Optimization as a Sequence of Decisions

Jeremias Berg

Department of Computer Science, University of Helsinki, Finland

May 22
MIAO

Outline

1. Constraint-based approaches to combinatorial optimization.
2. Constraint optimization algorithms:
   - solution improving
   - core guided
   - implicit hitting set
3. Why such algorithms are actually all based on hitting sets.
4. Our model: UniMaxSAT in more detail.

Joint work Hannes Ihalainen and Matti Järvisalo. Related papers published at IJCAI [Ihalainen et al., 2023] and JAIR (upcoming)

Outline

1. Constraint-based approaches to combinatorial optimization.
2. Constraint optimization algorithms:
   - solution improving
   - core guided
   - implicit hitting set
3. Why such algorithms are actually all based on hitting sets.
4. Our model: UniMaxSAT in more detail.

Joint work Hannes Ihalainen and Matti Järvisalo. Related papers published at IJCAI [Ihalainen et al., 2023] and JAIR (upcoming)

Outline

1. Constraint-based approaches to combinatorial optimization.
2. Constraint optimization algorithms:
   - solution improving
   - core guided
   - implicit hitting set
3. Why such algorithms are actually all based on hitting sets.
4. Our model: UniMaxSAT in more detail.

Joint work Hannes Ihalainen and Matti Järvisalo. Related papers published at IJCAI [Ihalainen et al., 2023] and JAIR (upcoming)

Outline

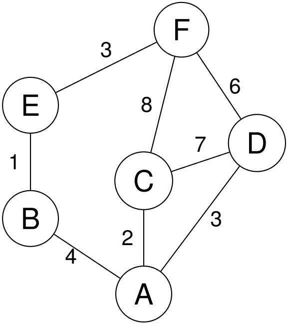
1. Constraint-based approaches to combinatorial optimization.
2. Constraint optimization algorithms:
   - solution improving
   - core guided
   - implicit hitting set
3. Why such algorithms are actually all based on hitting sets.
4. Our model: UniMaxSAT in more detail.

Joint work Hannes Ihalainen and Matti Järvisalo. Related papers published at IJCAI [Ihalainen et al., 2023] and JAIR (upcoming)

Part 1: Combinatorial Optimization

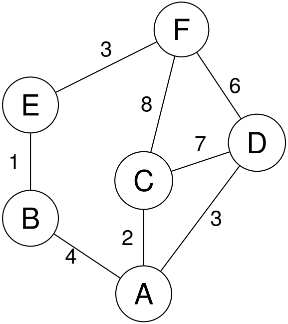
Traveling Salesperson

What is the shortest tour?



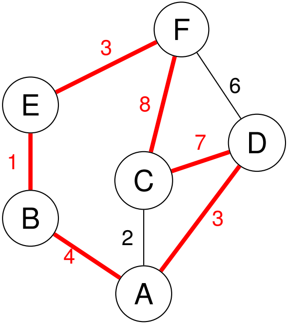
Traveling Salesperson

Is there a tour of length at most 23?



Traveling Salesperson

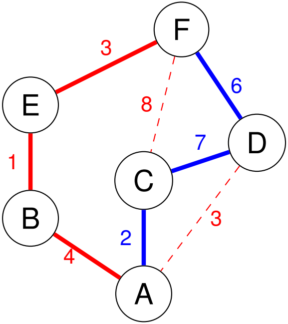
Is there a tour of length at most 23?



Length: 26

Computationally complex optimization problems

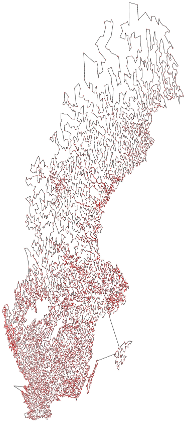
Is there a tour of length at most 23?



Length: 23

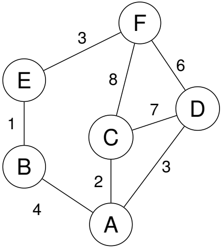
Computationally complex optimization problems

NP-hardness $\rightarrow$ Solutions are hard to find but easy to check



Constraint Optimization

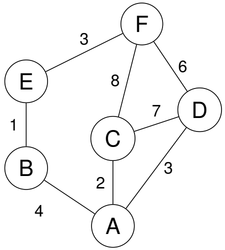
solving by describing solutions



Problem instance

Constraint Optimization

solving by describing solutions



$$\text{MINIMIZE:} \quad 4e_{AB} + e_{BE} + 3e_{EF} + 8e_{FC} + 7e_{CD} + 6e_{FD} + 2e_{AC} + 3e_{ED}$$

SUBJECT TO:

$$e_{AB} + e_{AC} + e_{AD} = 2$$

$$e_{AC} + e_{CD} + e_{CF} = 2$$

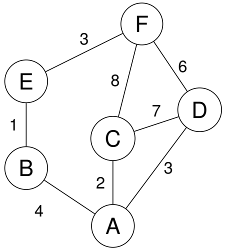
$$\vdots$$

$$e_{IJ} \in \{0, 1\}$$

Problem instance $\longrightarrow$ Constraint representation

Constraint Optimization

solving by describing solutions



$$\text{MINIMIZE: } \quad 4e_{AB} + e_{BE} + 3e_{EF} + 8e_{FC} + 7e_{CD} + 6e_{FD} + 2e_{AC} + 3e_{ED}$$

SUBJECT TO:

$$e_{AB} + e_{AC} + e_{AD} = 2$$

$$e_{AC} + e_{CD} + e_{CF} = 2$$

$$\vdots$$

$$e_{IJ} \in \{0, 1\}$$

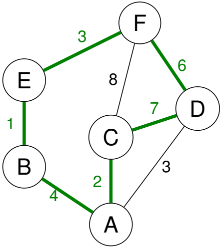
Problem instance $\longrightarrow$ Constraint representation

$\downarrow$

Solution to constraints
Green=1, Red=0

Constraint Optimization

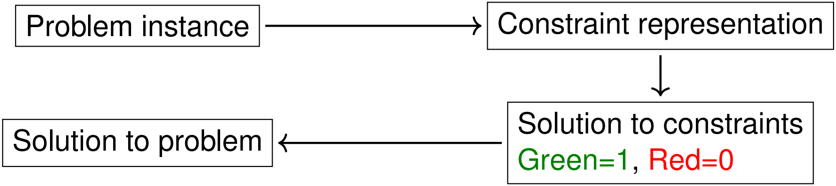
solving by describing solutions



MINIMIZE: $4e_{AB} + e_{BE} + 3e_{EF} + 8e_{FC} + 7e_{CD} + 6e_{FD} + 2e_{AC} + 3e_{ED}$

SUBJECT TO:

$$e_{AB} + e_{AC} + e_{AD} = 2$$
$$e_{AC} + e_{CD} + e_{CF} = 2$$
$$\vdots$$
$$e_{IJ} \in \{0, 1\}$$



Constraint Optimization Problems of Interest

- MaxSAT
- PBO
- Ideas also applied in:
  - CP
  - ASP
  - SMT
  - MILP (maybe...)

MINIMIZE: $\sum_i c_i \cdot b_i$

SUBJECT TO:

$$\left\{ \bigvee_j x_{ij} \mid i = 1 \dots n \right\}$$

all variables $\in \{0, 1\}$

Constraint Optimization Problems of Interest

- MaxSAT
- PBO
- Ideas also applied in:
  - CP
  - ASP
  - SMT
  - MILP (maybe...)

MINIMIZE: $\sum_i c_i \cdot b_i$

SUBJECT TO:

$$\left\{ \sum_j c_{ij} x_{ij} \geq B_i \mid i = 1 \ldots n \right\}$$

all variables $\in \{0, 1\}$

Constraint Optimization Problems of Interest

- MaxSAT
- PBO
- Ideas also applied in:
  - CP
  - ASP
  - SMT
  - MILP (maybe...)

[Smirnov, Berg, and Järvisalo, 2021; Gange, Berg, Demirovic, and Stuckey, 2020; Andres, Kaufmann, Matheis, and Schaub, 2012; Davies, Gange, and Stuckey, 2017; Devriendt, Gocht, Demirovic, Nordström, and Stuckey, 2021; Fazekas, Bacchus, and Biere, 2018]

Part 2: Optimization via a Sequence of Decisions

General Formulation of Constraint Optimization

$\mathcal{F}$	the constraints of the problem
$\textsc{Var}(\mathcal{F})$	the variables of the problem
$cost \equiv \sum_i c_i x_i$	the objective to minimize

$$\boxed{\text{Minimize } cost \text{ subject to } \mathcal{F}}$$

Example Problem

Shortest Path $S \to G$

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

Example Problem

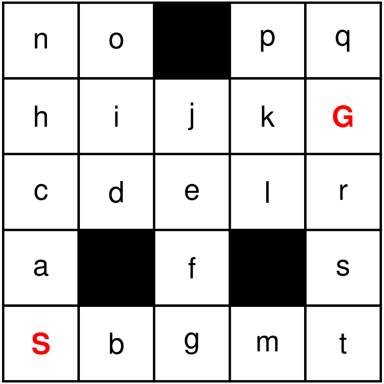
Shortest Path $S \to G$

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

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

Example Problem

Shortest Path $S \to G$



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

$$\mathcal{F} = \text{ISPATH}(S, G)$$

Example Problem

Shortest Path $S \rightarrow G$

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

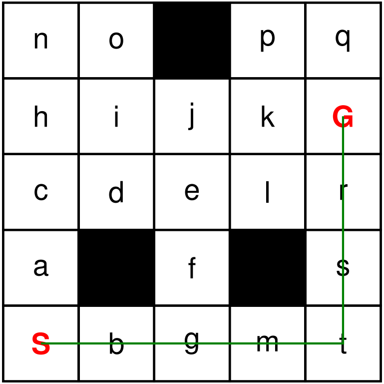
$\textsc{var}(\mathcal{F}) = \{a, b, c, \ldots, r, s, t\}$

