Abstract Cores in Implicit Hitting Set MaxSAT solving

Jeremias Berg

University of Helsinki
Finland

June 04, 2021 MIAO seminar / Online

Joint work with Fahiem Bacchus

Many thanks also to Matti Järvisalo and Ruben Martins for contributions to slides

Who am I?

- Post Doctoral researcher at the University of Helsinki
  - Constraint Reasoning and Optimisation Group led by Prof. Matti Järvisalo.
- Defended PhD thesis on algorithms and applications of MaxSAT in 2018
- Visits to Melbourne (Prof. Stuckey) and Toronto (Prof. Bacchus)
- Research focus atm. on declarative methods for solving NP-hard optimisation problems.

Maximum Satisfiability

Maximum Satisfiability—MaxSat

Exact Boolean optimization paradigm

- Builds on the success story of Boolean satisfiability (SAT) solving
- Great recent improvements in practical solver technology
- Expanding range of real-world applications

Offers an alternative to e.g. integer programming

- Solvers provide provably optimal solutions
- Propositional logic as the underlying declarative language: especially suited for inherently "Boolean" optimization problems

Implicit Hitting Set based Maximum Satisfiability

The IHS based approach to MaxSat

One of the central methods for exactly solving instances arising in real-world domains.

- Decouples MaxSat into separate reasoning (i.e. core-extraction) and optimization steps.
- Avoids increasing the complexity of SAT-calls.
- Top positions in annual evaluations since 2015
- IHS framework instantiated in various applications [Karp, 2010; Saikko, Wallner, and Järvisalo, 2016b; Fazekas, Bacchus, and Biere, 2018; Ignatiev, Previti, Liffiton, and Marques-Silva, 2015].

Outline

1. Motivation and Basic Concepts
2. (Short) Overview of MaxSAT solving Algorithms.
3. The implicit hitting set approach to MaxSAT
   - With Correction Sets
   - With Bounds.
4. Abstract Cores

Success of SAT

The Boolean satisfiability (SAT) Problem

Input: A propositional logic formula F.
Task: Is F satisfiable?

Success of SAT

The Boolean satisfiability (SAT) Problem

Input: A propositional logic formula F.
Task: Is F satisfiable?

SAT is a Great Success Story

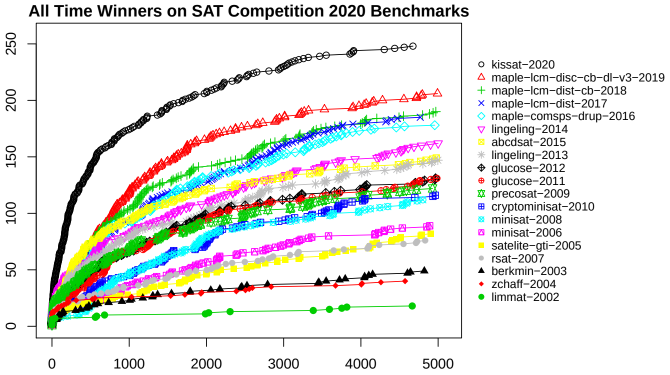
Not merely a central problem in theory:

Remarkable improvements since mid 90s in SAT solvers: practical decision procedures for SAT

- Find solutions if they exist
- Prove non-existence of solutions

SAT Solvers

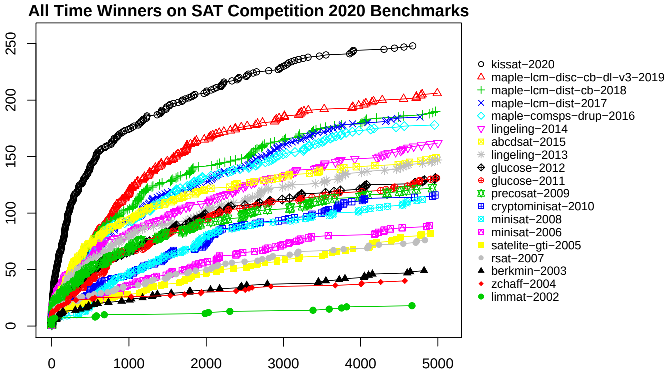
From 100s of variables and constraints (early 90s)
up to 10M variables and constraints. (21st century).



Plot provided by Armin Biere

SAT Solvers

From 100s of variables and constraints (early 90s)
up to 10M variables and constraints. (21st century).



Plot provided by Armin Biere

Core NP search procedures for solving various types of computational problems

Optimization

Most real-world problems involve an optimization component

Examples:

- Find a shortest path/plan/execution/...to a goal state
  - Planning, model checking, ...
- Find a smallest explanation
  - Debugging, configuration, ...
- Find a least resource-consuming schedule
  - Scheduling, logistics, ...
- Find a most probable explanation (MAP)
  - Probabilistic inference, ...

Optimization

Most real-world problems involve an optimization component

Examples:

- Find a shortest path/plan/execution/...to a goal state
  - Planning, model checking, ...
- Find a smallest explanation
  - Debugging, configuration, ...
- Find a least resource-consuming schedule
  - Scheduling, logistics, ...
- Find a most probable explanation (MAP)
  - Probabilistic inference, ...

High demand for automated approaches to
finding good solutions to computationally hard
optimization problems

$\leadsto$ Maximum satisfiability

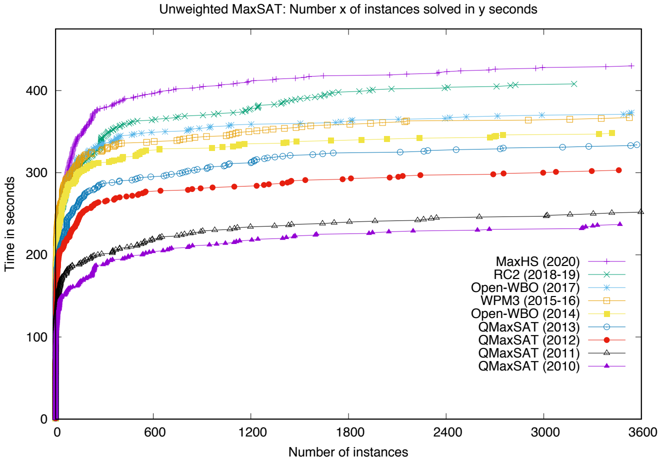
MaxSat Applications

Drastically increasing number of successful applications

- Planning, Scheduling, and Configuration
- Data Analysis and Machine Learning
- Knowledge Representation and Reasoning
- Combinatorial Optimization
- Verification and Security
- Bioinformatics
- ...
  - Tens of new problem domains in MaxSAT Evaluations

This progress is much due to significant progress in efficient MaxSAT solvers.

Progress in MaxSat Solver Performance



Comparing some of the best solvers from 2010–2020:

In 2020: 81% more instances solved than in 2010!

