
  - Optimization task reduced into a sequence of satisfiability queries.
  - Extensive use of incremental SAT.
- New application domains and solver improvements annually.

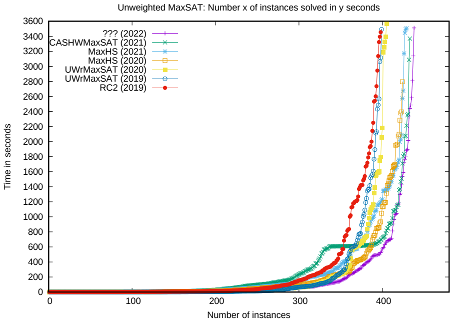
  (Recent survey in [Bacchus, Järvisalo, and Martins, 2021])

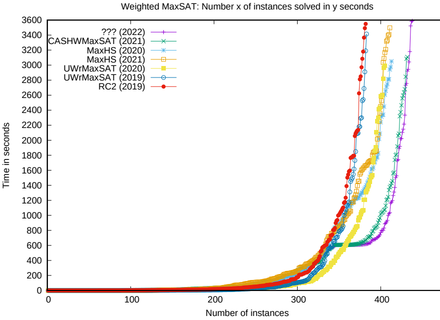
Solver Progression



Number of solvable instances doubled in ten years!

Solver Progression





Goals of the Talk

- Overview of existing *complete* SAT-based MaxSAT algorithms.
  - Solution Improving
  - Core Guided
  - Implicit Hitting Set
- Overview of the additional techniques employed by solvers.
- (If time) Other interesting MaxSAT-related research directions.

Thanks to Ruben Martins and Matti Järvisalo for contributions to the slides

Notation: Maximum Satisfiability

- Optimisation extension of Boolean Satisfiability (SAT)
- An instance:
  - a set of hard clauses,
  - a linear objective function *cost*
- Find τ that:
  - satisfies all hard clauses and
  - minimizes cost
- **Note:** All SAT-based solvers support weights.

Notation: Maximum Satisfiability

- Optimisation extension of Boolean Satisfiability (SAT)
- An instance:
  - a set of hard clauses,
  - a linear objective function *cost*
  - equivalent to a set of (unit) soft clauses.
- Find τ that:
  - satisfies all hard clauses and
  - minimizes cost
- **Note:** All SAT-based solvers support weights.

$$\mathcal{F}_H = \{(b_1 \lor x), (\neg x \lor b_2), (b_2 \lor y), (\neg y, b_3)\}$$

$$cost \equiv b_1 + b_2 + b_3$$

$$\mathcal{F}_S = \{(\neg b_1), (\neg b_2), (\neg b_3)\}$$

Notation: Maximum Satisfiability

- Optimisation extension of Boolean Satisfiability (SAT)
- An instance:
  - a set of hard clauses,
  - a linear objective function *cost*
- Find τ that:
  - satisfies all hard clauses and
  - minimizes cost
- **Note:** All SAT-based solvers support weights.

$$\boxed{\begin{array}{l}\tau(y) = \tau(b_1) = \tau(b_3) = 0 \\ \tau(x) = \tau(b_2) = 1\end{array}}$$

$$\mathcal{F}_H = \{(b_1 \vee x), (\neg x \vee b_2), (b_2 \vee y), (\neg y, b_3)\}$$

$$cost \equiv b_1 + b_2 + b_3$$

$$cost(\tau) = 1$$

Notation: Maximum Satisfiability

- Optimisation extension of Boolean Satisfiability (SAT)
- An instance:
  - a set of hard clauses,
  - a linear objective function *cost*
- Find τ that:
  - satisfies all hard clauses and
  - minimizes cost
- **Note:** All SAT-based solvers support weights.

$$\mathcal{F}_H = \{(b_1 \lor x), (\neg x \lor b_2), (b_2 \lor y), (\neg y, b_3)\}$$

$$cost \equiv w_1 b_1 + w_2 b_2 + w_3 b_3$$

Example Problem

Shortest Path

- find shortest path from S to G.
- one variable for each square.
  - true if path goes through square.
- (hard) clauses:
  - enforce correspondence between solutions and paths.
- cost function:
  - measures length of path.

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

Note: Best solved with other methods, used here to illustrate different MaxSAT algorithms.

Example Problem

Shortest Path

- find shortest path from S to G.
- one variable for each square.
  - true if path goes through square.
- (hard) clauses:
  - enforce correspondence between solutions and paths.
- cost function:
  - measures length of path.

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

$$\text{VAR}(\mathcal{F}) = \{a, b, c, \ldots, r, s, t\}$$

Note: Best solved with other methods, used here to illustrate different MaxSAT algorithms.

Example Problem

Shortest Path

- find shortest path from S to G.
- one variable for each square.
  - true if path goes through square.
- (hard) clauses:
  - enforce correspondence between solutions and paths.
- cost function:
  - measures length of path.

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

$$\textsc{var}(\mathcal{F}) = \{a, b, c, \ldots, r, s, t\}$$

$$\mathcal{F}_H = \textsc{IsPath}(S, G)$$

Note: Best solved with other methods, used here to illustrate different MaxSAT algorithms.

Example Problem

Shortest Path

- find shortest path from S to G.
- one variable for each square.
  - true if path goes through square.
- (hard) clauses:
  - enforce correspondence between solutions and paths.
- cost function:
  - measures length of path.

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

$$\textsc{Var}(\mathcal{F}) = \{a, b, c, \ldots, r, s, t\}$$

$$\mathcal{F}_H = \textsc{IsPath}(S, G)$$

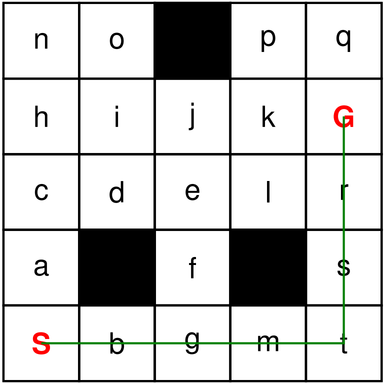
$$cost = a + b + c + \ldots + r + s + t$$
$$cost = \{(\neg a), (\neg b), \ldots, (\neg s), (\neg t)\}$$

Note: Best solved with other methods, used here to illustrate different MaxSAT algorithms.

Example Problem

Shortest Path

- find shortest path from S to G.
- one variable for each square.
  - true if path goes through square.
- (hard) clauses:
  - enforce correspondence between solutions and paths.
- cost function:
  - measures length of path.



$$\tau = \{b, g, m, t, s, r\} \cup \{\neg a, \neg c, ..\}$$

$$cost = a + b + c + \ldots + r + s + t = 6$$
$$cost = \{(\neg a), (\neg b), \ldots, (\neg s), (\neg t)\}$$

Note: Best solved with other methods, used here to illustrate different MaxSAT algorithms.

Solution Improving Search

Solution Improving Search

Intuition

1. Obtain a solution τ^*
2. Update UB
3. Improve τ^* until proven optimal

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

UB $= \infty$

Solution Improving Search

Intuition

1. Obtain a solution τ^*
2. Update UB
3. Improve τ^* until proven optimal

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

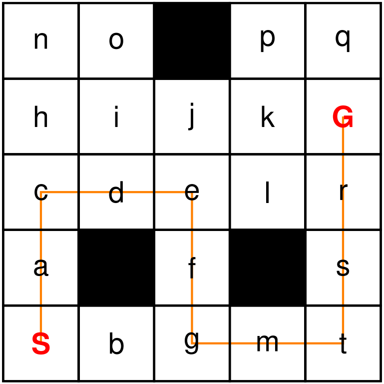
UB $= \infty$

SAT-SOLVE($\mathcal{F}_\mathbf{H}$)

Solution Improving Search

Intuition

1. Obtain a solution τ^*
2. Update UB
3. Improve τ^* until proven optimal



$\mathbf{UB = 10}$

SAT-SOLVE$(\mathcal{F}_H)$

$\tau^1 = \{a, c, d, e, f, g, m, u, t, r, \neg b, \neg l, \ldots, \neg q\}$

$cost(\tau^1) = 10$

Solution Improving Search

Intuition

1. Obtain a solution τ^*
2. Update UB
3. Improve τ^* until proven optimal

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

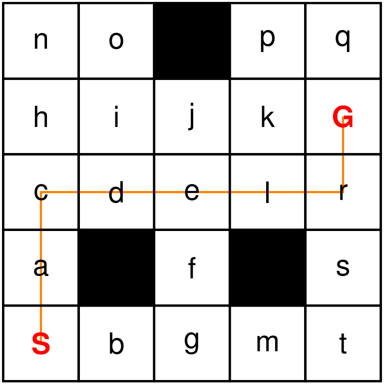
UB $= 10$

SAT-SOLVE $(\mathcal{F}_{\mathbf{H}} \wedge \textsc{CostLessThan}(\mathbf{UB}))$

Solution Improving Search

Intuition

1. Obtain a solution τ^*
2. Update UB
3. Improve τ^* until proven optimal



UB = 6

SAT-SOLVE $(\mathcal{F}_H \wedge \textsc{CostLessThan}(UB))$

$\tau^2 = \{a, c, d, e, l, r, \neg b, \neg g, \ldots, \neg q, \neg s, \neg t\}$
$cost(\tau^2) = 6$

Additional Techniques

Central Challenge

- Encoding CostLessThan(UB) as clauses.