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

$cost = a + b + c + \ldots + r + s + t$

Example Problem

Shortest Path $S \to G$



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

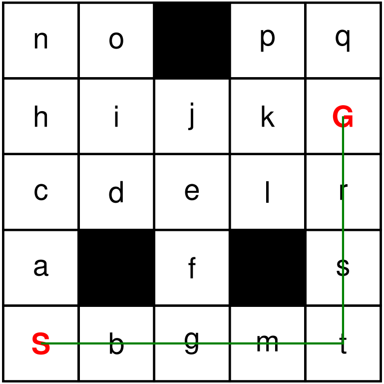
$\mathcal{F} = \text{IsPath}(S, G)$

$\tau : x = 1$ for $x \in \{b, g, m, t, s, r\}$
$x = 0$ for $x \in \{a, c, \ldots\}$

$cost = a + b + c + \ldots + r + s + t = 6$

Example Problem

Shortest Path $S \to G$



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

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

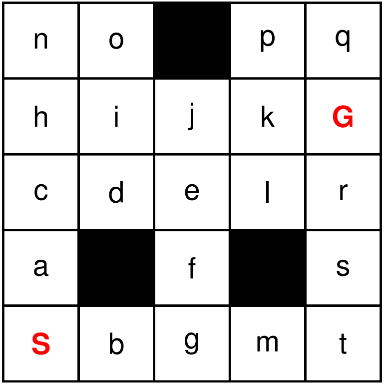
$\tau : x = 1$ for $x \in \{b, g, m, t, s, r\}$
$x = 0$ for $x \in \{a, c, \ldots\}$

$\tau \equiv \{b, g, m, t, s, r\}$

$cost = a + b + c + \ldots + r + s + t = 6$

Central Concept: UNSAT Cores

Constraint κ is a core if $\tau(\kappa) = 1$ for all solutions.



$\kappa^1 = (a + b \geq 1) = (a \vee b)$

all paths go through either a or b

$\kappa^2 = (h + d + f + m \geq 1)$

all paths go through (at least) one of h, d, f or m.

$\kappa^3 = (q + k + r \geq 1)$

all paths go through (at least) one of q, k or r.

Central Concept: UNSAT Cores

Constraint κ is a core if $\tau(\kappa) = 1$ for all solutions.

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

$\kappa^1 = (a + b \geq 1) = (a \vee b)$

all paths go through either a or b

$\kappa^2 = (h + d + f + m \geq 1)$

all paths go through (at least) one of h, d, f or m.

$\kappa^3 = (q + k + r \geq 1)$

all paths go through (at least) one of q, k or r.

$\kappa^4 = (h + d + f + m + q + k + r \geq 2)$

$\kappa^5 = (2a + b + g \geq 2)$

Important Assumption on Constraint Optimization Problems of Interest

Let $\mathcal{F}$ a set of constraints and γ a (partial) assignment of its variables.

Core & Solution Computation

We assume a decision procedure $\texttt{ORACLE}(\mathcal{F}, \gamma)$ that returns:

- SAT and a solution $\tau \supseteq \gamma$ of $\mathcal{F}$, or
- UNSAT and a *core* κ falsified by γ.

Important Assumption on Constraint Optimization Problems of Interest

Let $\mathcal{F}$ a set of constraints and γ a (partial) assignment of its variables.

Core & Solution Computation

We assume a decision procedure $\texttt{ORACLE}(\mathcal{F}, \gamma)$ that returns:

- SAT and a solution $\tau \supseteq \gamma$ of $\mathcal{F}$, or
- UNSAT and a *core* κ falsified by γ.

Important Assumption on Constraint Optimization Problems of Interest

Let $\mathcal{F}$ a set of constraints and γ a (partial) assignment of its variables.

Core & Solution Computation

We assume a decision procedure $\textsc{oracle}(\mathcal{F}, \gamma)$ that returns:

- SAT and a solution $\tau \supseteq \gamma$ of $\mathcal{F}$, or
- UNSAT and a *core* κ falsified by γ.

Assumption holds for:

- MaxSAT
- PBO
- ASP
- SMT
- CP

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

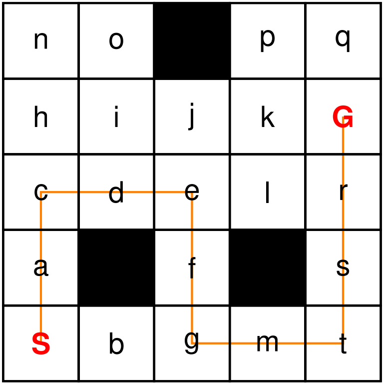
UB $= \infty$

ORACLE$(\mathcal{F}, \emptyset)$

Solution Improving Search

Intuition

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



UB $=$ **10**

$\textsc{Oracle}(\mathcal{F}, \emptyset)$

$\tau^1 = \{a, c, d, e, f, g, m, u, t, r\}$
$cost(\tau^1) = 10$

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

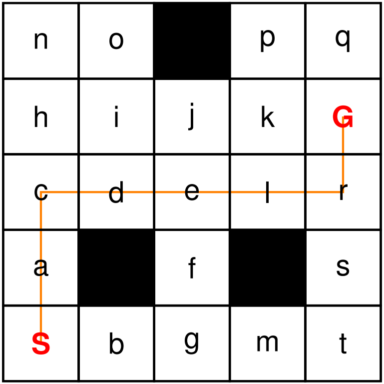
$\text{UB} = 10$

$\textsc{Oracle}(\mathcal{F} \wedge \textsc{CostLessThan}(\textbf{UB}), \emptyset)$

Solution Improving Search

Intuition

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



$\text{UB} = \mathbf{6}$

$\textsc{Oracle}\,(\mathcal{F} \wedge \textsc{CostLessThan}(UB), \emptyset)$

$\tau^2 = \{a, c, d, e, l, r\}$

$cost(\tau^2) = 6$

Core Guided Search

Core Guided Search

[Fu and Malik, 2006; Gange, Berg, Demirovic, and Stuckey, 2020; Devriendt, Gocht, Demirovic, Nordström, and Stuckey, 2021; Andres, Kaufmann, Matheis, and Schaub, 2012]

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 ORACLE for more effective relaxation.

Core Guided Search

[Fu and Malik, 2006; Gange, Berg, Demirovic, and Stuckey, 2020; Devriendt, Gocht, Demirovic, Nordström, and Stuckey, 2021; Andres, Kaufmann, Matheis, and Schaub, 2012]

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 ORACLE for more effective relaxation.

Shortest Path with Core Guided Search

Intuition

1. Initialise $\mathcal{F}^0 = \mathcal{F}$ and $\gamma_B^0 = \{b = 0 \mid b \in \textsc{Var}(cost)\}$
2. For $i = 0 \ldots$ check for solutions $\tau \supseteq \gamma_B^i$ of $\mathcal{F}^i$
3. If NO, update $\mathcal{F}^i$ and γ_B^i
4. If YES, such τ is optimal

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

Shortest Path with Core Guided Search

Intuition

1. Initialise $\mathcal{F}^0 = \mathcal{F}$ and $\gamma_B^0 = \{b = 0 \mid b \in \textsc{Var}(cost)\}$
2. For $i = 0 \ldots$ check for solutions $\tau \supseteq \gamma_B^i$ of $\mathcal{F}^i$
3. If NO, update $\mathcal{F}^i$ and γ_B^i
4. If YES, such τ is optimal

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

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

$\text{ORACLE}(\mathcal{F}^{\mathbf{i}}, \gamma_{\mathbf{B}}^{\mathbf{i}})$

$\mathcal{F}^i \equiv \mathcal{F}$

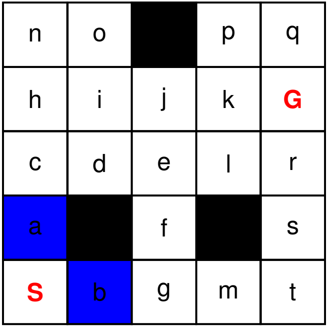
$\gamma_B^i \equiv \{b = 0 \mid b \notin \mathcal{K}\}$

Is there a path that visits at most 1 node from each found core

Shortest Path with Core Guided Search

Intuition

1. Initialise $\mathcal{F}^0 = \mathcal{F}$ and $\gamma_B^0 = \{b = 0 \mid b \in \text{VAR}(cost)\}$
2. For $i = 0 \ldots$ check for solutions $\tau \supseteq \gamma_B^i$ of $\mathcal{F}^i$
3. If NO, update $\mathcal{F}^i$ and γ_B^i
4. If YES, such τ is optimal



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

$\text{ORACLE}(\mathcal{F}^{\mathbf{i}}, \gamma_{\mathbf{B}}^{\mathbf{i}})$

Formula is unsatisfiable
Obtain new core: $\kappa_0 = a + b \geq 1$

Shortest Path with Core Guided Search

Intuition

1. Initialise $\mathcal{F}^0 = \mathcal{F}$ and $\gamma_B^0 = \{b = 0 \mid b \in \textsc{Var}(cost)\}$
2. For $i = 0 \ldots$ check for solutions $\tau \supseteq \gamma_B^i$ of $\mathcal{F}^i$
3. If NO, update $\mathcal{F}^i$ and γ_B^i
4. If YES, such τ 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 $= 1$, $\mathcal{K} = \{\kappa_0\}$

ORACLE$(\mathcal{F}^{\mathbf{i}}, \gamma_{\mathbf{B}}^{\mathbf{i}})$

$\mathcal{F}^i \equiv \mathcal{F} \wedge \bigwedge_{\kappa \in \mathcal{K}} \left(\bar{o}_2^\kappa \rightarrow \sum_{b \in \kappa} b \leq 1\right)$

$\gamma_B^i \equiv \{o_2^\kappa = 0 \mid \kappa \in \mathcal{K}\} \cup \{b = 0 \mid b \notin \mathcal{K}\}$

Is there a path that visits at most 1 node from each found core

Shortest Path with Core Guided Search

Intuition

1. Initialise $\mathcal{F}^0 = \mathcal{F}$ and $\gamma_B^0 = \{b = 0 \mid b \in \textsc{Var}(cost)\}$
2. For $i = 0 \ldots$ check for solutions $\tau \supseteq \gamma_B^i$ of $\mathcal{F}^i$
3. If NO, update $\mathcal{F}^i$ and γ_B^i
4. If YES, such τ is optimal