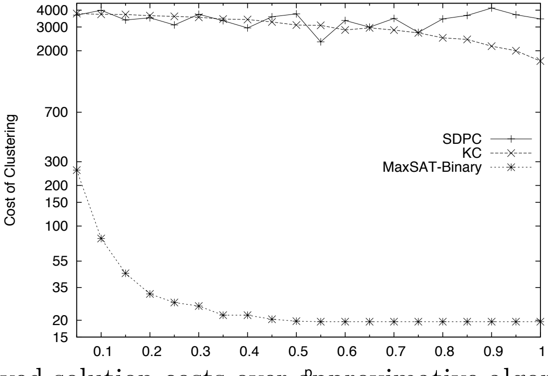
- On same computer, same set of benchmarks: 576 unweighted MaxSat Evaluation 2020 instances

Benefits of (IHS-based) MaxSat

Provably optimal solutions

Example: Correlation clustering by (IHS-based) MaxSat

[Berg and Järvisalo, 2017]



- Improved solution costs over approximative algorithms
- Good performance even on sparse data (missing values)

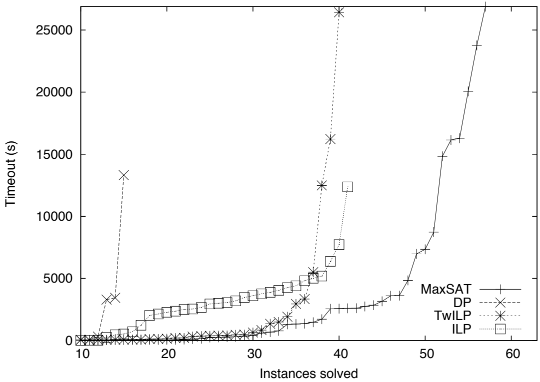
Benefits of (IHS-based) MaxSat

Surpassing the efficiency of specialized algorithms

Example:
Learning optimal bounded-treewidth Bayesian networks

[Berg, Järvisalo, and Malone, 2014]

MaxSat vs Dynamic Programming and MIP



Basic Concepts

MaxSat: Basic Definitions

- Simple constraint language:
  conjunctive normal form (CNF) propositional formulas
  - More high-level constraints encoded as sets of clauses
- Literal: a boolean variable x or ¬x.
- Clause C: a disjunction (∨) of literals. e.g $(x \vee y \vee \neg z)$
- Truth assignment τ: a function from Boolean variables to $\{0, 1\}$.
- Satisfaction:
  $\tau(C) = 1$ if
  $\tau(x) = 1$ for some literal $x \in C$, or
  $\tau(x) = 0$ for some literal $\neg x \in C$.

  At least one literal of C is made true by τ.

MaxSat: Basic Definitions

MaxSat

INPUT: a set of clauses F. (a CNF formula)

TASK: find τ s.t. $\sum_{C \in F} \tau(C)$ is maximized.

Find truth assignment that satisfies a maximum number of clauses

This is the standard definition, much studied in Theoretical Computer Science.

- Often inconvenient for modelling practical problems.

Central Generalizations of MaxSat

Weighted MaxSat

- Each clause C has an associated weight w_C
- Optimal solutions maximize the sum of weights of satisfied clauses: τ s.t. $\displaystyle\sum_{C \in F} w_c \tau(C)$ is maximized.

Partial MaxSat

- Two sets of clauses $\mathcal{F}_H$ and $\mathcal{F}_S$
- Clauses in $\mathcal{F}_H$ deemed hard
  - Any solution has to satisfy the hard clauses
- Clauses in $\mathcal{F}_S$ are soft.

Weighted Partial MaxSat

Hard clauses (partial) + weights on soft clauses (weighted)

Central Generalizations of MaxSat

Partial MaxSat

- Two sets of clauses $\mathcal{F}_{\mathrm{H}}$ and $\mathcal{F}_{\mathrm{S}}$
- Clauses in $\mathcal{F}_{\mathrm{H}}$ deemed hard
- Clauses in $\mathcal{F}_{\mathrm{S}}$ are soft.
- Find model τ that satisfies $\mathcal{F}_{\mathrm{H}}$ and maximizes

$$\sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} \tau(\mathrm{C})$$

Rest of the talk

unweighted examples

All techniques applicable in the weighted case as well.

MaxSat Algorithmically

In theory - a maximization problem

Find model τ that satisfies $\mathcal{F}_{\mathrm{H}}$ and maximizes

$$\sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} \tau(\mathrm{C})$$

MaxSat Algorithmically

In theory - a maximization problem

Find model τ that satisfies $\mathcal{F}_\mathrm{H}$ and maximizes

$$\sum_{\mathrm{C} \in \mathcal{F}_\mathrm{S}} \tau(\mathrm{C})$$

In practice - a minimization problem

Find model τ that satisfies $\mathcal{F}_\mathrm{H}$ and minimizes

$$\mathrm{cost}(\tau) = \sum_{\mathrm{C} \in \mathcal{F}_\mathrm{S}} (1 - \tau(\mathrm{C}))$$

Example

INPUT: Instance $\mathcal{I} = (\mathcal{F}_{\mathrm{H}}, \mathcal{F}_{\mathrm{S}})$
OUTPUT: Solution τ that:
(i) satisfies $\mathcal{F}_{\mathrm{H}}$
(ii) minimizes $\mathrm{cost}(\tau) = \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} 1 - \tau(\mathrm{C})$

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1 \vee \mathrm{b}_2), (\mathrm{b}_2 \vee \mathrm{b}_3)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$$

Example

INPUT: Instance $\mathcal{I} = (\mathcal{F}_{\mathrm{H}}, \mathcal{F}_{\mathrm{S}})$
OUTPUT: Solution τ that:
(i) satisfies $\mathcal{F}_{\mathrm{H}}$
(ii) minimizes $\mathrm{cost}(\tau) = \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} 1 - \tau(\mathrm{C})$

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1 \vee \mathrm{b}_2), (\mathrm{b}_2 \vee \mathrm{b}_3)\}$$

$$\boxed{\tau(\mathrm{b}_1) = \tau(\mathrm{b}_3) = \tau(\mathrm{b}_2) = 1}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$$

$$\mathrm{cost}(\tau) = 3$$

Example

INPUT: Instance $\mathcal{I} = (\mathcal{F}_{\mathrm{H}}, \mathcal{F}_{\mathrm{S}})$
OUTPUT: Solution τ that:
(i) satisfies $\mathcal{F}_{\mathrm{H}}$
(ii) minimizes $\mathrm{cost}(\tau) = \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} 1 - \tau(\mathrm{C})$

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1 \vee \mathrm{b}_2), (\mathrm{b}_2 \vee \mathrm{b}_3)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$$

$$\tau(\mathrm{b}_1) = \tau(\mathrm{b}_3) = 0$$
$$\tau(\mathrm{b}_2) = 1$$

$$\mathrm{cost}(\tau) = 1$$

Example

INPUT: Instance $\mathcal{I} = (\mathcal{F}_{\mathrm{H}}, \mathcal{F}_{\mathrm{S}})$
OUTPUT: Solution τ that:
(i) satisfies $\mathcal{F}_{\mathrm{H}}$
(ii) minimizes $\mathrm{cost}(\tau) = \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} 1 - \tau(\mathrm{C})$

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1 \vee \mathrm{b}_2), (\mathrm{b}_2 \vee \mathrm{b}_3)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$$

$$\boxed{\tau = \{\neg \mathrm{b}_1, \mathrm{b}_2, \neg \mathrm{b}_3\}}$$

Assignments treated as sets of literals.

MaxSat: Complexity

Deciding whether k clauses can be satisfied: NP-complete

Input: A CNF formula F, a positive integer k.
Question:
Is there an assignment that satisfies at least k clauses in F?

MaxSat: Complexity

Deciding whether k clauses can be satisfied: NP-complete

Input: A CNF formula F, a positive integer k.
Question:
Is there an assignment that satisfies at least k clauses in F?

MaxSat is $\mathrm{FP}^{\mathrm{NP}}$–complete

- Polynomial number of oracle calls
- A SAT solver acts as the NP oracle most often in practice

MaxSat: Complexity

Deciding whether k clauses can be satisfied: NP-complete

Input: A CNF formula F, a positive integer k.
Question:
Is there an assignment that satisfies at least k clauses in F?