- State of the art use the watchdog encoding [Paxian, Reimer, and Becker, 2018].
- Additional inference rules for decreasing the size of the objective.
  - e.g TrimMaxSAT for finding assignments of objective literals implied by the clauses [Paxian, Raiola, and Becker, 2021].

Solvers

QMaxSAT [Koshimura, Zhang, Fujita, and Hasegawa, 2012]
Pacose [Paxian, Reimer, and Becker, 2018]
Also commonly applied in *incomplete solving*.

Additional Techniques

Central Challenge

- Encoding CostLessThan(UB) as clauses.

- State of the art use the watchdog encoding [Paxian, Reimer, and Becker, 2018].
- Additional inference rules for decreasing the size of the objective.
  - e.g TrimMaxSAT for finding assignments of objective literals implied by the clauses [Paxian, Raiola, and Becker, 2021].

Solvers

QMaxSAT [Koshimura, Zhang, Fujita, and Hasegawa, 2012]
Pacose [Paxian, Reimer, and Becker, 2018]
Also commonly applied in *incomplete solving*.

Additional Techniques

Central Challenge

- Encoding CostLessThan(UB) as clauses.

- State of the art use the watchdog encoding [Paxian, Reimer, and Becker, 2018].
- Additional inference rules for decreasing the size of the objective.
  - e.g TrimMaxSAT for finding assignments of objective literals implied by the clauses [Paxian, Raiola, and Becker, 2021].

Solvers

QMaxSAT [Koshimura, Zhang, Fujita, and Hasegawa, 2012]
Pacose [Paxian, Reimer, and Becker, 2018]
Also commonly applied in *incomplete solving*.

Core Guided Search

Core Guided Search

First proposed for MaxSAT in [Fu and Malik, 2006]

Intuition

1. Starting from $LB = 0$ check existence of solution τ for which $\mathit{cost}(\tau) = LB$.
2. Increase LB until optimum reached by relaxing formula.
3. Use cores provided by SAT-solver for more effective relaxation.

Core Guided Search

First proposed for MaxSAT in [Fu and Malik, 2006]

Intuition

1. Starting from $LB = 0$ check existence of solution τ for which $cost(\tau) = LB$.
2. Increase LB until optimum reached by relaxing formula.
3. Use cores provided by SAT-solver for more effective relaxation.

Definition: UNSAT cores

- Clause (or set of) of objective literals satisfied by all solutions.
  - equivalent to an unsatisfiable set of unit soft clauses.
- Can be obtained from SAT solvers via assumptions.

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

$\kappa^1 = \{a, b\} \equiv (a \vee b) \equiv \{(\neg a), (\neg b)\}$
all paths go through either a or b

$\kappa^2 = \{h, d, f, m\}$
all paths go through (at least) one of h, d, f or m.

$\kappa^3 = \{q, k, r\}$
all paths go through (at least) one of q, k or r.

$\vdots$

Core Guided Search

Shortest path

Intuition

1. Initialise $\mathcal{F}_H^0 = \mathcal{F}_H$ and $\mathcal{F}_B^0 = \{\neg b \mid b \in cost\}$
2. For $i = 0, \ldots$ check if $\mathcal{F}_H^i \wedge \mathcal{F}_B^i$ is satisfiable
3. If not, relax $\mathcal{F}_H^i$ and $\mathcal{F}_B^i$
4. Otherwise, the obtained solution is optimal

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

Core Guided Search

Shortest path

Intuition

1. Initialise $\mathcal{F}_H^0 = \mathcal{F}_H$ and $\mathcal{F}_B^0 = \{\neg b \mid b \in cost\}$
2. For $i = 0, \ldots$ check if $\mathcal{F}_H^i \wedge \mathcal{F}_B^i$ is satisfiable
3. If not, relax $\mathcal{F}_H^i$ and $\mathcal{F}_B^i$
4. Otherwise, the obtained solution is optimal

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

$\mathrm{LB} = 0$, $\mathcal{K} = \emptyset$

SAT-SOLVE($\mathcal{F}_{\mathbf{H}}^{\mathbf{i}} \wedge \mathcal{F}_{\mathbf{B}}^{\mathbf{i}}$)

Informally: $\mathcal{F}_H \wedge \bigwedge_{\kappa \in \mathcal{K}} \sum_{b \in \kappa} b \leq 1 \wedge \bigwedge_{b \notin \mathcal{K}} \neg b$

i.e. is there a path that visits at most 1 node from each found core

Core Guided Search

Shortest path

Intuition

1. Initialise $\mathcal{F}_H^0 = \mathcal{F}_H$ and $\mathcal{F}_B^0 = \{\neg b \mid b \in cost\}$
2. For $i = 0, \ldots$ check if $\mathcal{F}_H^i \wedge \mathcal{F}_B^i$ is satisfiable
3. If not, relax $\mathcal{F}_H^i$ and $\mathcal{F}_B^i$
4. Otherwise, the obtained solution is optimal

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

LB $= 0$, $\mathcal{K} = \emptyset$

SAT-SOLVE($\mathcal{F}_{\mathbf{H}}^{\mathbf{i}} \wedge \mathcal{F}_{\mathbf{B}}^{\mathbf{i}}$)

Formula is unsatisfiable

Obtain new core: $\kappa_0 = \{(\neg a), (\neg b)\}$

Core Guided Search

Shortest path

Intuition

1. Initialise $\mathcal{F}_H^0 = \mathcal{F}_H$ and $\mathcal{F}_B^0 = \{\neg b \mid b \in cost\}$
2. For $i = 0, \ldots$ check if $\mathcal{F}_H^i \wedge \mathcal{F}_B^i$ is satisfiable
3. If not, relax $\mathcal{F}_H^i$ and $\mathcal{F}_B^i$
4. Otherwise, the obtained solution is optimal

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

$\mathrm{LB} = 1, \mathcal{K} = \{\kappa_0\}$

SAT-SOLVE($\mathcal{F}_{\mathbf{H}}^{\mathbf{i}} \wedge \mathcal{F}_{\mathbf{B}}^{\mathbf{i}}$)

Informally: $\mathcal{F}_H \wedge \bigwedge_{\kappa \in \mathcal{K}} \sum_{b \in \kappa} b \leq 1 \wedge \bigwedge_{b \notin \mathcal{K}} \neg b$

i.e. is there a path that visits at most 1 node from each found core

Core Guided Search

Shortest path

Intuition

1. Initialise $\mathcal{F}_H^0 = \mathcal{F}_H$ and $\mathcal{F}_B^0 = \{\neg b \mid b \in cost\}$
2. For $i = 0, \ldots$ check if $\mathcal{F}_H^i \wedge \mathcal{F}_B^i$ is satisfiable
3. If not, relax $\mathcal{F}_H^i$ and $\mathcal{F}_B^i$
4. Otherwise, the obtained solution is optimal

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

LB $= 1$, $\mathcal{K} = \{\kappa_0\}$

SAT-SOLVE($\mathcal{F}_H^i \wedge \mathcal{F}_B^i$)

Formula is unsatisfiable

Obtain new core: $\kappa_1 = \{(\neg q), (\neg k), (\neg r)\}$

Core Guided Search

Shortest path

Intuition

1. Initialise $\mathcal{F}_H^0 = \mathcal{F}_H$ and $\mathcal{F}_B^0 = \{\neg b \mid b \in cost\}$
2. For $i = 0, \ldots$ check if $\mathcal{F}_H^i \wedge \mathcal{F}_B^i$ is satisfiable
3. If not, relax $\mathcal{F}_H^i$ and $\mathcal{F}_B^i$
4. Otherwise, the obtained solution is optimal

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

LB $= 2$, $\mathcal{K} = \{\kappa_0, \kappa_1\}$

SAT-SOLVE($\mathcal{F}_{\mathbf{H}}^{\mathbf{i}} \wedge \mathcal{F}_{\mathbf{B}}^{\mathbf{i}}$)

Informally: $\mathcal{F}_H \wedge \bigwedge_{\kappa \in \mathcal{K}} \sum_{b \in \kappa} b \leq 1 \wedge \bigwedge_{b \notin \mathcal{K}} \neg b$

i.e. is there a path that visits at most 1 node from each found core

Core Guided Search

Shortest path

Intuition

1. Initialise $\mathcal{F}_H^0 = \mathcal{F}_H$ and $\mathcal{F}_B^0 = \{\neg b \mid b \in cost\}$
2. For $i = 0, \ldots$ check if $\mathcal{F}_H^i \wedge \mathcal{F}_B^i$ is satisfiable
3. If not, relax $\mathcal{F}_H^i$ and $\mathcal{F}_B^i$
4. Otherwise, the obtained solution is optimal