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

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

$\textsc{Oracle}(\mathcal{F}^{\mathbf{i}}, \gamma_{\mathbf{B}}^{\mathbf{i}})$

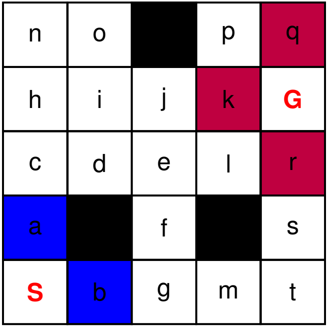
Formula is unsatisfiable

Obtain new core: $\kappa_1 = q + k + r \geq 1$

Shortest Path with Core Guided Search

Intuition

1. Initialise $\mathcal{F}^0 = \mathcal{F}$ and $\gamma_B^0 = \{b = 0 \mid b \in \textsc{Var}(cost)\}$
2. For $i = 0 \ldots$ check for solutions $\tau \supseteq \gamma_B^i$ of $\mathcal{F}^i$
3. If NO, update $\mathcal{F}^i$ and γ_B^i
4. If YES, such τ is optimal



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

$\textsc{Oracle}(\mathcal{F}^{\mathbf{i}}, \gamma_{\mathbf{B}}^{\mathbf{i}})$

$\mathcal{F}^i \equiv \mathcal{F} \wedge \bigwedge_{\kappa \in \mathcal{K}} \left(\bar{o}_2^\kappa \rightarrow \sum_{b \in \kappa} b \leq 1\right)$

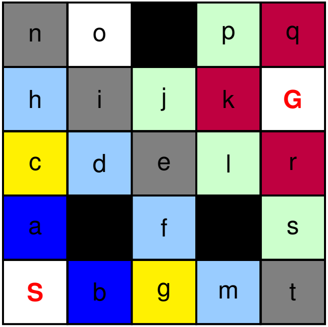
$\gamma_B^i \equiv \{o_2^\kappa = 0 \mid \kappa \in \mathcal{K}\} \cup \{b = 0 \mid b \notin \mathcal{K}\}$

Is there a path that visits at most 1 node from each found core

Shortest Path with Core Guided Search

Intuition

1. Initialise $\mathcal{F}^0 = \mathcal{F}$ and $\gamma_B^0 = \{b = 0 \mid b \in \textsc{Var}(cost)\}$
2. For $i = 0 \ldots$ check for solutions $\tau \supseteq \gamma_B^i$ of $\mathcal{F}^i$
3. If NO, update $\mathcal{F}^i$ and γ_B^i
4. If YES, such τ is optimal



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

$\textsc{Oracle}(\mathcal{F}^{\mathbf{i}}, \gamma_{\mathbf{B}}^{\mathbf{i}})$

$\mathcal{F}^i \equiv \mathcal{F} \wedge \bigwedge_{\kappa \in \mathcal{K}} \left(\bar{o}_2^{\kappa} \rightarrow \sum_{b \in \kappa} b \leq 1\right)$

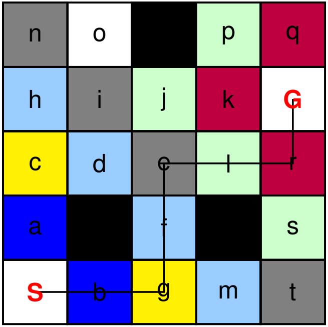
$\gamma_B^i \equiv \{o_2^{\kappa} = 0 \mid \kappa \in \mathcal{K}\} \cup \{b = 0 \mid b \notin \mathcal{K}\}$

Is there a path that visits at most 1 node from each found core

Shortest Path with Core Guided Search

Intuition

1. Initialise $\mathcal{F}^0 = \mathcal{F}$ and $\gamma_B^0 = \{b = 0 \mid b \in \textsc{Var}(cost)\}$
2. For $i = 0 \ldots$ check for solutions $\tau \supseteq \gamma_B^i$ of $\mathcal{F}^i$
3. If NO, update $\mathcal{F}^i$ and γ_B^i
4. If YES, such τ is optimal



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

$\textsc{Oracle}(\mathcal{F}^{\mathbf{i}}, \gamma_{\mathbf{B}}^{\mathbf{i}})$

Formula is satisfiable
Obtain optimal: $\tau = \{b, \ldots, l, r\}$
$cost(\tau) = 6$

There are many ways to relax

OLL [Andres, Kaufmann, Matheis, and Schaub, 2012; Morgado, Dodaro, and Marques-Silva, 2014; Devriendt, Gocht, Demirovic, Nordström, and Stuckey, 2021; Gange, Berg, Demirovic, and Stuckey, 2020]

i) for every core κ, introduce $\bar{o}_k \geq \sum_{b\kappa} b \leq (k-1)$.
ii) remove each $b \in \kappa$ from assumptions.
iii) assume $o_2 = 0$ (*).

PMRES [Narodytska and Bacchus, 2014]

i) for every $\kappa = \{b_1, \ldots, b_n\}$ add $b_i \wedge (b_{i+1} \vee \cdots \vee b_n) \leftrightarrow o_i$
ii) remove each $b \in \kappa$ from assumptions.
ii) assume $o_i = 0$ for each i.

MSU3 [Manquinho, Marques-Silva, and Planes, 2009]

i) maintain set InCore of variables that have been in a core.
ii) add $\bar{o} \rightarrow \sum_{b \in \text{InCore}} b \leq K$ where K is the number of cores.
iii) assume $\{o = 0\} \cup \{b = 0 \mid b \notin \text{InCore}\}$

There are many ways to relax

OLL [Andres, Kaufmann, Matheis, and Schaub, 2012; Morgado, Dodaro, and Marques-Silva, 2014; Devriendt, Gocht, Demirovic, Nordström, and Stuckey, 2021; Gange, Berg, Demirovic, and Stuckey, 2020]

i) for every core κ, introduce $\bar{o}_k \geq \sum_{b\kappa} b \leq (k-1)$.
ii) remove each $b \in \kappa$ from assumptions.
iii) assume $o_2 = 0$ (*).

PMRES [Narodytska and Bacchus, 2014]

i) for every $\kappa = \{b_1, \ldots, b_n\}$ add $b_i \wedge (b_{i+1} \vee \cdots \vee b_n) \leftrightarrow o_i$
ii) remove each $b \in \kappa$ from assumptions.
ii) assume $o_i = 0$ for each i.

MSU3 [Manquinho, Marques-Silva, and Planes, 2009]

i) maintain set INCORE of variables that have been in a core.
ii) add $\bar{o} \rightarrow \sum_{b \in \text{INCORE}} b \leq K$ where K is the number of cores.
iii) assume $\{o = 0\} \cup \{b = 0 \mid b \notin \text{INCORE}\}$

There are many ways to relax

OLL [Andres, Kaufmann, Matheis, and Schaub, 2012; Morgado, Dodaro, and Marques-Silva, 2014; Devriendt, Gocht, Demirovic, Nordström, and Stuckey, 2021; Gange, Berg, Demirovic, and Stuckey, 2020]

i) for every core κ, introduce $\bar{o}_k \geq \sum_{b\kappa} b \leq (k-1)$.
ii) remove each $b \in \kappa$ from assumptions.
iii) assume $o_2 = 0$ (*).

PMRES [Narodytska and Bacchus, 2014]

i) for every $\kappa = \{b_1, \ldots, b_n\}$ add $b_i \wedge (b_{i+1} \vee \cdots \vee b_n) \leftrightarrow o_i$
ii) remove each $b \in \kappa$ from assumptions.
ii) assume $o_i = 0$ for each i.

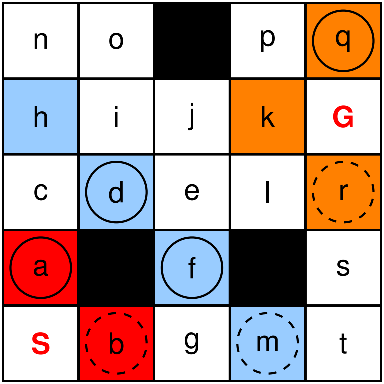
MSU3 [Manquinho, Marques-Silva, and Planes, 2009]

i) maintain set InCore of variables that have been in a core.
ii) add $\bar{o} \rightarrow \sum_{b \in \textsc{InCore}} b \leq K$ where K is the number of cores.
iii) assume $\{o = 0\} \cup \{b = 0 \mid b \notin \textsc{InCore}\}$

Implicit Hitting Set-Based Optimization

Definition: Hitting sets

- $\mathcal{K}$: a set of cores.
- hitting set *hs* - a solution of $\mathcal{K}$



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

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

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

$\mathcal{K} = \{\kappa^1, \kappa^2, \kappa^3\}$

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

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

Constraint Optimization with Hitting Sets

[Davies and Bacchus, 2011; Smirnov, Berg, and Järvisalo, 2021; Davies, Gange, and Stuckey, 2017]

Intuition

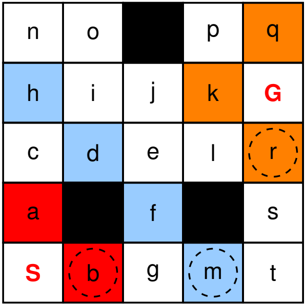
- Set of solutions to cores $\supset$ set of solutions to the instance.
- Minimum cost hitting sets over all cores are also minimum cost solutions of instance.
- Central insight - we do not need every core.
  - $cost(hs) \leq$ OPT-COST($\mathcal{F}$) for min-cost hs over *any* set of CORES

Constraint Optimization with Hitting Sets

[Davies and Bacchus, 2011; Smirnov, Berg, and Järvisalo, 2021; Davies, Gange, and Stuckey, 2017]

Intuition

- Set of solutions to cores $\supset$ set of solutions to the instance.
- Minimum cost hitting sets over all cores are also minimum cost solutions of instance.
- Central insight - we do not need every core.
  - $cost(hs) \leq \text{OPT-COST}(\mathcal{F})$ for min-cost hs over *any* set of CORES