MaxSat is FP^{NP}–complete

- Polynomial number of oracle calls
- A SAT solver acts as the NP oracle most often in practice

Complexity of IHS for solving MaxSat

Theorem

For every $n \in \mathbb{N}$ there exists an instance $\mathcal{I}$ on which an IHS algorithm needs to perform $\Omega(2^n)$ SAT-solver calls.

A (short) overview of MaxSat solvers

Types of MaxSat Solvers

MaxSat Solver

Practical implementation of an algorithm for finding (optimal) solutions to MaxSAT instances

Focus here: Complete MaxSat solving

- Guaranteed to output a provably optimal solution to any instance
  (given enough resources (time & space))

Push-Button Solvers

- Black-box, no command line parameters necessary
- Input: CNF formula, in the standard DIMACS WCNF file format
- Output: provably optimal solution, or UNSATISFIABLE
  - Complete solvers

```
mancoosi-test-i2000d0u98-26.wcnf
p wcnf 18169 112632 31540812410
31540812410 -1 2 3 0
31540812410 -4 2 3 0
31540812410 -5 6 0
...
18170 1133 0
18170 457 0
... truncated 2.4 MB
```

Internally rely especially on CDCL SAT solvers

for proving unsatisfiability of subsets of clauses

Availability

Open Source

Starting from 2017, solvers need to be open-source in order to participate in MaxSat Evaluations

- Incentive for openness
- Allow other to build on and test new ideas on establish solver source bases

https://maxsat-evaluations.github.io/

Types of Complete Solvers

- Branch and Bound
  - Can be effective of small-but hard & randomly generated instances

Types of Complete Solvers

- Branch and Bound
- SAT-based MaxSat algorithms

Types of Complete Solvers

- Branch and Bound
- SAT-based MaxSat algorithms
  - Model-improving

Upper Bounding

use a SAT-solver to extract solutions of increasing quality until no better ones can be found

Types of Complete Solvers

- Branch and Bound
- SAT-based MaxSat algorithms
  - Model-improving
  - Core-guided

Lower Bounding

use a SAT solver to extract small sets of unsatisfiable constraints and relax the instance in a controlled way.

Types of Complete Solvers

- Branch and Bound
- SAT-based MaxSat algorithms
  - Model-improving
  - Core-guided
  - Implicit hitting set

Hybrid

decouple MaxSAT solving into core-extraction and optimisation

Implicit Hitting Set Algorithms for MaxSat

[Davies and Bacchus, 2011, 2013b,a]

Goals for this Section

- Basic concepts:
  - Cores
  - Hitting Sets
- Implicit Hitting set for solving MaxSAT (the simple way)

Unsat Cores

- Central in IHS MaxSAT:

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1 \lor \mathrm{b}_2), (\mathrm{b}_2 \lor \mathrm{b}_3)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$$

Unsat Cores

- Central in IHS MaxSAT:
- $\kappa \subset \mathcal{F}_{\mathrm{S}}$ is an core if $\mathcal{F}_{\mathrm{H}} \wedge \kappa$ is unsatisfiable

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1 \vee \mathrm{b}_2), (\mathrm{b}_2 \vee \mathrm{b}_3)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$$

$$\kappa = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}$$

Unsat Cores

- Central in IHS MaxSAT:
- $\kappa \subset \mathcal{F}_{\mathrm{S}}$ is an core if $\mathcal{F}_{\mathrm{H}} \wedge \kappa$ is unsatisfiable
- $\kappa \subset \mathcal{F}_{\mathrm{S}}$ is an MUS if no $\kappa_{\mathrm{s}} \subsetneq \kappa$ is a core.

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1 \vee \mathrm{b}_2), (\mathrm{b}_2 \vee \mathrm{b}_3)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$$

$$\kappa = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}$$

Unsat Cores

- Central in IHS MaxSAT:
- $\kappa \subset \mathcal{F}_\mathrm{S}$ is an core if $\mathcal{F}_\mathrm{H} \wedge \kappa$ is unsatisfiable
- $\kappa \subset \mathcal{F}_\mathrm{S}$ is an MUS if no $\kappa_\mathrm{s} \subsetneq \kappa$ is a core.

In the rest of the presentation, we represent clauses $(\mathrm{b}_1 \vee \mathrm{b}_2)$ as $(\mathrm{b}_1, \mathrm{b}_2)$.

$\mathcal{F}_\mathrm{H} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3)\}$

$\mathcal{F}_\mathrm{S} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$

$\kappa = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}$

Hitting Sets over Cores

- C - a collection of cores

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$$

$$\begin{aligned} \mathrm{C} = \{&\{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}, \\ &\{(\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}\} \end{aligned}$$

Hitting Sets over Cores

- C - a collection of cores
- $\mathrm{hs} \subset \mathcal{F}_\mathrm{S}$ is an hitting set if $\mathrm{hs} \cap \kappa \neq \emptyset$ for all $\kappa \in \mathrm{C}$

$$\mathcal{F}_\mathrm{H} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3)\}$$

$$\mathcal{F}_\mathrm{S} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$$

$$\mathrm{C} = \{\{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}, \{(\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}\}$$

$$\mathrm{hs}_1 = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_3)\}$$
$$\mathrm{cost}(\mathrm{hs}_1) = 2$$

Hitting Sets over Cores

- C - a collection of cores
- $\text{hs} \subset \mathcal{F}_\text{S}$ is an hitting set if $\text{hs} \cap \kappa \neq \emptyset$ for all $\kappa \in \text{C}$
- $\text{cost}(\text{hs}) = |\text{hs}|$ (i.e. number of clauses in it)

$$\mathcal{F}_\text{H} = \{(\text{b}_1, \text{b}_2), (\text{b}_2, \text{b}_3)\}$$

$$\mathcal{F}_\text{S} = \{(\neg\text{b}_1), (\neg\text{b}_2), (\neg\text{b}_3)\}$$

$$\text{C} = \{\{(\neg\text{b}_1), (\neg\text{b}_2)\}, \{(\neg\text{b}_2), (\neg\text{b}_3)\}\}$$

$\text{hs}_1 = \{(\neg\text{b}_1), (\neg\text{b}_3)\}$
$\text{cost}(\text{hs}_1) = 2$

$\text{hs}_2 = \{(\neg\text{b}_2)\}$
$\text{cost}(\text{hs}_2) = 1$

Hitting Sets over Cores

- C - a collection of cores
- $\mathrm{hs} \subset \mathcal{F}_{\mathrm{S}}$ is an hitting set if $\mathrm{hs} \cap \kappa \neq \emptyset$ for all $\kappa \in \mathrm{C}$
- $\mathrm{cost}(\mathrm{hs}) = |\mathrm{hs}|$ (i.e. number of clauses in it)
- hs is minimum-cost if no other hs′ has $\mathrm{cost}(\mathrm{hs}') < \mathrm{cost}(\mathrm{hs})$

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$$

$$\mathrm{C} = \{\{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}, \{(\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}\}$$

$$\mathrm{hs}_1 = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_3)\}$$
$$\mathrm{cost}(\mathrm{hs}_1) = 2$$

$$\mathrm{hs}_2 = \{(\neg \mathrm{b}_2)\}$$
$$\mathrm{cost}(\mathrm{hs}_2) = 1$$

What does this have to do with MaxSat?