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

LB $= 6$, $\mathcal{K} = \{\kappa_0, \kappa_1, \kappa_2, \kappa_3, \kappa_4, \kappa_5\}$

SAT-SOLVE($\mathcal{F}_{\mathbf{H}}^{\mathbf{i}} \wedge \mathcal{F}_{\mathbf{B}}^{\mathbf{i}}$)

Informally: $\mathcal{F}_H \wedge \bigwedge_{\kappa \in \mathcal{K}} \sum_{b \in \kappa} b \leq 1 \wedge \bigwedge_{b \notin \mathcal{K}} \neg b$

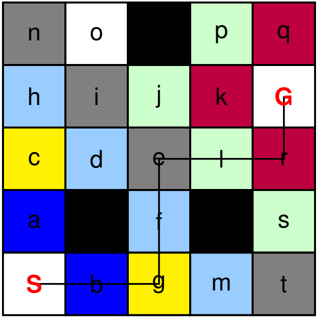
i.e. is there a path that visits at most 1 node from each found core

Core Guided Search

Shortest path

Intuition

1. Initialise $\mathcal{F}_H^0 = \mathcal{F}_H$ and $\mathcal{F}_B^0 = \{\neg b \mid b \in cost\}$
2. For $i = 0, \ldots$ check if $\mathcal{F}_H^i \wedge \mathcal{F}_B^i$ is satisfiable
3. If not, relax $\mathcal{F}_H^i$ and $\mathcal{F}_B^i$
4. Otherwise, the obtained solution is optimal



LB $= 6$, $\mathcal{K} = \{\kappa_0, \kappa_1, \kappa_2, \kappa_3, \kappa_4, \kappa_5\}$

SAT-SOLVE($\mathcal{F}_{\mathbf{H}}^{\mathbf{i}} \wedge \mathcal{F}_{\mathbf{B}}^{\mathbf{i}}$)

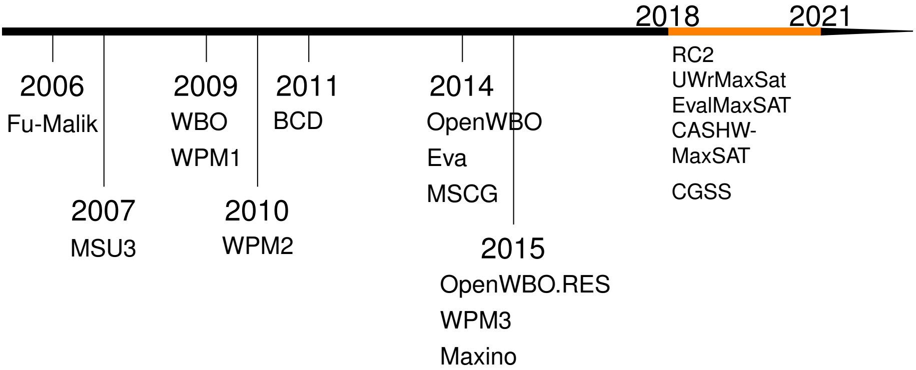
Formula is satisfiable

Obtain optimal model: $\tau = \{b, \ldots, l, r, \neg a, \ldots, \neg q\}$

$cost(\tau) = 6$

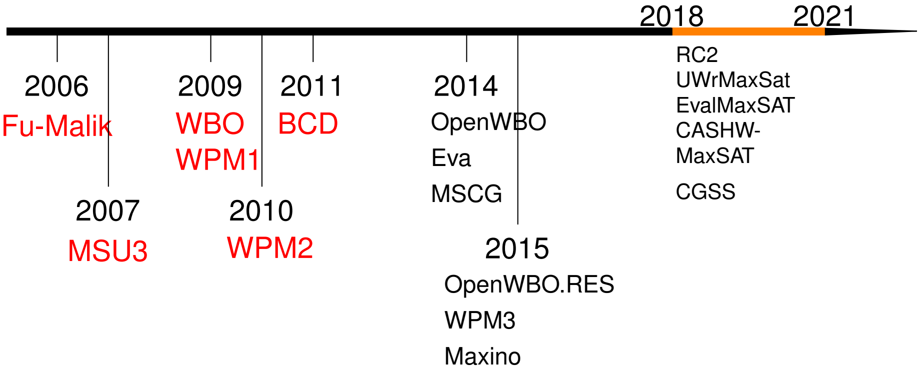
Core-Guided Algorithms

Central Developments



Core-Guided Algorithms

Central Developments

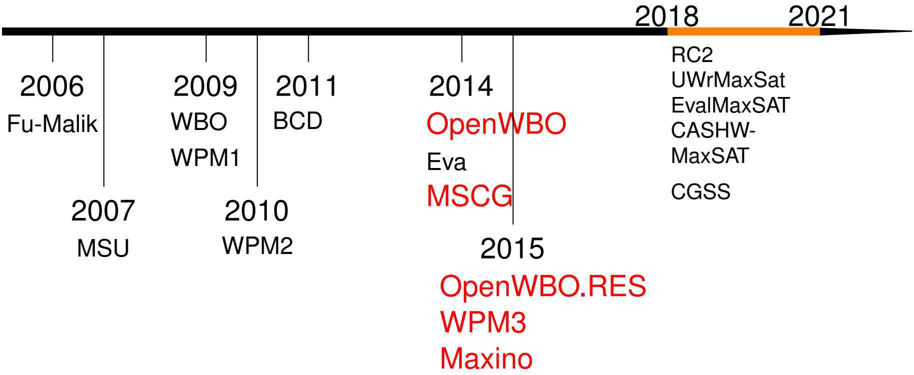


Early algorithms: Core relaxations with hard constraints

[Fu and Malik, 2006; Marques-Silva and Planes, 2007; Manquinho, Marques-Silva, and Planes, 2009; Ansótegui, Bonet, and Levy, 2009, 2010; Heras, Morgado, and Marques-Silva, 2011]

Core-Guided Algorithms

Central Developments



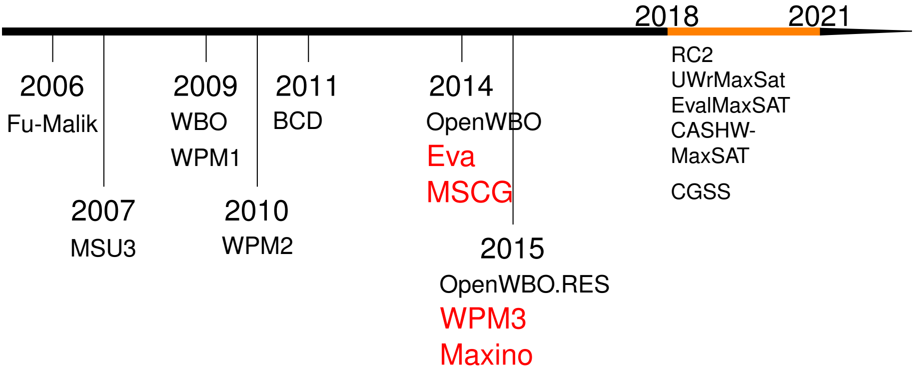
Central improvement 1: Incremental (Lazy) relaxation (cardinality) constraints

[Martins, Joshi, Manquinho, and Lynce, 2014; Morgado, Dodaro, and Marques-Silva, 2014]

Used by all current solvers

Core-Guided Algorithms

Central Developments



Central Improvement 2: Core relaxations with soft constraints

[Narodytska and Bacchus, 2014; Morgado, Dodaro, and Marques-Silva, 2014; Ansótegui, Didier, and Gabàs, 2015]

Core-Guided Algorithms

Central Developments

2006
Fu-Malik

2007
MSU3

2009
WBO
WPM1

2010
WPM2

2011
BCD

2014
OpenWBO
Eva
MSCG

2015
OpenWBO.RES
WPM3
Maxino

2018 2021
RC2
UWrMaxSat
EvalMaxSAT
CASHW-MaxSAT
CGSS

Central Improvement 2: Core relaxations with soft constraints [Narodytska and Bacchus, 2014; Morgado, Dodaro, and Marques-Silva, 2014; Ansótegui, Didier, and Gabàs, 2015]

Current state-of-the-art, effective implementations of OLL [Andres, Kaufmann, Matheis, and Schaub, 2012; Morgado, Dodaro, and Marques-Silva, 2014]

(A few) Additional Techniques

- cover optimization, structure sharing, intrinsic atmost1 [Ansótegui, Bonet, and Levy, 2010; Ansótegui, Didier, and Gabàs, 2015; Ansótegui and Gabàs, 2017; Ihalainen, Berg, and Järvisalo, 2021; Ignatiev, Morgado, and Marques-Silva, 2019]
  - Analyze underlying core structure in order to improve relaxation constraints.
- stratification, weight-aware core extraction, soft clause partitioning [Ansótegui, Bonet, Gabàs, and Levy, 2012; Berg and Järvisalo, 2017; Neves, Martins, Janota, Lynce, and Manquinho, 2015]
  - Extract cores that result in bigger lower bound increases.
  - Connections to multi-level optimization.
- hardening [Ansótegui, Bonet, Gabàs, and Levy, 2012]
  - Fix values of objective literals based on upper bounds
- Recently, presolving with an ILP solver.
  - More on this in the MSE talk.