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

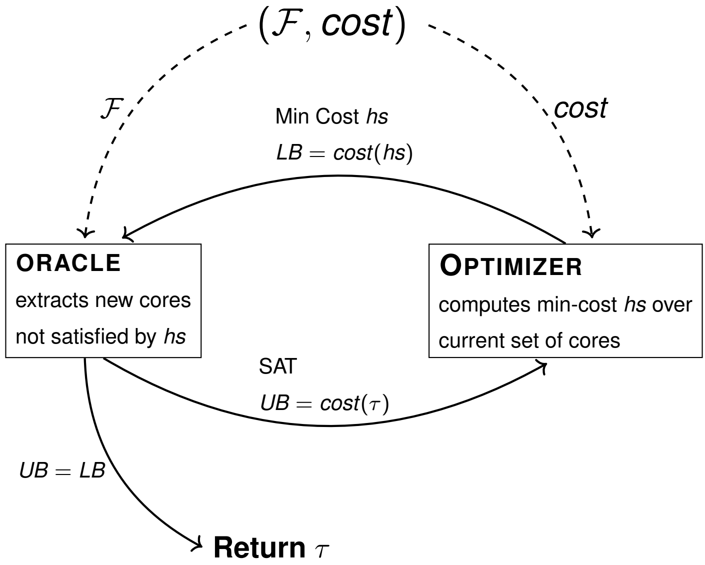
$\kappa^2 = \{h, d, f, m\}$

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

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

$hs = \{b, m, r\} \quad cost(hs) = 3 \leq 6 = \text{OPT-COST}(\mathcal{F})$

The IHS Approach



IHS

Shortest path

Intuition

1. Initialise $\mathcal{K} = \emptyset$
2. Check ORACLE($\mathcal{F}$, *hs*) for a min-cost $\mathcal{K}$-hitting set *hs*.
3. If not, obtain new core
4. Otherwise, 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

Intuition

1. Initialise $\mathcal{K} = \emptyset$
2. Check ORACLE$(\mathcal{F}, hs)$ for a min-cost $\mathcal{K}$-hitting set *hs*.
3. If not, obtain new core
4. Otherwise, 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 $= 0$, $\mathcal{K} = \emptyset$
$hs = \text{MIN-COST-HS}(\mathcal{K}, cost) = \emptyset$

ORACLE$(\mathcal{F}, hs)$

Is there a path that only visits squares in *hs*

IHS

Shortest path

Intuition

1. Initialise $\mathcal{K} = \emptyset$
2. Check ORACLE$(\mathcal{F}, hs)$ for a min-cost $\mathcal{K}$-hitting set hs.
3. If not, obtain new core
4. Otherwise, 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 $= 0$, $\mathcal{K} = \emptyset$
$hs = \text{MIN-COST-HS}(\mathcal{K}, cost) = \emptyset$

ORACLE$(\mathcal{F}, hs)$

Formula is unsatisfiable
Obtain new core: $\kappa_0 = a + b \geq 1$

IHS

Shortest path

Intuition

1. Initialise $\mathcal{K} = \emptyset$
2. Check ORACLE$(\mathcal{F}, hs)$ for a min-cost $\mathcal{K}$-hitting set hs.
3. If not, obtain new core
4. Otherwise, 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

$\text{LB} = 1$, $\mathcal{K} = \{\kappa_0\}$
$hs = \text{MIN-COST-HS}(\mathcal{K}, cost) = \{b\}$

ORACLE$(\mathcal{F}, hs)$

Is there a path that only visits squares in *hs*

Intuition

1. Initialise $\mathcal{K} = \emptyset$
2. Check ORACLE$(\mathcal{F}, hs)$ for a min-cost $\mathcal{K}$-hitting set hs.
3. If not, obtain new core
4. Otherwise, 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 $= 1$, $\mathcal{K} = \{\kappa_0\}$
$hs = \text{MIN-COST-HS}(\mathcal{K}, cost) = \{b\}$

ORACLE$(\mathcal{F}, hs)$

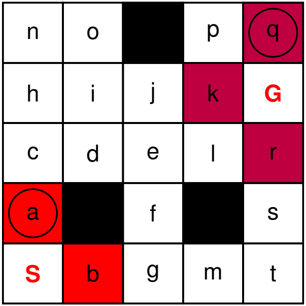
Formula is unsatisfiable
Obtain new core: $\kappa_1 = q + k + r \geq 1$

IHS

Shortest path

Intuition

1. Initialise $\mathcal{K} = \emptyset$
2. Check $\textsc{oracle}(\mathcal{F}, hs)$ for a min-cost $\mathcal{K}$-hitting set hs.
3. If not, obtain new core
4. Otherwise, solution is optimal



$\text{LB} = 2$, $\mathcal{K} = \{\kappa_0, \kappa_1\}$
$hs = \textsc{min-cost-hs}(\mathcal{K}, cost) = \{a, q\}$

$\textsc{oracle}(\mathcal{F}, hs)$

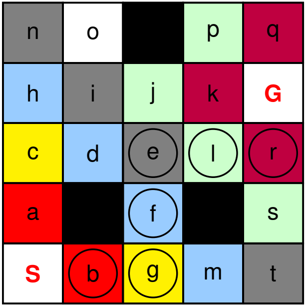
Is there a path that only visits squares in hs

IHS

Shortest path

Intuition

1. Initialise $\mathcal{K} = \emptyset$
2. Check ORACLE$(\mathcal{F}, hs)$ for a min-cost $\mathcal{K}$-hitting set hs.
3. If not, obtain new core
4. Otherwise, solution is optimal



$\mathrm{LB} = 6$, $\mathcal{K} = \{\kappa_0, \kappa_1, \kappa_2, \kappa_3, \kappa_4, \kappa_5\}$
$hs = \text{MIN-COST-HS}(\mathcal{K}, cost) = \{b, g, f, e, l, r\}$

ORACLE$(\mathcal{F}, hs)$

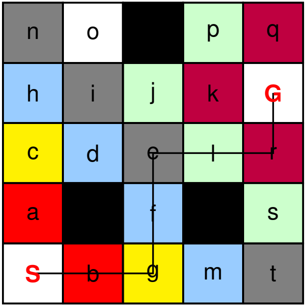
Is there a path that only visits squares in hs

IHS

Shortest path

Intuition

1. Initialise $\mathcal{K} = \emptyset$
2. Check $\textsc{oracle}(\mathcal{F}, hs)$ for a min-cost $\mathcal{K}$-hitting set *hs*.
3. If not, obtain new core
4. Otherwise, solution is optimal



LB $= 6$, $\mathcal{K} = \{\kappa_0, \kappa_1, \kappa_2, \kappa_3, \kappa_4, \kappa_5\}$
$hs = \textsc{min-cost-hs}(\mathcal{K}, cost) = \{b, g, f, e, l, r\}$

$\textsc{oracle}(\mathcal{F}, hs)$

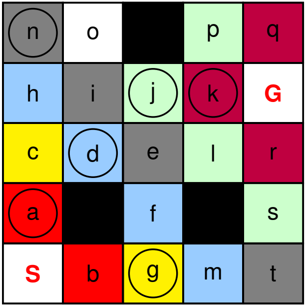
Formula is satisfiable
Obtain optimal $\tau = \{b, \ldots, l, r\}$
$cost(\tau) = 6$

IHS

Shortest path

Intuition

1. Initialise $\mathcal{K} = \emptyset$
2. Check $\textsc{oracle}(\mathcal{F}, hs)$ for a min-cost $\mathcal{K}$-hitting set hs.
3. If not, obtain new core
4. Otherwise, solution is optimal



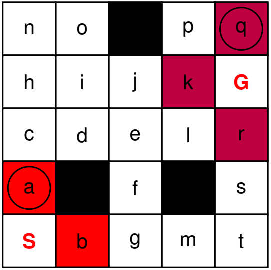
$\text{LB} = 6$, $\mathcal{K} = \{\kappa_0, \kappa_1, \kappa_2, \kappa_3, \kappa_4, \kappa_5\}$
$hs = \textsc{min-cost-hs}(\mathcal{K}, cost) = \{a, g, d, n, j, k\}$

$\textsc{oracle}(\mathcal{F}, hs)$

Formula is unsatisfiable
Obtain new core: $\kappa_6 = b + c \geq 1$

Everything is Hitting Sets over Cores

IHS
Slightly rephrased

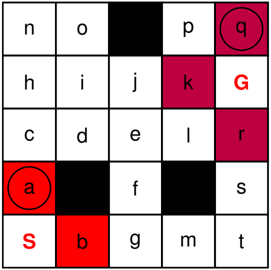


$\text{LB} = 2, \mathcal{K} = \{\kappa_0, \kappa_1\}$

$hs = \text{MIN-COST-HS}(\mathcal{K}, cost) = \{x = 0 \mid x \neq a, q\}$

IHS

Slightly rephrased



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

$hs = \text{MIN-COST-HS}(\mathcal{K}, cost) = \{x = 0 \mid x \neq a, q\}$

Hitting Sets of $\mathcal{K}$

$\{a, q\}$
$\{a, k\}$
$\{a, r\}$
$\{b, q\}$
$\{b, k\}$
$\{b, r\}$

IHS

Slightly rephrased

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

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

$hs = \text{MIN-COST-HS}(\mathcal{K}, cost) = \{x = 0 \mid x \neq a, q\}$

Hitting Sets of $\mathcal{K}$

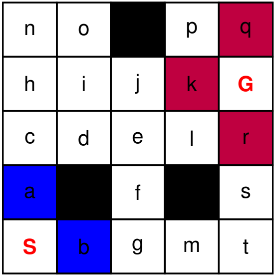
$\{a, q\}$
$\{a, k\}$
$\{a, r\}$
$\{b, q\}$
$\{b, k\}$
$\{b, r\}$

Extensions of *hs* to sols of $\mathcal{K}$

$\{a, q\}$

Hitting Sets in Core Guided Search

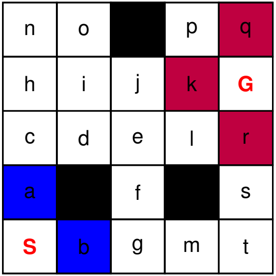
$\text{LB} = 2, \mathcal{K} = \{\kappa_0, \kappa_1\}$



$$\mathcal{F}^i \equiv \mathcal{F} \land \textsc{Extra-Constraints}$$
$$= \mathcal{F} \land \bigwedge_{\kappa \in \mathcal{K}} \left(\bar{o}_2^{\kappa} \rightarrow \sum_{b \in \kappa} b \leq 1 \right)$$

$$hs^i \equiv \{o_2^{\kappa_0} = 0, o_2^{\kappa_1} = 0\} \cup \{c = 0, d = 0 \ldots\}$$