- C all (subset minimal) cores.
- hs, minimum-cost over C

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3)\}$

$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$

$\mathrm{C} = \{\{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}, \{(\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}\}$

$\mathrm{hs} = \{(\neg \mathrm{b}_2)\} \quad \mathrm{cost}(\mathrm{hs}_2) = 1$

What does this have to do with MaxSat?

- C all (subset minimal) cores.
- hs, minimum-cost over C
- $\rightarrow$ exists τ^{hs} that satisfies exactly $\mathcal{F}_{\text{H}} \wedge (\mathcal{F}_{\text{S}} \setminus \text{hs})$.

$$\mathcal{F}_{\text{H}} = \{(\text{b}_1, \text{b}_2), (\text{b}_2, \text{b}_3)\}$$

$$\mathcal{F}_{\text{S}} = \{(\neg\text{b}_1), (\neg\text{b}_2), (\neg\text{b}_3)\}$$

$$\text{C} = \{\{(\neg\text{b}_1), (\neg\text{b}_2)\}, \{(\neg\text{b}_2), (\neg\text{b}_3)\}\}$$

$$\text{hs} = \{(\neg\text{b}_2)\} \quad \text{cost}(\text{hs}_2) = 1$$

$$\tau^{\text{hs}} = \{\neg\text{b}_1, \text{b}_2, \neg\text{b}_3\} \quad \text{cost}(\tau^{\text{hs}}) = 1$$

What does this have to do with MaxSat?

- C all (subset minimal) cores.
- hs, minimum-cost over C
- $\rightarrow$ exists τ^{hs} that satisfies exactly $\mathcal{F}_{\text{H}} \wedge (\mathcal{F}_{\text{S}} \setminus \text{hs})$.

Key insight

- Such hs can be computed implicitly.

What does this have to do with MaxSat?

- C all (subset minimal) cores.
- hs, minimum-cost over C
- $\rightarrow$ exists τ^{hs} that satisfies exactly $\mathcal{F}_{\text{H}} \wedge (\mathcal{F}_{\text{S}} \setminus \text{hs})$.

Key insight

- Such hs can be computed implicitly.
  - Compute a minimum-cost hs over any set of cores

What does this have to do with MaxSat?

- C all (subset minimal) cores.
- hs, minimum-cost over C
- $\rightarrow$ exists τ^{hs} that satisfies exactly $\mathcal{F}_{\text{H}} \wedge (\mathcal{F}_{\text{S}} \setminus \text{hs})$.

Key insight

- Such hs can be computed implicitly.
  - Compute a minimum-cost hs over any set of cores
  - Check if $\mathcal{F}_{\text{H}} \wedge (\mathcal{F}_{\text{S}} \setminus \text{hs})$ is satisfiable.

Implicit Hitting Set Approach to MaxSat

Iterate over the following steps:

- Accumulate a collection $\mathcal{K}$ of UNSAT cores

  using a SAT solver
- Find an optimal hitting set hs over $\mathcal{K}$,
  and rule out the clauses in hs for the next SAT solver call

  using an IP solver

...until the SAT solver returns satisfying assignment.

Implicit Hitting Set Approach to MaxSat

Iterate over the following steps:

- Accumulate a collection $\mathcal{K}$ of UNSAT cores

  using a SAT solver
- Find an optimal hitting set hs over $\mathcal{K}$,
  and rule out the clauses in hs for the next SAT solver call

  using an IP solver

...until the SAT solver returns satisfying assignment.

Hitting Set Problem as Integer Programming

$$\mathsf{min} \sum_{\mathrm{C} \in \cup \mathcal{K}} \mathrm{w}(\mathrm{C}) \cdot \mathrm{b}_{\mathrm{C}}$$

$$\text{subject to} \quad \sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_{\mathrm{C}} \geq 1 \quad \forall \mathrm{K} \in \mathcal{K}$$

- $\mathrm{b}_{\mathrm{C}} = 1$ iff clause C in the hitting set
- Weight function w: works also for weighted MaxSat

Implicit Hitting Set Approach to MaxSat

"Best out of both worlds"

Combining the main strengths of SAT and IP solvers:

Implicit Hitting Set Approach to MaxSat

"Best out of both worlds"

Combining the main strengths of SAT and IP solvers:

- SAT solvers are very good at proving unsatisfiability
  - Explanations for unsatisfiability in terms of cores
  - Each SAT solver call made on a subset of the clauses in the instance

Implicit Hitting Set Approach to MaxSat

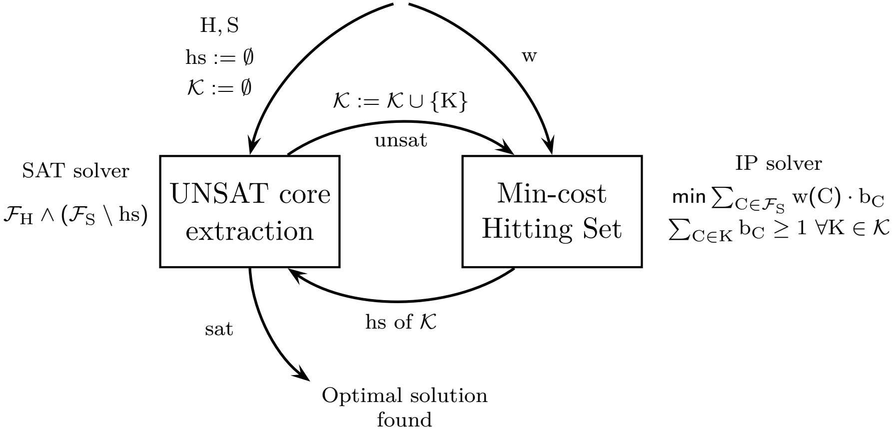
"Best out of both worlds"

Combining the main strengths of SAT and IP solvers:

- SAT solvers are very good at proving unsatisfiability
  - Explanations for unsatisfiability in terms of cores
  - Each SAT solver call made on a subset of the clauses in the instance
- IP solvers at optimization
  - Instead of directly solving the input MaxSAT instance: solve a sequence of simpler hitting set problems over the cores

Solving MaxSat by SAT and Hitting Set Computations

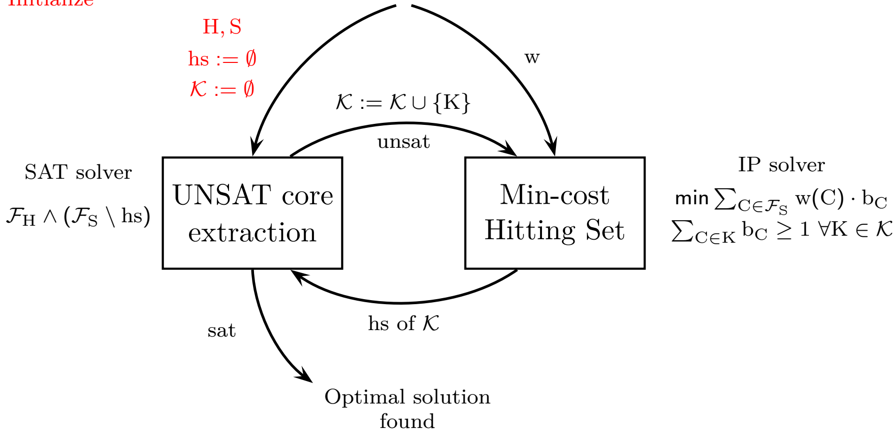
Input:
hard clauses $\mathcal{F}_H$, soft clauses $\mathcal{F}_S$, weight function $w : S \mapsto \mathbb{R}^+$