(A few) Additional Techniques

- cover optimization, structure sharing, intrinsic atmost1 [Ansótegui, Bonet, and Levy, 2010; Ansótegui, Didier, and Gabàs, 2015; Ansótegui and Gabàs, 2017; Ihalainen, Berg, and Järvisalo, 2021; Ignatiev, Morgado, and Marques-Silva, 2019]
  - Analyze underlying core structure in order to improve relaxation constraints.
- stratification, weight-aware core extraction, soft clause partitioning [Ansótegui, Bonet, Gabàs, and Levy, 2012; Berg and Järvisalo, 2017; Neves, Martins, Janota, Lynce, and Manquinho, 2015]
  - Extract cores that result in bigger lower bound increases.
  - Connections to multi-level optimization.
- hardening [Ansótegui, Bonet, Gabàs, and Levy, 2012]
  - Fix values of objective literals based on upper bounds
- Recently, presolving with an ILP solver.
  - More on this in the MSE talk.

(A few) Additional Techniques

- cover optimization, structure sharing, intrinsic atmost1 [Ansótegui, Bonet, and Levy, 2010; Ansótegui, Didier, and Gabàs, 2015; Ansótegui and Gabàs, 2017; Ihalainen, Berg, and Järvisalo, 2021; Ignatiev, Morgado, and Marques-Silva, 2019]
  - Analyze underlying core structure in order to improve relaxation constraints.
- stratification, weight-aware core extraction, soft clause partitioning [Ansótegui, Bonet, Gabàs, and Levy, 2012; Berg and Järvisalo, 2017; Neves, Martins, Janota, Lynce, and Manquinho, 2015]
  - Extract cores that result in bigger lower bound increases.
  - Connections to multi-level optimization.
- hardening [Ansótegui, Bonet, Gabàs, and Levy, 2012]
  - Fix values of objective literals based on upper bounds
- Recently, presolving with an ILP solver.
  - More on this in the MSE talk.

(A few) Additional Techniques

- cover optimization, structure sharing, intrinsic atmost1 [Ansótegui, Bonet, and Levy, 2010; Ansótegui, Didier, and Gabàs, 2015; Ansótegui and Gabàs, 2017; Ihalainen, Berg, and Järvisalo, 2021; Ignatiev, Morgado, and Marques-Silva, 2019]
  - Analyze underlying core structure in order to improve relaxation constraints.
- stratification, weight-aware core extraction, soft clause partitioning [Ansótegui, Bonet, Gabàs, and Levy, 2012; Berg and Järvisalo, 2017; Neves, Martins, Janota, Lynce, and Manquinho, 2015]
  - Extract cores that result in bigger lower bound increases.
  - Connections to multi-level optimization.
- hardening [Ansótegui, Bonet, Gabàs, and Levy, 2012]
  - Fix values of objective literals based on upper bounds
- Recently, presolving with an ILP solver.
  - More on this in the MSE talk.

(A few) Additional Techniques

- cover optimization, structure sharing, intrinsic atmost1 [Ansótegui, Bonet, and Levy, 2010; Ansótegui, Didier, and Gabàs, 2015; Ansótegui and Gabàs, 2017; Ihalainen, Berg, and Järvisalo, 2021; Ignatiev, Morgado, and Marques-Silva, 2019]
  - Analyze underlying core structure in order to improve relaxation constraints.
- stratification, weight-aware core extraction, soft clause partitioning [Ansótegui, Bonet, Gabàs, and Levy, 2012; Berg and Järvisalo, 2017; Neves, Martins, Janota, Lynce, and Manquinho, 2015]
  - Extract cores that result in bigger lower bound increases.
  - Connections to multi-level optimization.
- hardening [Ansótegui, Bonet, Gabàs, and Levy, 2012]
  - Fix values of objective literals based on upper bounds
- Recently, presolving with an ILP solver.
  - More on this in the MSE talk.

Implicit Hitting Set Based MaxSAT Solving

Definition: Hitting sets

- Set of objective literals with non-empty intersection with cores.
- *cost* of hitting set, number of literals in it.

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

$$\kappa^1 = \{a, b\}$$

$$\kappa^2 = \{h, d, f, m\}$$

$$\kappa^3 = \{q, k, r\}$$

$$\text{CORES} = \{\kappa^1, \kappa^2, \kappa^3\}$$

$$hs_1 = \{a, d, f, q\} \quad cost(hs_1) = 4$$

$$hs_2 = \{b, m, q\} \quad cost(hs_2) = 3$$

MaxSAT with hitting sets

First proposed in [Davies and Bacchus, 2011]

Intuition

- Every solution corresponds to a hitting set over all cores.
- Cost of solutions match cost of corresponding hitting sets.
- Central insight - we do not need every core.
  - $cost(hs) \leq$ OPT-COST($\mathcal{F}$) holds for minimum cost hs over *any* set of CORES

MaxSAT with hitting sets

First proposed in [Davies and Bacchus, 2011]

Intuition

- Every solution corresponds to a hitting set over all cores.
- Cost of solutions match cost of corresponding hitting sets.
- Central insight - we do not need every core.
  - $cost(hs) \leq \textsc{opt-cost}(\mathcal{F})$ holds for minimum cost *hs* over *any* set of CORES

n	o		p	q
h	i	j	k	**G**
c	d	e	l	r
a		f		s
S	b	g	m	t

$\kappa^1 = \{a, b\}$

$\kappa^2 = \{h, d, f, m\}$

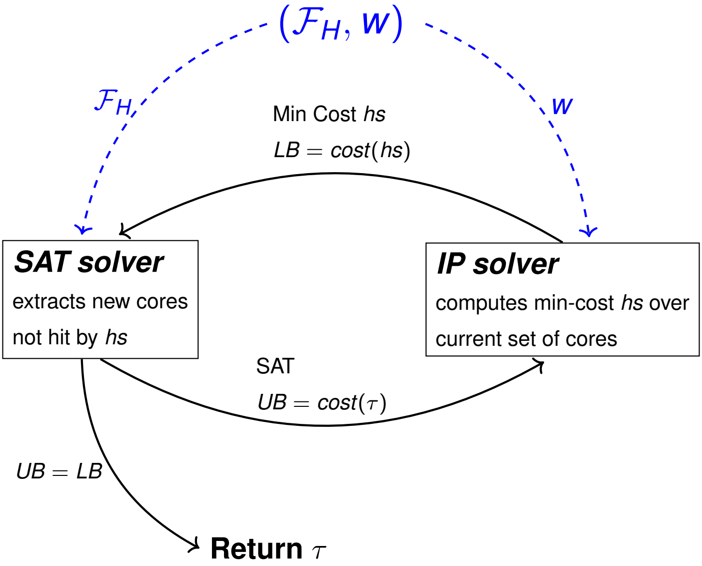
$\kappa^3 = \{q, k, r\}$

$\textsc{cores} = \{\kappa^1, \kappa^2, \kappa^3\}$

$hs = \{b, m, q\} \quad cost(hs) = 3 \leq 6 = \textsc{opt-cost}(\mathcal{F})$

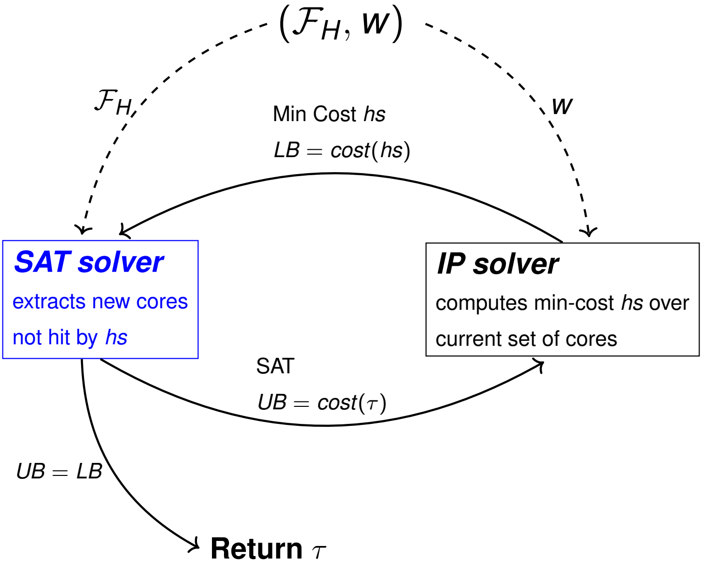
The IHS approach to MaxSAT

[Davies and Bacchus, 2011]