Hitting Sets in Core Guided Search



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

$$\mathcal{F}^i \equiv \mathcal{F} \wedge \textsc{Extra-Constraints}$$
$$= \mathcal{F} \wedge \bigwedge_{\kappa \in \mathcal{K}} \left(\bar{o}_2^{\kappa} \rightarrow \sum_{b \in \kappa} b \leq 1 \right)$$

$$hs^i \equiv \{o_2^{\kappa_0} = 0, o_2^{\kappa_1} = 0\} \cup \{c = 0, d = 0 \ldots\}$$

Hitting Sets of $\mathcal{K}$

$\{a, q\}$
$\{a, k\}$
$\{a, r\}$
$\{b, q\}$
$\{b, k\}$
$\{b, r\}$

Hitting Sets in Core Guided Search

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

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

$$\mathcal{F}^i \equiv \mathcal{F} \wedge \textsc{Extra-Constraints}$$
$$= \mathcal{F} \wedge \bigwedge_{\kappa \in \mathcal{K}} \left(\bar{o}_2^{\kappa} \rightarrow \sum_{b \in \kappa} b \leq 1 \right)$$

$$hs^i \equiv \{o_2^{\kappa_0} = 0, o_2^{\kappa_1} = 0\} \cup \{c = 0, d = 0 \dots\}$$

Hitting Sets of $\mathcal{K}$	**Extensions of hs^i to sols of $\mathcal{K} \wedge$ Extra-Constraints**
$\{a, q\}$	$\{a, q\}$
$\{a, k\}$	$\{a, k\}$
$\{a, r\}$	$\{a, r\}$
$\{b, q\}$	$\{b, q\}$
$\{b, k\}$	$\{b, k\}$
$\{b, r\}$	$\{b, r\}$

Cores and Hitting Sets in Solution Improving Search

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

$\textsc{Oracle}(\mathcal{F}^i, \emptyset)$

$\mathcal{F}^i = \mathcal{F} \wedge (\textsc{CostLessThan}(k))$

Cores and Hitting Sets in Solution Improving Search

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

$\textsc{oracle}(\mathcal{F}^i, \emptyset)$

$\mathcal{F}^i = \mathcal{F} \land (\textsc{CostLessThan}(k))$

$\textsc{oracle}(\mathcal{F}^i, hs^i)$

$\mathcal{F}^i = \mathcal{F} \land (\bar{o}_k \rightarrow \textsc{CostLessThan}(k))$

$hs^i \equiv \{o_k = 0\}$

Cores and Hitting Sets in Solution Improving Search

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

$\textsc{oracle}(\mathcal{F}^i, \emptyset)$

$\mathcal{F}^i = \mathcal{F} \wedge (\textsc{CostLessThan}(k))$

$\textsc{oracle}(\mathcal{F}^i, hs^i)$

$\mathcal{F}^i = \mathcal{F} \wedge (\bar{o}_k \rightarrow \textsc{CostLessThan}(k))$

$hs^i \equiv \{o_k = 0\}$

$(o_k \geq 1)$ a core of

$\mathcal{F} \wedge (\bar{o}_k \rightarrow \textsc{CostLessThan}(k))$ iff

$\textsc{opt}(\mathcal{F}) \geq k$

Intuition Underlying UniMaxSAT:

IHS Solvers

1) Extract cores over $\mathcal{F}$
2) Impose one hitting set of those cores as assumptions.

Core-Guided Solvers

1) Extract cores over $\mathcal{F} \land \textsc{Extra-Constraints}$
2) Impose all hitting sets of those cores (*).

Solution Improving Search

1) Extract unit cores over $\mathcal{F} \land (\bar{o}_k \rightarrow \textsc{CostLessThan}(k))$.
2) Impose a (trivial) hitting set over all those cores.

Intuition Underlying UniMaxSAT:

IHS Solvers

1) Extract cores over $\mathcal{F}$
2) Impose one hitting set of those cores as assumptions.

Core-Guided Solvers

1) Extract cores over $\mathcal{F} \wedge \textsc{Extra-Constraints}$
2) Impose all hitting sets of those cores (*).

Solution Improving Search

1) Extract unit cores over $\mathcal{F} \wedge (\bar{o}_k \rightarrow \textsc{CostLessThan}(k))$.
2) Impose a (trivial) hitting set over all those cores.

Intuition Underlying UniMaxSAT:

IHS Solvers

1) Extract cores over $\mathcal{F} \wedge \emptyset$
2) Impose one hitting set of those cores as assumptions.

Core-Guided Solvers

1) Extract cores over $\mathcal{F} \wedge$ EXTRA-CONSTRAINTS
2) Impose all hitting sets of those cores (*).

Solution Improving Search

1) Extract unit cores over $\mathcal{F} \wedge (\bar{o}_k \rightarrow \textsc{CostLessThan}(k))$.
2) Impose a (trivial) hitting set over all those cores.

Intuition Underlying UniMaxSAT:

IHS Solvers

1) Extract cores over $\mathcal{F} \land \emptyset$
2) Impose one hitting set of those cores as assumptions.

Core-Guided Solvers

1) Extract cores over $\mathcal{F} \land \textsc{Extra-Constraints}$
2) Impose all hitting sets of those cores (*).

Solution Improving Search

1) Extract unit cores over $\mathcal{F} \land (\bar{o}_k \rightarrow \textsc{CostLessThan}(k))$.
2) Impose a (trivial) hitting set over all those cores.

Summary

- Many state-of-the-art constraint optimization solvers reduce optimization into a sequence of decisions.
- On the surface, these algorithms appear very different.
- Our work: Constraint-agnostic model that captures all instantiations of core-guided, IHS, and SIS solvers we are aware of.
  - Unified proofs of correctness
  - New insights into their relationship
  - Basis for new algorithmic instantiations.

UniMaxSAT in (a bit more) detail

High Level Idea

for solving ($\mathcal{F}$, *cost*)

- Maintain EXTRA-CONSTRAINTS and $\mathcal{K}$.
- Iterate:
  1. Compute min-cost hitting set γ of $\mathcal{K} \wedge$ EXTRA-CONSTRAINTS.
  2. Check if γ can be extended to a solution of $\mathcal{F}$.
  3. If so, return found solution.
  4. Else obtain a new core of $\mathcal{F} \wedge$ EXTRA-CONSTRAINTS
  5. (Optionally) extend EXTRA-CONSTRAINTS based on core.

Challenges:

Termination

Optimality of Solution

High Level Idea

for solving ($\mathcal{F}$, *cost*)

- Maintain EXTRA-CONSTRAINTS and $\mathcal{K}$.
- Iterate:
  1. Compute min-cost hitting set γ of $\mathcal{K} \wedge$ EXTRA-CONSTRAINTS.
  2. Check if γ can be extended to a solution of $\mathcal{F}$.
  3. If so, return found solution.
  4. Else obtain a new core of $\mathcal{F} \wedge$ EXTRA-CONSTRAINTS
  5. (Optionally) extend EXTRA-CONSTRAINTS based on core.

Challenges:

Termination

Optimality of Solution

High Level Idea

for solving ($\mathcal{F}$, *cost*)

- Maintain EXTRA-CONSTRAINTS and $\mathcal{K}$.
- Iterate:
  1. Compute min-cost hitting set γ of $\mathcal{K} \wedge$ EXTRA-CONSTRAINTS.
  2. Check if γ can be extended to a solution of $\mathcal{F}$.
  3. If so, return found solution.
  4. Else obtain a new core of $\mathcal{F} \wedge$ EXTRA-CONSTRAINTS
  5. (Optionally) extend EXTRA-CONSTRAINTS based on core.

Challenges:

Termination

Optimality of Solution

High Level Idea

for solving ($\mathcal{F}$, *cost*)

- Maintain EXTRA-CONSTRAINTS and $\mathcal{K}$.
- Iterate:
  1. Compute min-cost hitting set γ of $\mathcal{K} \wedge$ EXTRA-CONSTRAINTS.
  2. Check if γ can be extended to a solution of $\mathcal{F}$.
  3. If so, return found solution.
  4. Else obtain a new core of $\mathcal{F} \wedge$ EXTRA-CONSTRAINTS
  5. (Optionally) extend EXTRA-CONSTRAINTS based on core.

Challenges:

Termination

Optimality of Solution

High Level Idea

for solving ($\mathcal{F}$, *cost*)

- Maintain EXTRA-CONSTRAINTS and $\mathcal{K}$.
- Iterate:
  1. Compute min-cost hitting set γ of $\mathcal{K} \wedge$ EXTRA-CONSTRAINTS.
  2. Check if γ can be extended to a solution of $\mathcal{F}$.
  3. If so, return found solution.
  4. Else obtain a new core of $\mathcal{F} \wedge$ EXTRA-CONSTRAINTS
  5. (Optionally) extend EXTRA-CONSTRAINTS based on core.

Challenges:

Termination

Optimality of Solution

High Level Idea

for solving ($\mathcal{F}$, *cost*)

- Maintain EXTRA-CONSTRAINTS and $\mathcal{K}$.
- Iterate:
  1. Compute min-cost hitting set γ of $\mathcal{K} \wedge$ EXTRA-CONSTRAINTS.
  2. Check if γ can be extended to a solution of $\mathcal{F}$.
  3. If so, return found solution.
  4. Else obtain a new core of $\mathcal{F} \wedge$ EXTRA-CONSTRAINTS
  5. (Optionally) extend EXTRA-CONSTRAINTS based on core.

Challenges:

Termination

Optimality of Solution

High Level Idea

for solving ($\mathcal{F}$, *cost*)

- Maintain EXTRA-CONSTRAINTS and $\mathcal{K}$.
- Iterate:
  1. Compute min-cost hitting set γ of $\mathcal{K} \wedge$ EXTRA-CONSTRAINTS.
  2. Check if γ can be extended to a solution of $\mathcal{F}$.
  3. If so, return found solution.
  4. Else obtain a new core of $\mathcal{F} \wedge$ EXTRA-CONSTRAINTS
  5. (Optionally) extend EXTRA-CONSTRAINTS based on core.