Solving MaxSat by SAT and Hitting Set Computations

Input:
hard clauses $\mathcal{F}_H$, soft clauses $\mathcal{F}_S$, weight function $w : S \mapsto \mathbb{R}^+$

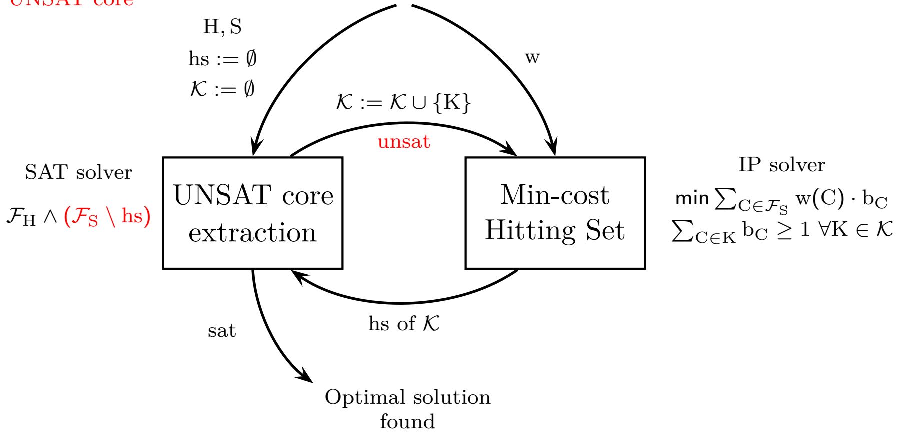
1. Initialize



Solving MaxSat by SAT and Hitting Set Computations

Input:
hard clauses $\mathcal{F}_H$, soft clauses $\mathcal{F}_S$, weight function $w : S \mapsto \mathbb{R}^+$

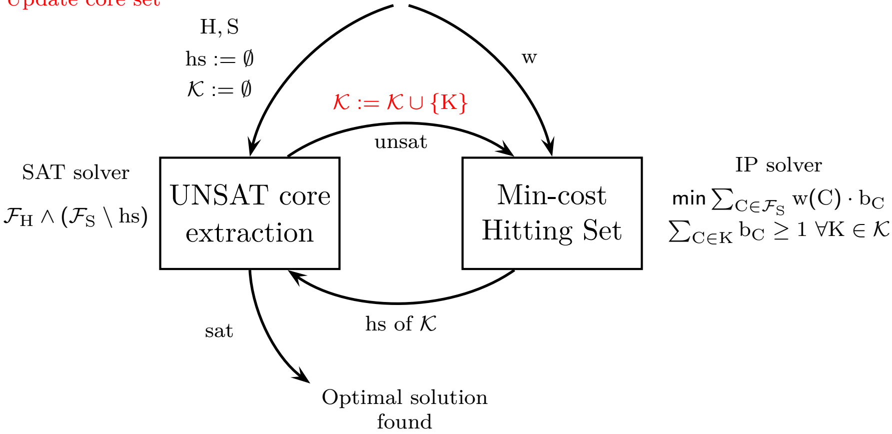
2. UNSAT core



Solving MaxSat by SAT and Hitting Set Computations

Input:
hard clauses $\mathcal{F}_H$, soft clauses $\mathcal{F}_S$, weight function $w : S \mapsto \mathbb{R}^+$

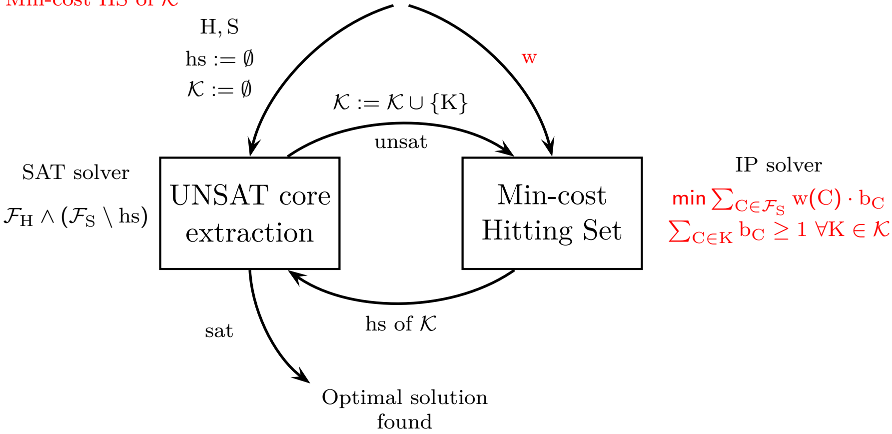
3. Update core set



Solving MaxSat by SAT and Hitting Set Computations

Input:
hard clauses $\mathcal{F}_H$, soft clauses $\mathcal{F}_S$, weight function $w : S \mapsto \mathbb{R}^+$

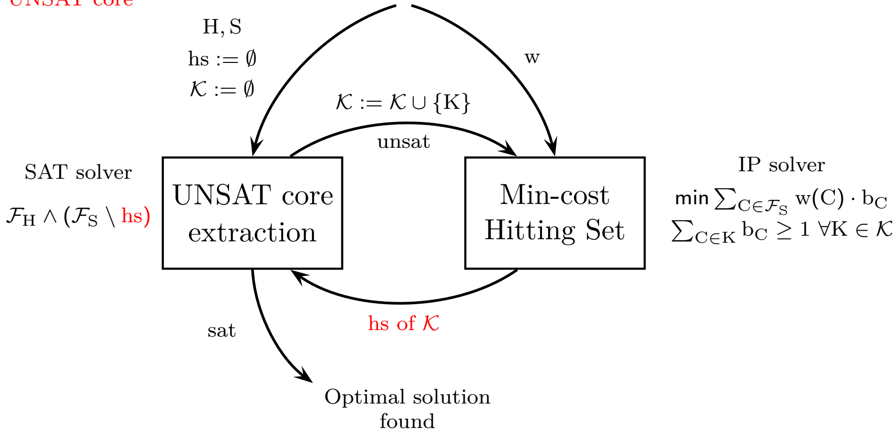
4. Min-cost HS of $\mathcal{K}$



Solving MaxSat by SAT and Hitting Set Computations

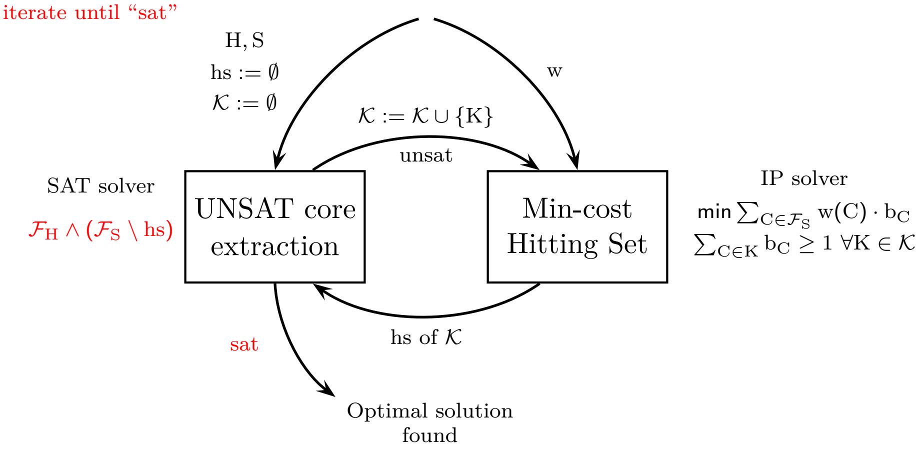
Input:
hard clauses $\mathcal{F}_H$, soft clauses $\mathcal{F}_S$, weight function $w : S \mapsto \mathbb{R}^+$