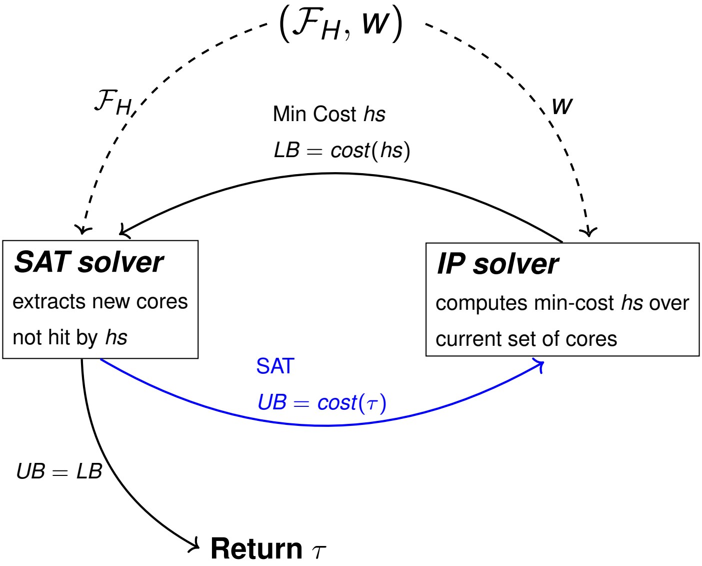
The IHS approach to MaxSAT

[Davies and Bacchus, 2011]



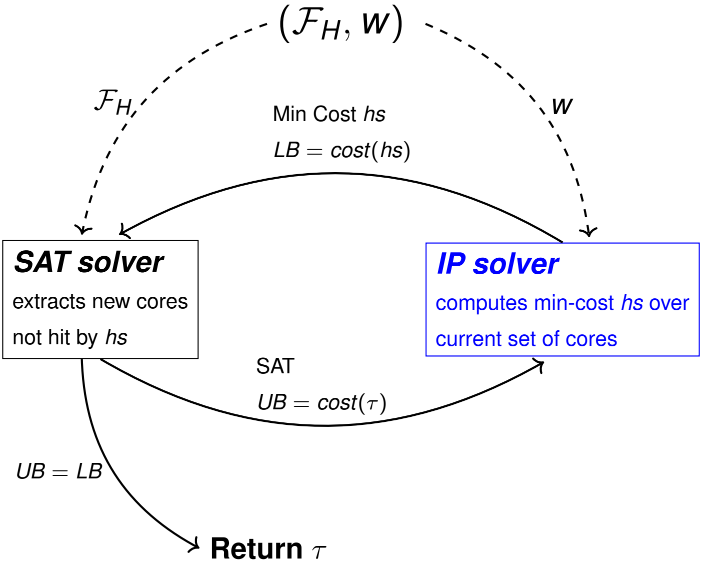
The IHS approach to MaxSAT

[Davies and Bacchus, 2011]



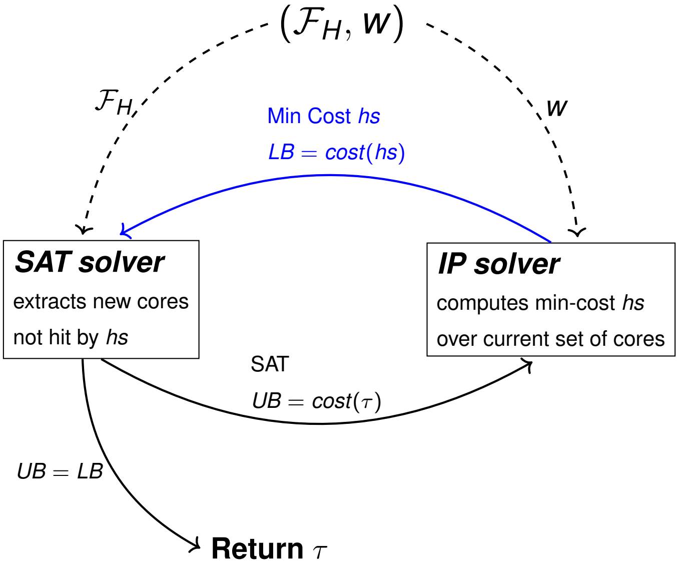
The IHS approach to MaxSAT

[Davies and Bacchus, 2011]



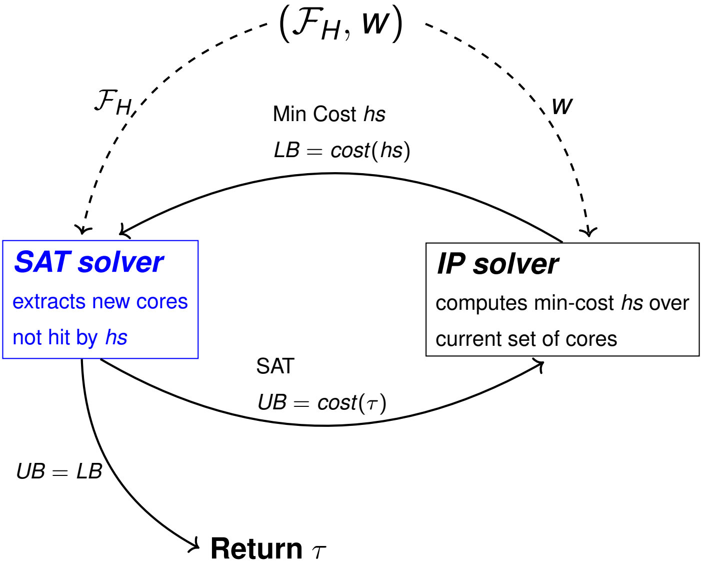
The IHS approach to MaxSAT

[Davies and Bacchus, 2011]



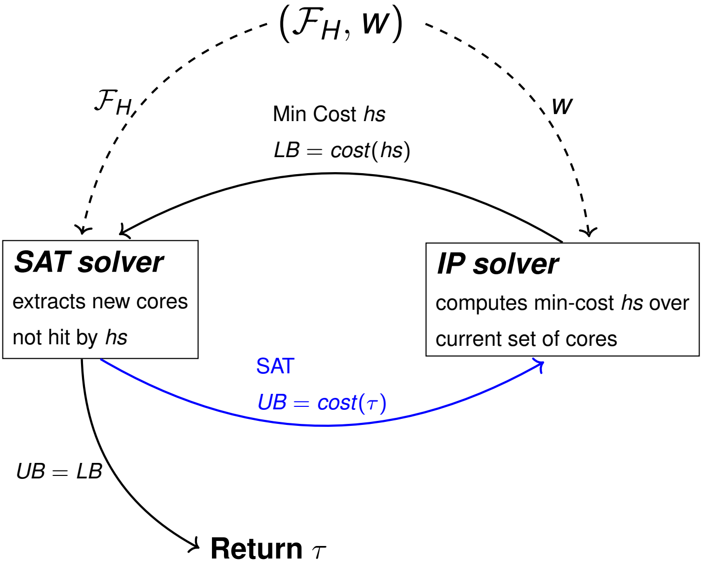
The IHS approach to MaxSAT

[Davies and Bacchus, 2011]



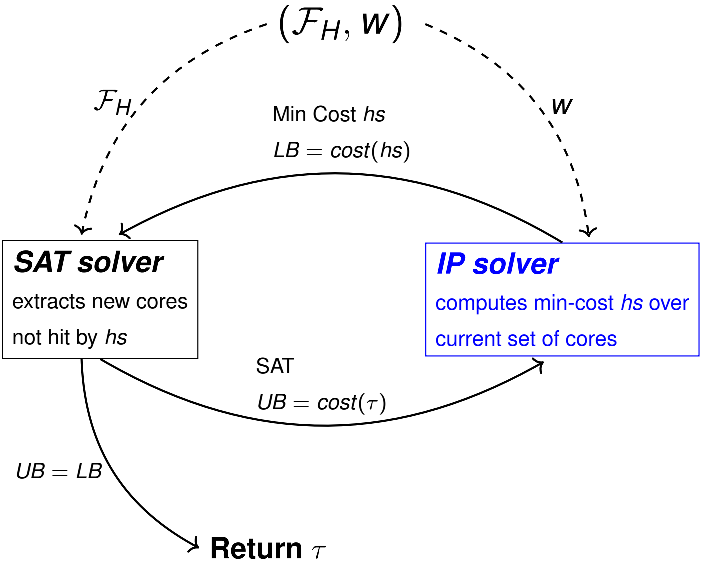
The IHS approach to MaxSAT

[Davies and Bacchus, 2011]