Challenges:

Termination

Optimality of Solution

High Level Idea

for solving ($\mathcal{F}$, *cost*)

- Maintain EXTRA-CONSTRAINTS and $\mathcal{K}$.
- Iterate:
  1. Compute min-cost hitting set γ of $\mathcal{K} \wedge$ EXTRA-CONSTRAINTS.
  2. Check if γ can be extended to a solution of $\mathcal{F}$.
  3. If so, return found solution.
  4. Else obtain a new core of $\mathcal{F} \wedge$ EXTRA-CONSTRAINTS
  5. (Optionally) extend EXTRA-CONSTRAINTS based on core.

Challenges:

Termination

Optimality of Solution

High Level Idea

for solving ($\mathcal{F}$, *cost*)

- Maintain EXTRA-CONSTRAINTS and $\mathcal{K}$.
- Iterate:
  1. Compute min-cost hitting set γ of $\mathcal{K} \wedge$ EXTRA-CONSTRAINTS.
  2. Check if γ can be extended to a solution of $\mathcal{F}$.
  3. If so, return found solution.
  4. Else obtain a new core of $\mathcal{F} \wedge$ EXTRA-CONSTRAINTS
  5. (Optionally) extend EXTRA-CONSTRAINTS based on core.

Challenges:

Termination
Optimality of Solution

Formalizing EXTRA-CONSTRAINTS

Abstraction Sets

$$(in, D, out)$$

(Abstract) Core

A constraint κ satisfied by all solutions to $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$

Formalizing EXTRA-CONSTRAINTS

Abstraction Sets

a set of
input
literals

$$(in, D, out)$$

(Abstract) Core

A constraint κ satisfied by all solutions to $\mathcal{F} \land \text{DEF}(\mathcal{AB})$

Formalizing EXTRA-CONSTRAINTS

Abstraction Sets

a set of input literals

a set of definition constraints

$$(in, D, out)$$

(Abstract) Core

A constraint κ satisfied by all solutions to $\mathcal{F} \wedge \mathrm{DEF}(\mathcal{AB})$

Formalizing EXTRA-CONSTRAINTS
Abstraction Sets

a set of **input** literals

a set of **definition** constraints

a set of **output** literals

$$(in, D, out)$$

(Abstract) Core

A constraint κ satisfied by all solutions to $\mathcal{F} \wedge \mathrm{DEF}(\mathcal{AB})$

Formalizing EXTRA-CONSTRAINTS

Abstraction Sets

a set of input literals

a set of definition constraints

a set of output literals

$$(in, D, out)$$

$\mathcal{AB}$, a collection of abstraction sets

DEF($\mathcal{AB}$), the set of all definitions

(Abstract) Core

A constraint κ satisfied by all solutions to $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$

Formalizing EXTRA-CONSTRAINTS

Abstraction Sets

a set of input literals

a set of definition constraints

a set of output literals

$$(in, D, out)$$

$\mathcal{AB}$, a collection of abstraction sets
DEF($\mathcal{AB}$), the set of all definitions

(Abstract) Core

A constraint κ satisfied by all solutions to $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$

Formalizing Hitting Sets

Abstract Candidates

- Instance $(\mathcal{F}, \mathit{cost})$, abstraction sets $\mathcal{AB}$, (abstract) cores $\mathcal{K}$.
- A (partial) assignment γ is a $(\mathcal{AB}, \mathcal{K})$-abstract candidate if:
  - γ extends to at least one solution of $\mathcal{F} \wedge \mathrm{DEF}(\mathcal{AB})$
  - All such extensions are minimum cost.

Example

- $\mathcal{F} = \{(b_1 \vee b_2, \vee x), (\bar{x} \vee b_3), (b_3 \vee b_4 \vee b_5)\}$,
- $\mathit{cost} = b_1 + b_2 + 3b_3 + b_4 + 2b_5$
- $\mathcal{K} = \{(b_1 \vee b_2 \vee b_3 \vee b_4 \vee b_5)\}$
- $\mathrm{AB} = (\{b_1, b_2, b_3, b_4, b_5\}, \sum_{i=1}^{5} b_i \leq 2 \leftrightarrow \bar{o}_2, \{o_2\})$
- $\{\bar{b}_3, \bar{b}_5, \bar{o}_2\}$ & $\{\bar{b}_1, \bar{b}_3, \bar{b}_5, \bar{o}_2\}$ are abstract candidates
- $\{\bar{b}_5, \bar{o}_2\}$ is not.

Formalizing Hitting Sets

Abstract Candidates

- Instance $(\mathcal{F}, cost)$, abstraction sets $\mathcal{AB}$, (abstract) cores $\mathcal{K}$.
- A (partial) assignment γ is a $(\mathcal{AB}, \mathcal{K})$-abstract candidate if:
  - γ extends to at least one solution of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$
  - All such extensions are minimum cost.

Example

- $\mathcal{F} = \{(b_1 \vee b_2, \vee x), (\bar{x} \vee b_3), (b_3 \vee b_4 \vee b_5)\}$,
- $cost = b_1 + b_2 + 3b_3 + b_4 + 2b_5$
- $\mathcal{K} = \{(b_1 \vee b_2 \vee b_3 \vee b_4 \vee b_5)\}$
- $\text{AB} = (\{b_1, b_2, b_3, b_4, b_5\}, \sum_{i=1}^{5} b_i \leq 2 \leftrightarrow \bar{o}_2, \{o_2\})$
- $\{\bar{b}_3, \bar{b}_5, \bar{o}_2\}$ & $\{\bar{b}_1, \bar{b}_3, \bar{b}_5, \bar{o}_2\}$ are abstract candidates
- $\{\bar{b}_5, \bar{o}_2\}$ is not.

Formalizing Hitting Sets

Abstract Candidates

- Instance $(\mathcal{F}, cost)$, abstraction sets $\mathcal{AB}$, (abstract) cores $\mathcal{K}$.
- A (partial) assignment γ is a $(\mathcal{AB}, \mathcal{K})$-abstract candidate if:
  - γ extends to at least one solution of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$
  - All such extensions are minimum cost.

Example

- $\mathcal{F} = \{(b_1 \vee b_2, \vee x), (\bar{x} \vee b_3), (b_3 \vee b_4 \vee b_5)\}$,
- $cost = b_1 + b_2 + 3b_3 + b_4 + 2b_5$
- $\mathcal{K} = \{(b_1 \vee b_2 \vee b_3 \vee b_4 \vee b_5)\}$
- $\text{AB} = (\{b_1, b_2, b_3, b_4, b_5\}, \sum_{i=1}^{5} b_i \leq 2 \leftrightarrow \bar{o}_2, \{o_2\})$
- $\{\bar{b}_3, \bar{b}_5, \bar{o}_2\}$ & $\{\bar{b}_1, \bar{b}_3, \bar{b}_5, \bar{o}_2\}$ are abstract candidates
- $\{\bar{b}_5, \bar{o}_2\}$ is not.

Formalizing Hitting Sets

Abstract Candidates

- Instance $(\mathcal{F}, cost)$, abstraction sets $\mathcal{AB}$, (abstract) cores $\mathcal{K}$.
- A (partial) assignment γ is a $(\mathcal{AB}, \mathcal{K})$-abstract candidate if:
  - γ extends to at least one solution of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$
  - All such extensions are minimum cost.

Example

- $\mathcal{F} = \{(b_1 \vee b_2, \vee x), (\bar{x} \vee b_3), (b_3 \vee b_4 \vee b_5)\}$,
- $cost = b_1 + b_2 + 3b_3 + b_4 + 2b_5$
- $\mathcal{K} = \{(b_1 \vee b_2 \vee b_3 \vee b_4 \vee b_5)\}$
- $\text{AB} = (\{b_1, b_2, b_3, b_4, b_5\}, \sum_{i=1}^{5} b_i \leq 2 \leftrightarrow \bar{o}_2, \{o_2\})$
- $\{\bar{b}_3, \bar{b}_5, \bar{o}_2\}$ & $\{\bar{b}_1, \bar{b}_3, \bar{b}_5, \bar{o}_2\}$ are abstract candidates
- $\{\bar{b}_5, \bar{o}_2\}$ is not.

Formalizing Hitting Sets
Abstract Candidates

- Instance $(\mathcal{F}, \mathit{cost})$, abstraction sets $\mathcal{AB}$, (abstract) cores $\mathcal{K}$.
- A (partial) assignment γ is a $(\mathcal{AB}, \mathcal{K})$-abstract candidate if:
  - γ extends to at least one solution of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$
  - All such extensions are minimum cost.

Example

- $\mathcal{F} = \{(b_1 \vee b_2, \vee x), (\bar{x} \vee b_3), (b_3 \vee b_4 \vee b_5)\}$,
- $\mathit{cost} = b_1 + b_2 + 3b_3 + b_4 + 2b_5$
- $\mathcal{K} = \{(b_1 \vee b_2 \vee b_3 \vee b_4 \vee b_5)\}$
- $\text{AB} = (\{b_1, b_2, b_3, b_4, b_5\}, \sum_{i=1}^{5} b_i \leq 2 \leftrightarrow \bar{o}_2, \{o_2\})$
- $\{\bar{b}_3, \bar{b}_5, \bar{o}_2\}$ & $\{\bar{b}_1, \bar{b}_3, \bar{b}_5, \bar{o}_2\}$ are abstract candidates
- $\{\bar{b}_5, \bar{o}_2\}$ is not.

Formalizing Hitting Sets

Abstract Candidates

- Instance $(\mathcal{F}, cost)$, abstraction sets $\mathcal{AB}$, (abstract) cores $\mathcal{K}$.
- A (partial) assignment γ is a $(\mathcal{AB}, \mathcal{K})$-abstract candidate if:
  - γ extends to at least one solution of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$
  - All such extensions are minimum cost.