5. UNSAT core



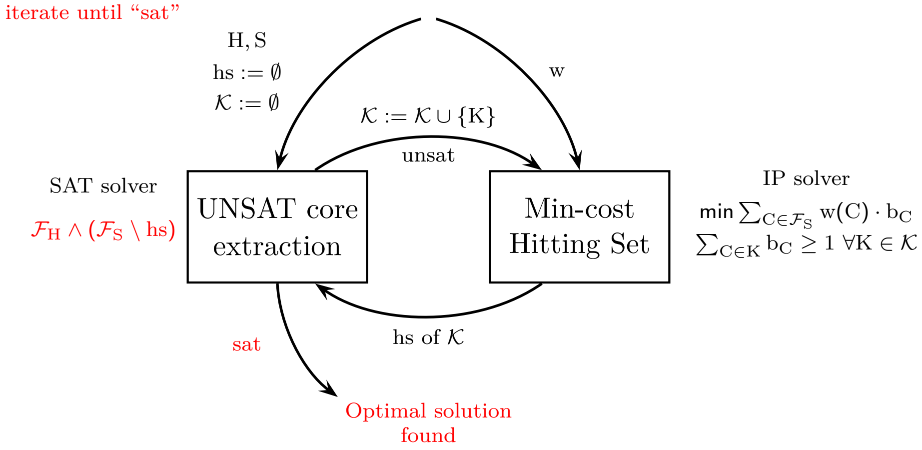
Solving MaxSat by SAT and Hitting Set Computations

Input:
hard clauses $\mathcal{F}_H$, soft clauses $\mathcal{F}_S$, weight function $w : S \mapsto \mathbb{R}^+$



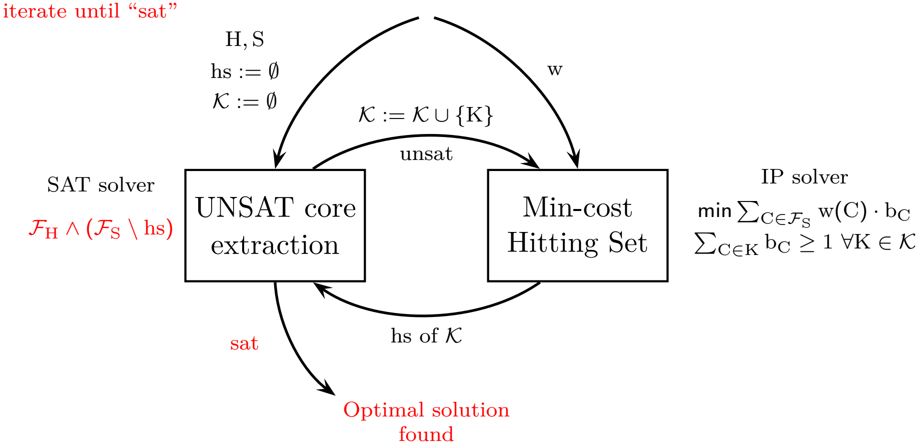
Solving MaxSat by SAT and Hitting Set Computations

Input:
hard clauses $\mathcal{F}_H$, soft clauses $\mathcal{F}_S$, weight function $w : S \mapsto \mathbb{R}^+$



Solving MaxSat by SAT and Hitting Set Computations

Intuition: After optimally hitting all cores of $\mathcal{F}_H \wedge \mathcal{F}_S$ by hs: any solution to $\mathcal{F}_H \wedge (\mathcal{F}_S \setminus \text{hs})$ is guaranteed to be optimal.



Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{S}} = \{(\neg\mathrm{b}_1), (\neg\mathrm{b}_2), (\neg\mathrm{b}_3), (\neg\mathrm{b}_4)\}$

Basic-IHS ($\mathcal{F}$)

SAT solver
$\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$

IP solver
$\mathsf{min} \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} \mathrm{b}_{\mathrm{C}}$
$\sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_{\mathrm{C}} \geq 1 \ \forall \mathrm{K} \in \mathrm{C}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$

Basic-IHS ($\mathcal{F}$)

Initialize

SAT solver
$\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$

IP solver
$\mathsf{min} \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} \mathrm{b}_{\mathrm{C}}$
$\sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_{\mathrm{C}} \geq 1 \ \forall \mathrm{K} \in \mathrm{C}$

$\mathrm{C} = \emptyset$

Solving (unweighted) MaxSat with IHS

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$$

Basic-IHS ($\mathcal{F}$)

Initialize
while True

SAT solver
$\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$

IP solver
$\min \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} \mathrm{b}_{\mathrm{C}}$
$\sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_{\mathrm{C}} \geq 1 \ \forall \mathrm{K} \in \mathcal{C}$

$\mathcal{C} = \emptyset$

Solving (unweighted) MaxSat with IHS

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$$

Basic-IHS $(\mathcal{F})$

Initialize
while True
  Compute hs

SAT solver
$\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$

IP solver
$\mathsf{min} \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} \mathrm{b}_{\mathrm{C}}$
$\sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_{\mathrm{C}} \geq 1 \ \forall \mathrm{K} \in \mathrm{C}$

ip-solve
$\mathrm{hs} = \emptyset$

$\mathrm{C} = \emptyset$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_\mathrm{H} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_\mathrm{S} = \{(\neg\mathrm{b}_1), (\neg\mathrm{b}_2), (\neg\mathrm{b}_3), (\neg\mathrm{b}_4)\}$

Basic-IHS ($\mathcal{F}$)

Initialize
while True
    Compute hs
    SAT-solve $\mathcal{F}_\mathrm{H} \wedge (\mathcal{F}_\mathrm{S} \setminus \mathrm{hs})$

SAT solver
$\mathcal{F}_\mathrm{H} \wedge (\mathcal{F}_\mathrm{S} \setminus \mathrm{hs})$

IP solver
$\min \sum_{\mathrm{C} \in \mathcal{F}_\mathrm{S}} \mathrm{b}_\mathrm{C}$
$\sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_\mathrm{C} \geq 1 \ \forall \mathrm{K} \in \mathrm{C}$

sat-solve
$\mathcal{F}_\mathrm{H} \wedge \{(\neg\mathrm{b}_1), (\neg\mathrm{b}_2), (\neg\mathrm{b}_3), (\neg\mathrm{b}_4)\}$ $\quad$ $\mathrm{hs} = \emptyset$

$\mathrm{C} = \emptyset$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_H = \{(b_1, b_2), (b_2, b_3), (b_3, b_4)\}$

$\mathcal{F}_S = \{(\neg b_1), (\neg b_2), (\neg b_3), (\neg b_4)\}$

Basic-IHS ($\mathcal{F}$)

Initialize
while True
  Compute hs
  SAT-solve $\mathcal{F}_H \wedge (\mathcal{F}_S \setminus hs)$
  If UNSAT
    add core to C

SAT solver
$\mathcal{F}_H \wedge (\mathcal{F}_S \setminus hs)$

IP solver
$\min \sum_{C \in \mathcal{F}_S} b_C$
$\sum_{C \in K} b_C \geq 1 \;\forall K \in C$

sat-solve
$\mathcal{F}_H \wedge \{(\neg b_1), (\neg b_2), (\neg b_3), (\neg b_4)\}$