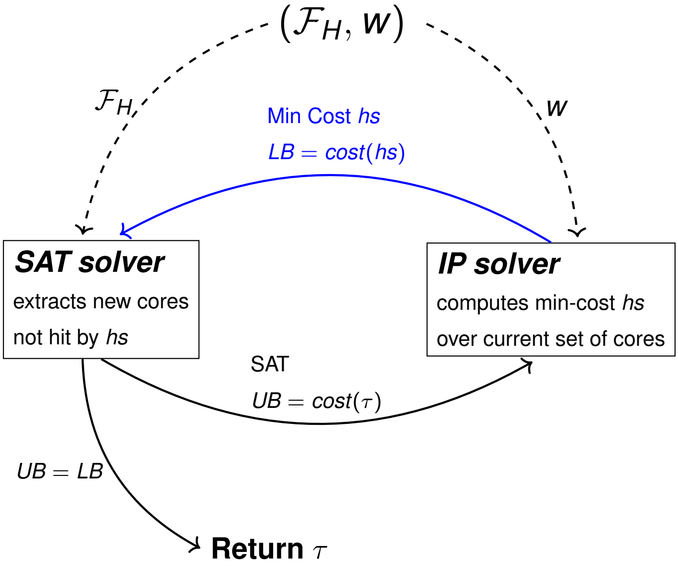
The IHS approach to MaxSAT

[Davies and Bacchus, 2011]



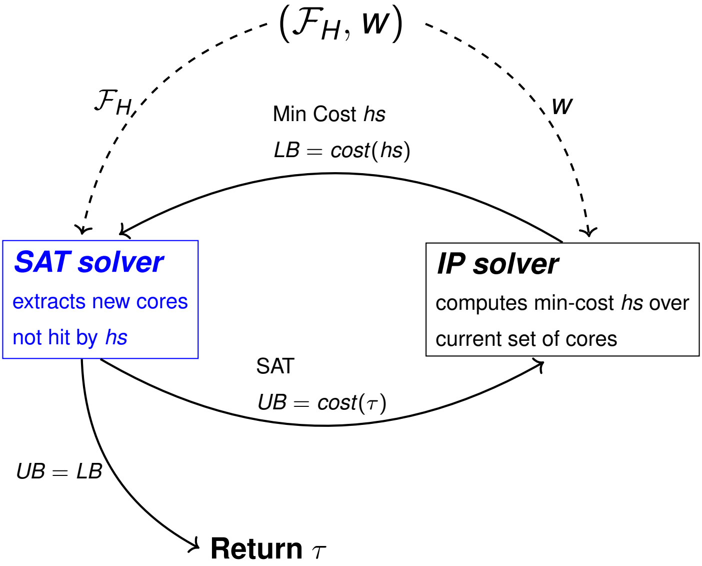
The IHS approach to MaxSAT

[Davies and Bacchus, 2011]



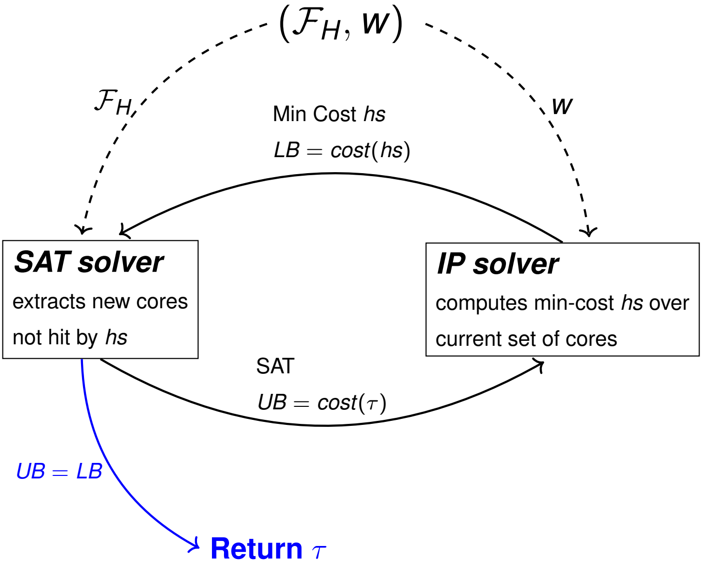
The IHS approach to MaxSAT

[Davies and Bacchus, 2011]



The IHS approach to MaxSAT

[Davies and Bacchus, 2011]



(Some) Further Techniques

- non-optimal hitting sets [Saikko, Berg, and Järvisalo, 2016].
  - extract more cores.
- reduced cost fixing [Bacchus, Hyttinen, Järvisalo, and Saikko, 2017]
  - use information from the IP solver in order to fix objective literals in the SAT-solver
- abstract cores [Berg, Bacchus, and Poole, 2020].
  - Add extension variables and extract cores over those.

Solvers

MaxHS [Davies and Bacchus, 2013],
LMHS [Saikko, Berg, and Järvisalo, 2016]

(Some) Further Techniques

- non-optimal hitting sets [Saikko, Berg, and Järvisalo, 2016].
  - extract more cores.
- reduced cost fixing [Bacchus, Hyttinen, Järvisalo, and Saikko, 2017]
  - use information from the IP solver in order to fix objective literals in the SAT-solver
- abstract cores [Berg, Bacchus, and Poole, 2020].
  - Add extension variables and extract cores over those.

Solvers

MaxHS [Davies and Bacchus, 2013],
LMHS [Saikko, Berg, and Järvisalo, 2016]

Algorithms

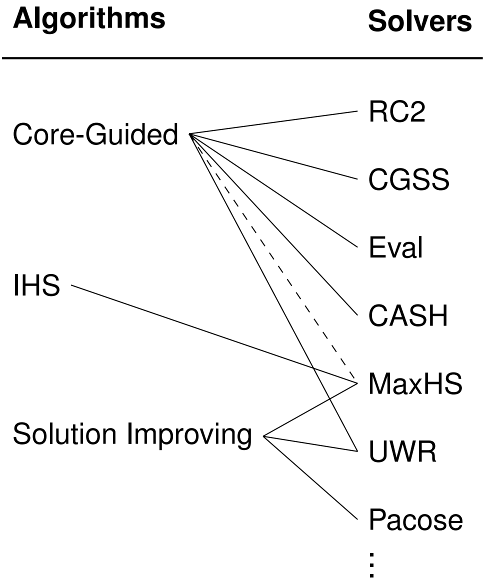
Core-Guided

IHS

Solution Improving

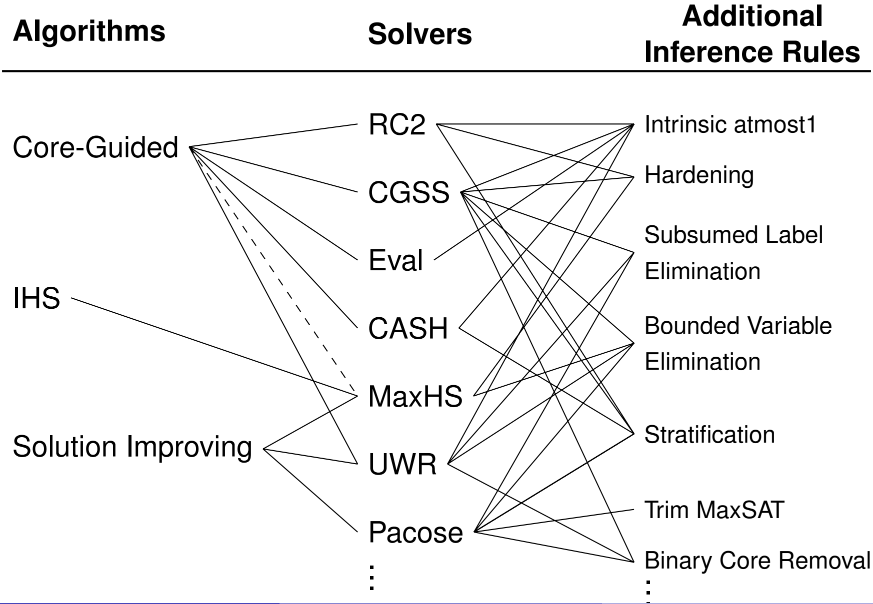
SAT-Based MaxSAT solving

Conclusion



SAT-Based MaxSAT solving

Conclusion



Take Home Message
Modern SAT-Based MaxSAT solvers implement a large number of different heuristics and algorithms that interact in intricate ways.

A central challenge
of the field is understanding these interactions and which techniques are effective on the benchmarks of interest.

Other Interesting Topics

Other Research Directions

Incremental MaxSAT

- Solve sequences of related MaxSAT instances.
- IPAMIR interface for incremental computations.
  - Used to realize the new incremental track in this years MSE.