Example

- $\mathcal{F} = \{(b_1 \vee b_2, \vee x), (\bar{x} \vee b_3), (b_3 \vee b_4 \vee b_5)\}$,
- $cost = b_1 + b_2 + 3b_3 + b_4 + 2b_5$
- $\mathcal{K} = \{(b_1 \vee b_2 \vee b_3 \vee b_4 \vee b_5)\}$
- $\text{AB} = (\{b_1, b_2, b_3, b_4, b_5\}, \sum_{i=1}^{5} b_i \leq 2 \leftrightarrow \bar{o}_2, \{o_2\})$
- $\{\bar{b}_3, \bar{b}_5, \bar{o}_2\}$ & $\{\bar{b}_1, \bar{b}_3, \bar{b}_5, \bar{o}_2\}$ are abstract candidates
- $\{\bar{b}_5, \bar{o}_2\}$ is not.

Formalizing Hitting Sets

Abstract Candidates

- Instance $(\mathcal{F}, cost)$, abstraction sets $\mathcal{AB}$, (abstract) cores $\mathcal{K}$.
- A (partial) assignment γ is a $(\mathcal{AB}, \mathcal{K})$-abstract candidate if:
  - γ extends to at least one solution of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$
  - All such extensions are minimum cost.

Example

- $\mathcal{F} = \{(b_1 \vee b_2, \vee x), (\bar{x} \vee b_3), (b_3 \vee b_4 \vee b_5)\}$,
- $cost = b_1 + b_2 + 3b_3 + b_4 + 2b_5$
- $\mathcal{K} = \{(b_1 \vee b_2 \vee b_3 \vee b_4 \vee b_5)\}$
- $\text{AB} = (\{b_1, b_2, b_3, b_4, b_5\}, \sum_{i=1}^{5} b_i \leq 2 \leftrightarrow \bar{o}_2, \{o_2\})$
- $\{\bar{b}_3, \bar{b}_5, \bar{o}_2\}$ & $\{\bar{b}_1, \bar{b}_3, \bar{b}_5, \bar{o}_2\}$ are abstract candidates
- $\{\bar{b}_5, \bar{o}_2\}$ is not.

Formalizing Hitting Sets

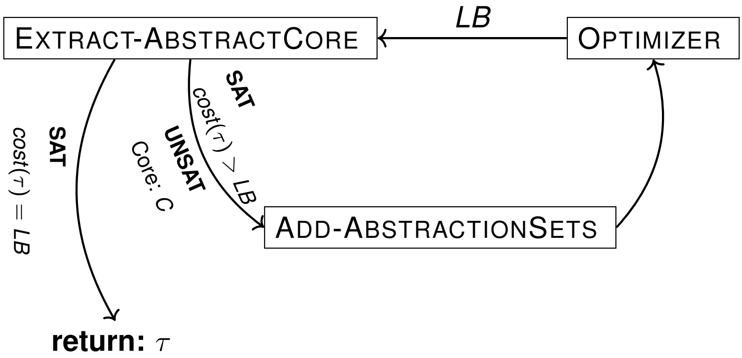
Abstract Candidates

- Instance ($\mathcal{F}$, *cost*), abstraction sets $\mathcal{AB}$, (abstract) cores $\mathcal{K}$.
- A (partial) assignment γ is a $(\mathcal{AB}, \mathcal{K})$-abstract candidate if:
  - γ extends to at least one solution of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$
  - All such extensions are minimum cost.

Example

- $\mathcal{F} = \{(b_1 \vee b_2, \vee x), (\bar{x} \vee b_3), (b_3 \vee b_4 \vee b_5)\}$,
- $cost = b_1 + b_2 + 3b_3 + b_4 + 2b_5$
- $\mathcal{K} = \{(b_1 \vee b_2 \vee b_3 \vee b_4 \vee b_5)\}$
- $\text{AB} = (\{b_1, b_2, b_3, b_4, b_5\}, \sum_{i=1}^{5} b_i \leq 2 \leftrightarrow \bar{o}_2, \{o_2\})$
- $\{\bar{b}_3, \bar{b}_5, \bar{o}_2\}$ & $\{\bar{b}_1, \bar{b}_3, \bar{b}_5, \bar{o}_2\}$ are abstract candidates
- $\{\bar{b}_5, \bar{o}_2\}$ is not.

UniMaxSAT


EXTRACT-ABSTRACTCORE
LB
OPTIMIZER
SAT
cost(τ) = LB
SAT
cost(τ) > LB
UNSAT
Core: C
ADD-ABSTRACTIONSETS
return: τ

Formalizing the EXTRACT-ABSTRACTCORE subroutine

- A formula $\mathcal{F}$, $\mathcal{AB}$ of abstraction sets and a partial assignment γ.
- EXTRACT-ABSTRACTCORE$(\mathcal{F}, \mathcal{AB}, \gamma)$ returns:
  - SAT and a solution τ of $\mathcal{F} \wedge$ DEF$(\mathcal{AB})$ that extends γ, or
  - UNSAT and an abstract core κ falsified by γ.

Formalizing the ADD-ABSTRACTIONSETS subroutine

Challenges

- How to ensure previous cores are not invalidated?
- How to ensure the preservation of optimal solutions?

Our solution - Feasible abstraction sets

$\mathcal{AB}$ is feasible for $\mathcal{F}$ if every solution to $\mathcal{F}$ can be uniquely extended to a solution of $\mathcal{K} \wedge \text{DEF}(\mathcal{AB})$.

In Practice

New abstraction sets can only share input variables.

Formalizing the ADD-ABSTRACTIONSETS subroutine

Challenges

- How to ensure previous cores are not invalidated?
- How to ensure the preservation of optimal solutions?

Our solution - Feasible abstraction sets

$\mathcal{AB}$ is feasible for $\mathcal{F}$ if every solution to $\mathcal{F}$ can be uniquely extended to a solution of $\mathcal{K} \wedge \text{DEF}(\mathcal{AB})$.

In Practice

New abstraction sets can only share input variables.

Formalizing the ADD-ABSTRACTIONSETS subroutine

Challenges

- How to ensure previous cores are not invalidated?
- How to ensure the preservation of optimal solutions?

Our solution - Feasible abstraction sets

$\mathcal{AB}$ is feasible for $\mathcal{F}$ if every solution to $\mathcal{F}$ can be uniquely extended to a solution of $\mathcal{K} \wedge \mathrm{DEF}(\mathcal{AB})$.

In Practice

New abstraction sets can only share input variables.

Formalizing the OPTIMIZER subroutine

Challenges

- How to ensure progress?
- How to ensure the detection of optimal solutions?

First Idea

OPTIMIZER($\mathcal{AB}, \mathcal{K}$, *cost*) returns an abstract candidate γ and the cost *LB* of its extension(s).

Formalizing the OPTIMIZER subroutine

Challenges

- How to ensure progress?
- How to ensure the detection of optimal solutions?

First Idea

OPTIMIZER($\mathcal{AB}, \mathcal{K}$, *cost*) returns an abstract candidate γ and the cost *LB* of its extension(s).

Formalizing the OPTIMIZER subroutine

Challenges

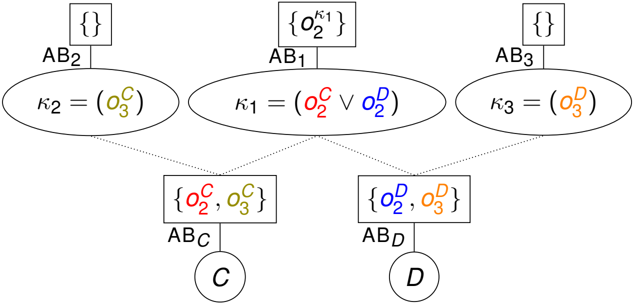
- How to ensure progress?
- How to ensure the detection of optimal solutions?

First Idea

OPTIMIZER($\mathcal{AB}, \mathcal{K}$, *cost*) returns an abstract candidate γ and the cost *LB* of its extension(s).
does not quite work.

Unexpected Observation

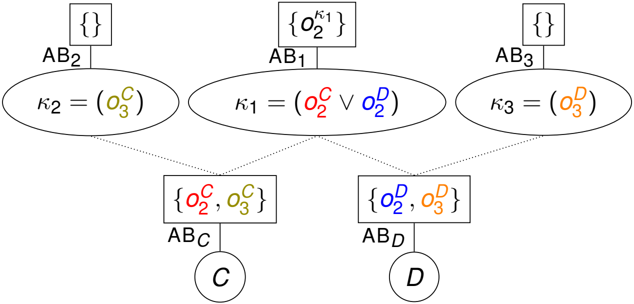
Core-guided solvers do not always compute abstract candidates



- C and D cores.
- $o_k^C \leftrightarrow \sum_{b \in C} b \geq k$.

Unexpected Observation

Core-guided solvers do not always compute abstract candidates



Theorem

If the assumptions γ used by OLL are extendable to a solution of $\mathcal{K} \wedge \text{DEF}(\mathcal{AB})$, then γ is an abstract candidate.

Formalizing the OPTIMIZER subroutine

Second Idea - Correctness Condition

- OPTIMIZER($\mathcal{AB}, \mathcal{K}$, *cost*) returns a partial assignment γ and any lower bound *LB*.
- For every iteration *i*, there should be a $k \geq 0$ s.t. OPTIMIZER($\mathcal{AB}, \mathcal{K}$, *cost*) returns an abstract candidate and its cost on iteration $i + k$.

UniMaxSAT - Summary
solving ($\mathcal{F}$, *cost*)

- Maintain $\mathcal{AB}$ and $\mathcal{K}$.
- Iterate:
  1. Compute (hopefully) an abstract candidate γ and lower bound *LB* over $\mathcal{K} \wedge \text{DEF}(\mathcal{AB})$.
  2. Check if γ can be extended to a solution τ of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$.
  3. If so and $cost(\tau) = LB$, return found solution.
  4. Else obtain a new (abstract) core
  5. (Optionally) extend $\mathcal{AB}$ with (feasible) abstraction sets.

Theorem:

Assume: (i) γ is an abstract candidate "sufficiently" often,
(ii) *LB* is a lower bound $\text{OPT}(\mathcal{F})$ and
(iii) if γ is abstract candidate, *LB* is the cost of its extensions.

UniMaxSAT - Summary

solving ($\mathcal{F}$, *cost*)

- Maintain $\mathcal{AB}$ and $\mathcal{K}$.
- Iterate:
  1. Compute (hopefully) an abstract candidate γ and lower bound LB over $\mathcal{K} \wedge \text{DEF}(\mathcal{AB})$.
  2. Check if γ can be extended to a solution τ of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$.
  3. If so and $cost(\tau) = LB$, return found solution.
  4. Else obtain a new (abstract) core
  5. (Optionally) extend $\mathcal{AB}$ with (feasible) abstraction sets.