Result UNSAT $\kappa = \{(\neg b_1), (\neg b_2)\}$

$C = \{\{(\neg b_1), (\neg b_2)\}\}$

Solving (unweighted) MaxSat with IHS

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$$

Basic-IHS $(\mathcal{F})$

Initialize
while True
  Compute hs
  SAT-solve $\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$
  If UNSAT
    add core to C

SAT solver
$\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$

IP solver
$\min \sum_{C \in \mathcal{F}_{\mathrm{S}}} \mathrm{b}_C$
$\sum_{C \in K} \mathrm{b}_C \geq 1 \ \forall K \in \mathrm{C}$

ip-solve
$\mathrm{hs} = \{(\neg \mathrm{b}_1)\}$

$$\mathrm{C} = \{\{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}\}$$

Solving (unweighted) MaxSat with IHS

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$$

Basic-IHS ($\mathcal{F}$)

Initialize
while True
  Compute hs
  SAT-solve $\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$
  If UNSAT
    add core to C

SAT solver
$\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$

IP solver
$\min \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} \mathrm{b}_{\mathrm{C}}$
$\sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_{\mathrm{C}} \geq 1 \; \forall \mathrm{K} \in \mathrm{C}$

sat-solve
$\mathcal{F}_{\mathrm{H}} \wedge \{(\neg \mathrm{b}_2), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$

$\mathrm{hs} = \{(\neg \mathrm{b}_1)\}$

$\mathrm{C} = \{\{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}\}$

Solving (unweighted) MaxSat with IHS

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$$

SAT solver
$\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$

IP solver
$\mathsf{min} \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} \mathrm{b}_{\mathrm{C}}$
$\sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_{\mathrm{C}} \geq 1 \ \forall \mathrm{K} \in \mathrm{C}$

sat-solve
$\mathcal{F}_{\mathrm{H}} \wedge \{(\neg \mathrm{b}_2), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$

Result UNSAT $\quad \kappa = \{(\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$

$\mathrm{C} = \{\{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}, \{(\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}\}$

Basic-IHS ($\mathcal{F}$)

Initialize
while True
  Compute hs
  SAT-solve $\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$
  If UNSAT
    add core to C

Solving (unweighted) MaxSat with IHS

$$\mathcal{F}_\mathrm{H} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\mathcal{F}_\mathrm{S} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$$

Basic-IHS ($\mathcal{F}$)

Initialize
while True
  Compute hs
  SAT-solve $\mathcal{F}_\mathrm{H} \wedge (\mathcal{F}_\mathrm{S} \setminus \mathrm{hs})$
  If UNSAT
    add core to C

SAT solver
$\mathcal{F}_\mathrm{H} \wedge (\mathcal{F}_\mathrm{S} \setminus \mathrm{hs})$

IP solver
$\min \sum_{\mathrm{C} \in \mathcal{F}_\mathrm{S}} \mathrm{b}_\mathrm{C}$
$\sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_\mathrm{C} \geq 1 \ \forall \mathrm{K} \in \mathrm{C}$

ip-solve
$\mathrm{hs} = \{(\neg \mathrm{b}_2)\}$

$$\mathrm{C} = \{\{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}, \{(\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}\}$$

Solving (unweighted) MaxSat with IHS

$$\mathcal{F}_\mathrm{H} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\mathcal{F}_\mathrm{S} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$$

Basic-IHS ($\mathcal{F}$)

Initialize
while True
  Compute hs
  SAT-solve $\mathcal{F}_\mathrm{H} \wedge (\mathcal{F}_\mathrm{S} \setminus \mathrm{hs})$
  If UNSAT
    add core to C

SAT solver
$\mathcal{F}_\mathrm{H} \wedge (\mathcal{F}_\mathrm{S} \setminus \mathrm{hs})$

IP solver
$\min \sum_{\mathrm{C} \in \mathcal{F}_\mathrm{S}} \mathrm{b}_\mathrm{C}$
$\sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_\mathrm{C} \geq 1 \ \forall \mathrm{K} \in \mathrm{C}$

sat-solve
$\mathcal{F}_\mathrm{H} \wedge \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$

$\mathrm{hs} = \{(\neg \mathrm{b}_2)\}$

$$\mathrm{C} = \{\{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}, \{(\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}\}$$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$

Basic-IHS ($\mathcal{F}$)

Initialize
while True
  Compute hs
  SAT-solve $\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$
  If UNSAT
    add core to C

SAT solver
$\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$

IP solver
$\mathsf{min} \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} \mathrm{b}_{\mathrm{C}}$
$\sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_{\mathrm{C}} \geq 1 \ \forall \mathrm{K} \in \mathrm{C}$

sat-solve
$\mathcal{F}_{\mathrm{H}} \wedge \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$

Result UNSAT $\quad \kappa = \{(\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$

$\mathrm{C} = \{\{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}, \{(\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\},$
$(\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}\}$

Solving (unweighted) MaxSat with IHS

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$$

Basic-IHS ($\mathcal{F}$)

Initialize
while True
  Compute hs
  SAT-solve $\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$
  If UNSAT
    add core to C

SAT solver
$\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$

IP solver
$\min \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} \mathrm{b}_{\mathrm{C}}$
$\sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_{\mathrm{C}} \geq 1 \ \forall \mathrm{K} \in \mathrm{C}$

ip-solve
$\mathrm{hs} = \{(\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\}$

$\mathrm{C} = \{\{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}, \{(\neg \mathrm{b}_2), (\neg \mathrm{b}_3)\},$
$(\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{S}} = \{(\neg\mathrm{b}_1), (\neg\mathrm{b}_2), (\neg\mathrm{b}_3), (\neg\mathrm{b}_4)\}$

Basic-IHS ($\mathcal{F}$)

Initialize
while True
  Compute hs
  SAT-solve $\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$
  If UNSAT
    add core to C

SAT solver
$\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$

IP solver
$\min \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} \mathrm{b}_{\mathrm{C}}$
$\sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_{\mathrm{C}} \geq 1 \ \forall \mathrm{K} \in \mathrm{C}$

sat-solve
$\mathcal{F}_{\mathrm{H}} \wedge \{(\neg\mathrm{b}_1), (\neg\mathrm{b}_4)\}$

$\mathrm{hs} = \{(\neg\mathrm{b}_2), (\neg\mathrm{b}_3)\}$

$\mathrm{C} = \{\{(\neg\mathrm{b}_1), (\neg\mathrm{b}_2)\}, \{(\neg\mathrm{b}_2), (\neg\mathrm{b}_3)\},$
$(\neg\mathrm{b}_3), (\neg\mathrm{b}_4)\}\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{S}} = \{(\neg\mathrm{b}_1), (\neg\mathrm{b}_2), (\neg\mathrm{b}_3), (\neg\mathrm{b}_4)\}$

SAT solver
$\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$

IP solver
$\min \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} \mathrm{b}_{\mathrm{C}}$
$\sum_{\mathrm{C} \in \mathrm{K}} \mathrm{b}_{\mathrm{C}} \geq 1 \ \forall \mathrm{K} \in \mathrm{C}$

sat-solve
$\mathcal{F}_{\mathrm{H}} \wedge \{(\neg\mathrm{b}_1), (\neg\mathrm{b}_4)\}$

Result SAT $\qquad \tau = \{\neg\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \neg\mathrm{b}_4\}$

$\mathrm{C} = \{\{(\neg\mathrm{b}_1), (\neg\mathrm{b}_2)\}, \{(\neg\mathrm{b}_2), (\neg\mathrm{b}_3)\},$
$(\neg\mathrm{b}_3), (\neg\mathrm{b}_4)\}\}$

Basic-IHS ($\mathcal{F}$)

Initialize
while True
  Compute hs
  SAT-solve $\mathcal{F}_{\mathrm{H}} \wedge (\mathcal{F}_{\mathrm{S}} \setminus \mathrm{hs})$
  If UNSAT
    add core to C
  ELSE
    return τ

Implicit Hitting Sets with Bounds

Goals for this Section

1. MaxSat in terms of blocking variables
2. IHS in terms of bounds.

Blocking Variables

- Various modern CDCL SAT solvers implement an API for solving under assumptions.