[Niskanen, Berg, and Järvisalo, 2022; Si, Zhang, Manquinho, Janota, Ignatiev, and Naik, 2016; Niskanen, Berg, and Järvisalo, 2021]

Incomplete (any-time) MaxSAT

- Compute a solution of as low cost as possible within limited time & memory.
- Many specialised solvers, including local search solvers.

[Cohen, Nadel, and Ryvchin, 2021; Zheng, He, Zhou, Jin, Li, and Manyà, 2022; Cai and Lei, 2020; Nadel, 2018; Berg, Demirovic, and Stuckey, 2019; Demirovic and Stuckey, 2019]

Other Research Directions

Incremental MaxSAT

- Solve sequences of related MaxSAT instances.
- IPAMIR interface for incremental computations.
  - Used to realize the new incremental track in this years MSE.

[Niskanen, Berg, and Järvisalo, 2022; Si, Zhang, Manquinho, Janota, Ignatiev, and Naik, 2016; Niskanen, Berg, and Järvisalo, 2021]

Incomplete (any-time) MaxSAT

- Compute a solution of as low cost as possible within limited time & memory.
- Many specialised solvers, including local search solvers.

[Cohen, Nadel, and Ryvchin, 2021; Zheng, He, Zhou, Jin, Li, and Manyà, 2022; Cai and Lei, 2020; Nadel, 2018; Berg, Demirovic, and Stuckey, 2019; Demirovic and Stuckey, 2019]

Other Research Directions

Incremental MaxSAT

- Solve sequences of related MaxSAT instances.
- IPAMIR interface for incremental computations.
  - Used to realize the new incremental track in this years MSE.

[Niskanen, Berg, and Järvisalo, 2022; Si, Zhang, Manquinho, Janota, Ignatiev, and Naik, 2016; Niskanen, Berg, and Järvisalo, 2021]

Incomplete (any-time) MaxSAT

- Compute a solution of as low cost as possible within limited time & memory.
- Many specialised solvers, including local search solvers.

[Cohen, Nadel, and Ryvchin, 2021; Zheng, He, Zhou, Jin, Li, and Manyà, 2022; Cai and Lei, 2020; Nadel, 2018; Berg, Demirovic, and Stuckey, 2019; Demirovic and Stuckey, 2019]

Other Research Directions

Branch & Bound for MaxSAT

- Early work effective especially on small, difficult problems. [Abramé and Habet, 2016, 2014; Li, Manyà, and Planes, 2005]
- Recent paper on adding clause learning to B&B algorithms.
  - MaxCDCL solver [Li, Xu, Coll, Manyà, Habet, and He, 2021].

Abstract Reasoning & Preprocessing

- MaxSAT resolution [Bonet, Levy, and Manyà, 2007; Bonet, Buss, Ignatiev, Marques-Silva, and Morgado, 2018]
- Clause redundancy notions lifted from SAT to MaxSAT [Ihalainen, Berg, and Järvisalo, 2022; Belov, Morgado, and Marques-Silva, 2013].
- Standalone preprocessors available.
  - MaxPRE [Korhonen, Berg, Saikko, and Järvisalo, 2017; Ihalainen, Berg, and Järvisalo, 2022].
  - Coprocessor [Manthey, 2012]

Other Research Directions

Branch & Bound for MaxSAT

- Early work effective especially on small, difficult problems. [Abramé and Habet, 2016, 2014; Li, Manyà, and Planes, 2005]
- Recent paper on adding clause learning to B&B algorithms.
  - MaxCDCL solver [Li, Xu, Coll, Manyà, Habet, and He, 2021].

Abstract Reasoning & Preprocessing

- MaxSAT resolution [Bonet, Levy, and Manyà, 2007; Bonet, Buss, Ignatiev, Marques-Silva, and Morgado, 2018]
- Clause redundancy notions lifted from SAT to MaxSAT [Ihalainen, Berg, and Järvisalo, 2022; Belov, Morgado, and Marques-Silva, 2013].
- Standalone preprocessors available.
  - MaxPRE [Korhonen, Berg, Saikko, and Järvisalo, 2017; Ihalainen, Berg, and Järvisalo, 2022].
  - Coprocessor [Manthey, 2012]

Conclusions

SAT-based MaxSAT solving:

- Effective approach for solving industrial MaxSAT instances
- Combine various algorithms and heuristics in non trivial ways.
- Active (and interesting) area of research.
  - Many ideas have also been studied in e.g. PBO and CP.
    [Devriendt, Gocht, Demirovic, Nordström, and Stuckey, 2021; Smirnov, Berg, and Järvisalo, 2021; Gange, Berg, Demirovic, and Stuckey, 2020]

Further Resources

Surveys in the handbook of satisfiability [Bacchus, Järvisalo, and Martins, 2021; Li and Manyà, 2021]

The webpage of the MaxSAT Evaluation:
https://maxsat-evaluations.github.io/.

Conclusions

SAT-based MaxSAT solving:

- Effective approach for solving industrial MaxSAT instances
- Combine various algorithms and heuristics in non trivial ways.
- Active (and interesting) area of research.
  - Many ideas have also been studied in e.g. PBO and CP.
    [Devriendt, Gocht, Demirovic, Nordström, and Stuckey, 2021; Smirnov, Berg, and Järvisalo, 2021; Gange, Berg, Demirovic, and Stuckey, 2020]

Further Resources

Surveys in the handbook of satisfiability [Bacchus, Järvisalo, and Martins, 2021; Li and Manyà, 2021]

The webpage of the MaxSAT Evaluation:
`https://maxsat-evaluations.github.io/`.

Bibliography I

André Abramé and Djamal Habet. Ahmaxsat: Description and evaluation of a branch and bound max-sat solver. *J. Satisf. Boolean Model. Comput.*, 9(1):89–128, 2014.

André Abramé and Djamal Habet. Learning nobetter clauses in max-sat branch and bound solvers. In *ICTAI*, pages 452–459. IEEE Computer Society, 2016.

B. Andres, B. Kaufmann, O. Matheis, and T. Schaub. Unsatisfiability-based optimization in clasp. In *Proc. ICLP Technical Communications*, volume 17 of *LIPIcs*, pages 211–221. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.

C. Ansótegui, M.L. Bonet, and J. Levy. Solving (Weighted) Partial MaxSAT through Satisfiability Testing. In *Proc. SAT*, volume 5584 of *Lecture Notes in Computer Science*, pages 427–440. Springer, 2009.

C. Ansótegui, M.L. Bonet, and J. Levy. A New Algorithm for Weighted Partial MaxSAT. In *Proc. AAAI*. AAAI Press, 2010.

Carlos Ansótegui and Joel Gabàs. WPM3: An (in)complete algorithm for weighted partial MaxSAT. *Artificial Intelligence*, 250: 37–57, 2017. ISSN 0004-3702. doi: 10.1016/j.artint.2017.05.003.

Carlos Ansótegui, Maria Luisa Bonet, Joel Gabàs, and Jordi Levy. Improving SAT-based weighted MaxSAT solvers. In *Proc. CP*, volume 7514 of *Lecture Notes in Computer Science*, pages 86–101. Springer, 2012.

Carlos Ansótegui, Frédéric Didier, and Joel Gabàs. Exploiting the Structure of Unsatisfiable Cores in MaxSAT. In *Proc. IJCAI*, pages 283–289. AAAI Press, 2015.

Fahiem Bacchus, Antti Hyttinen, Matti Järvisalo, and Paul Saikko. Reduced cost fixing in maxsat. In *Proc. CP*, volume 10416 of *Lecture Notes in Computer Science*, pages 641–651. Springer, 2017. doi: 10.1007/978-3-319-66158-2_41. URL https://doi.org/10.1007/978-3-319-66158-2_41.

Fahiem Bacchus, Matti Järvisalo, and Ruben Martins. Maximum satisfiability. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, Frontiers in Artificial Intelligence and Applications, chapter 24, pages 929–991. IOS Press, 2021.

A. Belov, A. Morgado, and J. Marques-Silva. SAT-based preprocessing for MaxSAT. In *Proc. LPAR-19*, volume 8312 of *Lecture Notes in Computer Science*, pages 96–111. Springer, 2013.

Jeremias Berg and Matti Järvisalo. Weight-aware core extraction in SAT-based MaxSAT solving. In *Proc. CP*, Lecture Notes in Computer Science, 2017. to appear.

Jeremias Berg, Emir Demirovic, and Peter J. Stuckey. Core-boosted linear search for incomplete MaxSAT. In *Proc. CPAIOR*, volume 11494 of *Lecture Notes in Computer Science*, pages 39–56. Springer, 2019.

Bibliography II

Jeremias Berg, Fahiem Bacchus, and Alex Poole. Abstract cores in implicit hitting set maxsat solving. In *SAT*, volume 12178 of *Lecture Notes in Computer Science*, pages 277–294. Springer, 2020.

Maria Luisa Bonet, Jordi Levy, and Felip Manyà. Resolution for Max-SAT. *Artificial Intelligence*, 171(8-9):606–618, 2007.