Theorem:

Assume: (i) γ is an abstract candidate "sufficiently" often,
(ii) LB is a lower bound $\text{OPT}(\mathcal{F})$ and
(iii) if γ is abstract candidate, LB is the cost of its extensions.
UniMaxSAT returns an optimal solution of any MaxSAT instance.

UniMaxSAT - Summary

solving ($\mathcal{F}$, *cost*)

- Maintain $\mathcal{AB}$ and $\mathcal{K}$.
- Iterate:
  1. Compute (hopefully) an abstract candidate γ and lower bound *LB* over $\mathcal{K} \wedge \text{DEF}(\mathcal{AB})$.
  2. Check if γ can be extended to a solution τ of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$.
  3. If so and $cost(\tau) = LB$, return found solution.
  4. Else obtain a new (abstract) core
  5. (Optionally) extend $\mathcal{AB}$ with (feasible) abstraction sets.

Theorem:

Assume: (i) γ is an abstract candidate "sufficiently" often,
(ii) *LB* is a lower bound $\text{OPT}(\mathcal{F})$ and
(iii) if γ is abstract candidate, *LB* is the cost of its extensions.
UniMaxSAT returns an optimal solution of any MaxSAT instance.

UniMaxSAT - Summary

solving ($\mathcal{F}$, *cost*)

- Maintain $\mathcal{AB}$ and $\mathcal{K}$.
- Iterate:
  1. Compute (hopefully) an abstract candidate γ and lower bound *LB* over $\mathcal{K} \wedge \text{DEF}(\mathcal{AB})$.
  2. Check if γ can be extended to a solution τ of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$.
  3. If so and $cost(\tau) = LB$, return found solution.
  4. Else obtain a new (abstract) core
  5. (Optionally) extend $\mathcal{AB}$ with (feasible) abstraction sets.

Theorem:

Assume: (i) γ is an abstract candidate "sufficiently" often,
(ii) *LB* is a lower bound $\text{OPT}(\mathcal{F})$ and
(iii) if γ is abstract candidate, *LB* is the cost of its extensions.
UniMaxSAT returns an optimal solution of any MaxSAT instance.

UniMaxSAT - Summary

solving ($\mathcal{F}$, *cost*)

- Maintain $\mathcal{AB}$ and $\mathcal{K}$.
- Iterate:
  1. Compute (hopefully) an abstract candidate γ and lower bound *LB* over $\mathcal{K} \wedge \text{DEF}(\mathcal{AB})$.
  2. Check if γ can be extended to a solution τ of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$.
  3. If so and $cost(\tau) = LB$, return found solution.
  4. Else obtain a new (abstract) core
  5. (Optionally) extend $\mathcal{AB}$ with (feasible) abstraction sets.

Theorem:

Assume: (i) γ is an abstract candidate "sufficiently" often,
(ii) *LB* is a lower bound $\text{OPT}(\mathcal{F})$ and
(iii) if γ is abstract candidate, *LB* is the cost of its extensions.
UniMaxSAT returns an optimal solution of any MaxSAT instance.

UniMaxSAT - Summary

solving ($\mathcal{F}$, *cost*)

- Maintain $\mathcal{AB}$ and $\mathcal{K}$.
- Iterate:
  1. Compute (hopefully) an abstract candidate γ and lower bound *LB* over $\mathcal{K} \wedge \text{DEF}(\mathcal{AB})$.
  2. Check if γ can be extended to a solution τ of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$.
  3. If so and $cost(\tau) = LB$, return found solution.
  4. Else obtain a new (abstract) core
  5. (Optionally) extend $\mathcal{AB}$ with (feasible) abstraction sets.

Theorem:

Assume: (i) γ is an abstract candidate ”sufficiently” often,
(ii) *LB* is a lower bound $\text{OPT}(\mathcal{F})$ and
(iii) if γ is abstract candidate, *LB* is the cost of its extensions.
UniMaxSAT returns an optimal solution of any MaxSAT instance.

UniMaxSAT - Summary

solving ($\mathcal{F}$, *cost*)

- Maintain $\mathcal{AB}$ and $\mathcal{K}$.
- Iterate:
  1. Compute (hopefully) an abstract candidate γ and lower bound *LB* over $\mathcal{K} \wedge \text{DEF}(\mathcal{AB})$.
  2. Check if γ can be extended to a solution τ of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$.
  3. If so and $\textit{cost}(\tau) = \textit{LB}$, return found solution.
  4. Else obtain a new (abstract) core
  5. (Optionally) extend $\mathcal{AB}$ with (feasible) abstraction sets.

Theorem:

Assume: (i) γ is an abstract candidate "sufficiently" often,
(ii) *LB* is a lower bound $\text{OPT}(\mathcal{F})$ and
(iii) if γ is abstract candidate, *LB* is the cost of its extensions.
UniMaxSAT returns an optimal solution of any MaxSAT instance.

UniMaxSAT - Summary

solving ($\mathcal{F}$, *cost*)

- Maintain $\mathcal{AB}$ and $\mathcal{K}$.
- Iterate:
  1. Compute (hopefully) an abstract candidate γ and lower bound *LB* over $\mathcal{K} \wedge \text{DEF}(\mathcal{AB})$.
  2. Check if γ can be extended to a solution τ of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$.
  3. If so and $cost(\tau) = LB$, return found solution.
  4. Else obtain a new (abstract) core
  5. (Optionally) extend $\mathcal{AB}$ with (feasible) abstraction sets.

Theorem:

Assume: (i) γ is an abstract candidate "sufficiently" often,
(ii) *LB* is a lower bound $\text{OPT}(\mathcal{F})$ and
(iii) if γ is abstract candidate, *LB* is the cost of its extensions.
UniMaxSAT returns an optimal solution of any MaxSAT instance.

UniMaxSAT - Summary

solving ($\mathcal{F}$, *cost*)

- Maintain $\mathcal{AB}$ and $\mathcal{K}$.
- Iterate:
  1. Compute (hopefully) an abstract candidate γ and lower bound *LB* over $\mathcal{K} \wedge \text{DEF}(\mathcal{AB})$.
  2. Check if γ can be extended to a solution τ of $\mathcal{F} \wedge \text{DEF}(\mathcal{AB})$.
  3. If so and $\textit{cost}(\tau) = \textit{LB}$, return found solution.
  4. Else obtain a new (abstract) core
  5. (Optionally) extend $\mathcal{AB}$ with (feasible) abstraction sets.

Theorem:

Assume: (i) γ is an abstract candidate "sufficiently" often,
(ii) *LB* is a lower bound $\text{OPT}(\mathcal{F})$ and
(iii) if γ is abstract candidate, *LB* is the cost of its extensions.
UniMaxSAT returns an optimal solution of any MaxSAT instance.

Interesting Theorems & Open Questions

Differences Between Core Guided Algorithms

Assume an unweighted instance; then MSU3, PMRES, only compute abstract candidates.

Strength of Cores Extracted by Core-Guided Solvers

Assume a core-guided solver has computed an abstract candidate γ and extracted a core κ falsified, then κ is falsified by all abstract candidates.

Open Questions

1. Can we proof-log UniMaxSAT directly?
2. How do we model symmetry breaking with abstraction sets?
3. Does the framework fit e.g. B&B solvers?

Interesting Theorems & Open Questions

Differences Between Core Guided Algorithms

Assume an unweighted instance; then MSU3, PMRES, only compute abstract candidates.

Strength of Cores Extracted by Core-Guided Solvers

Assume a core-guided solver has computed an abstract candidate γ and extracted a core κ falsified, then κ is falsified by all abstract candidates.

Open Questions

1. Can we proof-log UniMaxSAT directly?
2. How do we model symmetry breaking with abstraction sets?
3. Does the framework fit e.g. B&B solvers?

Interesting Theorems & Open Questions

Differences Between Core Guided Algorithms

Assume an unweighted instance; then MSU3, PMRES, only compute abstract candidates.

Strength of Cores Extracted by Core-Guided Solvers

Assume a core-guided solver has computed an abstract candidate γ and extracted a core κ falsified, then κ is falsified by all abstract candidates.

Open Questions

1. Can we proof-log UniMaxSAT directly?
2. How do we model symmetry breaking with abstraction sets?
3. Does the framework fit e.g. B&B solvers?

Interesting Theorems & Open Questions

Differences Between Core Guided Algorithms

Assume an unweighted instance; then MSU3, PMRES, only compute abstract candidates.

Strength of Cores Extracted by Core-Guided Solvers

Assume a core-guided solver has computed an abstract candidate γ and extracted a core κ falsified, then κ is falsified by all abstract candidates.

Open Questions

1. Can we proof-log UniMaxSAT directly?
2. How do we model symmetry breaking with abstraction sets?
3. Does the framework fit e.g. B&B solvers?

Summary of Contributions

- Formalization of the correctness of UniMaxSAT subject to generic properties of subroutines.
- Simulation of essentially all core-guided, IHS, and solution-improving MaxSAT algorithms.
- New insight into the relationships between the algorithms.
- Proof-of-concept new algorithm AbstCG.

Bibliography I

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.

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.

Toby O. Davies, Graeme Gange, and Peter J. Stuckey. Automatic logic-based benders decomposition with minizinc. In Satinder Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 787–793. AAAI Press, 2017. doi: 10.1609/AAAI.V31I1.10654. URL `https://doi.org/10.1609/aaai.v31i1.10654`.

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.

Katalin Fazekas, Fahiem Bacchus, and Armin Biere. Implicit hitting set algorithms for maximum satisfiability modulo theories. In *Proc. IJCAR*, volume 10900 of *Lecture Notes in Computer Science*, pages 134–151. Springer, 2018.

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.

Hannes Ihalainen, Jeremias Berg, and Matti Järvisalo. Unifying core-guided and implicit hitting set based optimization. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pages 1935–1943. ijcai.org, 2023. doi: 10.24963/IJCAI.2023/215. URL `https://doi.org/10.24963/ijcai.2023/215`.

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.

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.

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

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.