Maria Luisa Bonet, Sam Buss, Alexey Ignatiev, Joao Marques-Silva, and António Morgado. MaxSAT resolution with the dual rail encoding. In *AAAI*. AAAI Press, 2018.

Shaowei Cai and Zhendong Lei. Old techniques in new ways: Clause weighting, unit propagation and hybridization for maximum satisfiability. *Artif. Intell.*, 287:103354, 2020.

Aviad Cohen, Alexander Nadel, and Vadim Ryvchin. Local search with a SAT oracle for combinatorial optimization. In *TACAS (2)*, volume 12652 of *Lecture Notes in Computer Science*, pages 87–104. Springer, 2021.

J. Davies and F. Bacchus. Solving MAXSAT by Solving a Sequence of Simpler SAT Instances. In *Proc. CP*, volume 6876 of *Lecture Notes in Computer Science*, pages 225–239. Springer, 2011.

J. Davies and F. Bacchus. Exploiting the power of MIP solvers in MaxSAT. In *Proc. SAT*, volume 7962 of *Lecture Notes in Computer Science*, pages 166–181. Springer, 2013.

Emir Demirovic and Peter J. Stuckey. Techniques inspired by local search for incomplete maxsat and the linear algorithm: Varying resolution and solution-guided search. In Thomas Schiex and Simon de Givry, editors, *Principles and Practice of Constraint Programming - 25th International Conference, CP 2019, Stamford, CT, USA, September 30 - October 4, 2019, Proceedings*, volume 11802 of *Lecture Notes in Computer Science*, pages 177–194. Springer, 2019. doi: 10.1007/978-3-030-30048-7_11. URL `https://doi.org/10.1007/978-3-030-30048-7_11`.

Jo Devriendt, Stephan Gocht, Emir Demirovic, Jakob Nordström, and Peter J. Stuckey. Cutting to the core of pseudo-boolean optimization: Combining core-guided search with cutting planes reasoning. In *AAAI*, pages 3750–3758. AAAI Press, 2021.

Z. Fu and S. Malik. On solving the partial MaxSAT problem. In *Proc. SAT*, volume 4121 of *Lecture Notes in Computer Science*, pages 252–265. Springer, 2006.

Graeme Gange, Jeremias Berg, Emir Demirovic, and Peter J. Stuckey. Core-guided and core-boosted search for CP. In *CPAIOR*, volume 12296 of *Lecture Notes in Computer Science*, pages 205–221. Springer, 2020.

F. Heras, A. Morgado, and J. Marques-Silva. Core-Guided Binary Search Algorithms for Maximum Satisfiability. In *Proc. AAAI*. AAAI Press, 2011.

Bibliography III

Alexey Ignatiev, António Morgado, and João Marques-Silva. RC2: an efficient MaxSAT solver. *J. Satisf. Boolean Model. Comput.*, 11(1):53–64, 2019.

Hannes Ihalainen, Jeremias Berg, and Matti Järvisalo. Refined core relaxation for core-guided maxsat solving. In *Proc. CP*, volume 210 of *LIPIcs*, pages 28:1–28:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

Hannes Ihalainen, Jeremias Berg, and Matti Järvisalo. Clause redundancy and preprocessing in maximum satisfiability. In *IJCAR*. Springer, 2022.

Tuukka Korhonen, Jeremias Berg, Paul Saikko, and Matti Järvisalo. MaxPre: An extended MaxSAT preprocessor. In *Proc. SAT*, volume 10491 of *Lecture Notes in Computer Science*, pages 449–456, 2017.

M. Koshimura, T. Zhang, H. Fujita, and R. Hasegawa. QMaxSAT: A Partial Max-SAT Solver. *Journal of Satisfiability, Boolean Modeling and Computation*, 8(1/2):95–100, 2012.

Chu Min Li and Felip Manyà. Maxsat, hard and soft constraints. In *Handbook of Satisfiability*, volume 336 of *Frontiers in Artificial Intelligence and Applications*, pages 903–927. IOS Press, 2021.

Chu Min Li, Felip Manyà, and Jordi Planes. Exploiting unit propagation to compute lower bounds in branch and bound max-sat solvers. In Peter van Beek, editor, *Principles and Practice of Constraint Programming - CP 2005, 11th International Conference, CP 2005, Sitges, Spain, October 1-5, 2005, Proceedings*, volume 3709 of *Lecture Notes in Computer Science*, pages 403–414. Springer, 2005. doi: 10.1007/11564751_31. URL https://doi.org/10.1007/11564751_31.

Chu-Min Li, Zhenxing Xu, Jordi Coll, Felip Manyà, Djamal Habet, and Kun He. Combining clause learning and branch and bound for maxsat. In *CP*, volume 210 of *LIPIcs*, pages 38:1–38:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

V.M. Manquinho, J.P. Marques-Silva, and J. Planes. Algorithms for Weighted Boolean Optimization. In *Proc. SAT*, volume 5584 of *Lecture Notes in Computer Science*, pages 495–508. Springer, 2009.

N. Manthey. Coprocessor 2.0 - A flexible CNF simplifier. In *Proc. SAT*, volume 7317 of *Lecture Notes in Computer Science*, pages 436–441. Springer, 2012.

João Marques-Silva and Jordi Planes. On Using Unsatisfiability for Solving Maximum Satisfiability. *CoRR*, abs/0712.1097, 2007.

R. Martins, S. Joshi, V.M. Manquinho, and I. Lynce. Incremental Cardinality Constraints for MaxSAT. In *Proc. CP*, volume 8656 of *Lecture Notes in Computer Science*, pages 531–548. Springer, 2014.

Bibliography IV

A. Morgado, C. Dodaro, and J. Marques-Silva. Core-Guided MaxSAT with Soft Cardinality Constraints. In *Proc. CP*, volume 8656 of *Lecture Notes in Computer Science*, pages 564–573. Springer, 2014.

Alexander Nadel. Solving maxsat with bit-vector optimization. In *SAT*, volume 10929 of *Lecture Notes in Computer Science*, pages 54–72. Springer, 2018.

N. Narodytska and F. Bacchus. Maximum satisfiability using core-guided MaxSAT resolution. In *Proc. AAAI*, pages 2717–2723. AAAI Press, 2014.

Miguel Neves, Ruben Martins, Mikolás Janota, Inês Lynce, and Vasco M. Manquinho. Exploiting resolution-based representations for MaxSAT solving. In Marijn Heule and Sean Weaver, editors, *Theory and Applications of Satisfiability Testing - SAT 2015 - 18th International Conference, Austin, TX, USA, September 24-27, 2015, Proceedings*, volume 9340 of *Lecture Notes in Computer Science*, pages 272–286. Springer, 2015. ISBN 978-3-319-24317-7. doi: 10.1007/978-3-319-24318-4. URL `http://dx.doi.org/10.1007/978-3-319-24318-4`.

Andreas Niskanen, Jeremias Berg, and Matti Järvisalo. Enabling incrementality in the implicit hitting set approach to maxsat under changing weights. In *CP*, volume 210 of *LIPIcs*, pages 44:1–44:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

Andreas Niskanen, Jeremias Berg, and Matti Järvisalo. Incremental maximum satisfiability. In *SAT*, volume ??? of *LIPIcs*, page ??? Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

Tobias Paxian, Sven Reimer, and Bernd Becker. Dynamic polynomial watchdog encoding for solving weighted maxsat. In *SAT*, volume 10929 of *Lecture Notes in Computer Science*, pages 37–53. Springer, 2018.

Tobias Paxian, Pascal Raiola, and Bernd Becker. On preprocessing for weighted maxsat. In *Proc. VMCAI*, volume 12597 of *Lecture Notes in Computer Science*, pages 556–577. Springer, 2021.

P. Saikko, J. Berg, and M. Järvisalo. LMHS: A SAT-IP hybrid MaxSAT solver. In *Proc. SAT*, volume 9710 of *Lecture Notes in Computer Science*, pages 539–546. Springer, 2016.

Xujie Si, Xin Zhang, Vasco M. Manquinho, Mikolás Janota, Alexey Ignatiev, and Mayur Naik. On incremental core-guided MaxSAT solving. In *Proc. CP*, volume 9892 of *Lecture Notes in Computer Science*, pages 473–482. Springer, 2016. doi: 10.1007/978-3-319-44953-1_30. URL `https://doi.org/10.1007/978-3-319-44953-1_30`.

Pavel Smirnov, Jeremias Berg, and Matti Järvisalo. Pseudo-boolean optimization by implicit hitting sets. In *CP*, volume 210 of *LIPIcs*, pages 51:1–51:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

Jiongzhi Zheng, Kun He, Jianrong Zhou, Yan Jin, Chu-Min Li, and Felip Manyà. Bandmaxsat: A local search maxsat solver with multi-armed bandit. In *IJCAI*, pages 1901–1907. ijcai.org, 2022.