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\mathrm{assumps} = \{\neg \mathrm{b}_1, \neg \mathrm{b}_3\}$$

Blocking Variables

- Various modern CDCL SAT solvers implement an API for solving under assumptions.
  - assumps: a set of literals
  - sat-assume($\mathcal{F}$, assumps) returns either:

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\mathrm{assumps} = \{\neg \mathrm{b}_1, \neg \mathrm{b}_3\}$$

Blocking Variables

- Various modern CDCL SAT solvers implement an API for solving under assumptions.
  - assumps: a set of literals
  - sat-assume($\mathcal{F}$, assumps) returns either:
    - a solution τ, that satisfies $\mathcal{F}$ and sets $\tau(\mathrm{l}) = 1$ for all $\mathrm{l} \in$ assumps.

$$\mathcal{F}_\mathrm{H} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\text{assumps} = \{\neg \mathrm{b}_1, \neg \mathrm{b}_3\}$$

$$\text{sat-assume}(\mathcal{F}_\mathrm{H}, \text{assumps}) = \text{SAT}$$

$$\tau = \{\neg \mathrm{b}_1, \mathrm{b}_2, \neg \mathrm{b}_3, \mathrm{b}_4\}$$

Blocking Variables

- Various modern CDCL SAT solvers implement an API for solving under assumptions.
  - assumps: a set of literals
  - sat-assume($\mathcal{F}$, assumps) returns either:
    - a solution τ, that satisfies $\mathcal{F}$ and sets $\tau(\mathrm{l}) = 1$ for all $\mathrm{l} \in$ assumps.

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\text{assumps} = \{\neg \mathrm{b}_1, \neg \mathrm{b}_2, \neg \mathrm{b}_3\}$$

Blocking Variables

- Various modern CDCL SAT solvers implement an API for solving under assumptions.
  - assumps: a set of literals
  - sat-assume($\mathcal{F}$, assumps) returns either:
    - a solution τ, that satisfies $\mathcal{F}$ and sets $\tau(\mathrm{l}) = 1$ for all $\mathrm{l} \in$ assumps.
    - unsat if no such solution exists.

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\text{assumps} = \{\neg \mathrm{b}_1, \neg \mathrm{b}_2, \neg \mathrm{b}_3\}$$

$$\text{sat-assume}(\mathcal{F}_{\mathrm{H}}, \text{assumps}) = \text{UNSAT}$$

Blocking Variables

- Various modern CDCL SAT solvers implement an API for solving under ~~assumptions~~ partial assignments.
  - assumps: ~~a set of literals~~ a partial assignment
  - sat-assume($\mathcal{F}$, assumps) returns either:
    - τ, an extension of assumps that satisfies $\mathcal{F}$
    - unsat if no such solution exists.

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\mathrm{assumps} = \{\neg \mathrm{b}_1, \neg \mathrm{b}_2, \neg \mathrm{b}_3\}$$

$$\text{sat-assume}(\mathcal{F}_{\mathrm{H}}, \mathrm{assumps}) = \mathrm{UNSAT}$$

What does this have to do with MaxSAT?

CDCL SAT solvers determine unsatisfiability when learning the empty clause

- By propagating a conflict at decision level 0

What does this have to do with MaxSAT?

CDCL SAT solvers determine unsatisfiability when learning the empty clause

- By propagating a conflict at decision level 0

Explaining unsatisfiability under assumptions

- Trace the reason for unsatisfiability back to assumptions that were necessary for the conflict.
- Essentially:
  - Force the assumptions as the first “decisions”
  - When one of these decisions results in a conflict: trace the reason of the conflict back to the forced assumptions

Extracting cores via Assumptions

- Instrument each soft clause C_i with a new "assumption" variable a_i

  $\rightsquigarrow$ replace C_i with $(C_i \vee a_i)$ for each soft clause C_i
  - $a_i = 0$ switches C_i "on",
    $a_i = 1$ switches C_i "off"
  - $\mathcal{F}_S^E$: soft clauses extended with assumption variables
  - $\neg\mathcal{A} = \{\neg a_i\}$ negation of all assumption variables

Extracting cores via Assumptions

- Instrument each soft clause $\mathrm{C_i}$ with a new "assumption" variable $\mathrm{a_i}$
  $\rightsquigarrow$ replace $\mathrm{C_i}$ with $(\mathrm{C_i} \vee \mathrm{a_i})$ for each soft clause $\mathrm{C_i}$
  - $\mathrm{a_i} = 0$ switches $\mathrm{C_i}$ "on",
    $\mathrm{a_i} = 1$ switches $\mathrm{C_i}$ "off"
  - $\mathcal{F}_\mathrm{S}^\mathrm{E}$: soft clauses extended with assumption variables
  - $\neg\mathcal{A} = \{\neg \mathrm{a_i}\}$ negation of all assumption variables
- MaxSat core: a subset of the assumptions variables

Extracting cores via Assumptions

- Instrument each soft clause C_i with a new "assumption" variable a_i
  $\rightsquigarrow$ replace C_i with $(C_i \vee a_i)$ for each soft clause C_i
  - $a_i = 0$ switches C_i "on",
    $a_i = 1$ switches C_i "off"
  - $\mathcal{F}_S^E$: soft clauses extended with assumption variables
  - $\neg\mathcal{A} = \{\neg a_i\}$ negation of all assumption variables
- MaxSat core: a subset of the assumptions variables
  - Invoke sat-assume$(\mathcal{F}_H \wedge \mathcal{F}_S^E, \neg\mathcal{A})$

Extracting cores via Assumptions

- Instrument each soft clause C_i with a new "assumption" variable a_i
  $\rightsquigarrow$ replace C_i with $(C_i \vee a_i)$ for each soft clause C_i
  - $a_i = 0$ switches C_i "on",
    $a_i = 1$ switches C_i "off"
  - $\mathcal{F}_S^E$: soft clauses extended with assumption variables
  - $\neg\mathcal{A} = \{\neg a_i\}$ negation of all assumption variables
- MaxSat core: a subset of the assumptions variables
  - Invoke sat-assume$(\mathcal{F}_H \wedge \mathcal{F}_S^E, \neg\mathcal{A})$
  - If UNSAT, obtain subset $\kappa_a \subset \mathcal{A}$

Extracting cores via Assumptions

- Instrument each soft clause $\mathrm{C_i}$ with a new "assumption" variable $\mathrm{a_i}$
  $\rightsquigarrow$ replace $\mathrm{C_i}$ with $(\mathrm{C_i} \lor \mathrm{a_i})$ for each soft clause $\mathrm{C_i}$
  - $\mathrm{a_i} = 0$ switches $\mathrm{C_i}$ "on",
    $\mathrm{a_i} = 1$ switches $\mathrm{C_i}$ "off"
  - $\mathcal{F}_{\mathrm{S}}^{\mathrm{E}}$: soft clauses extended with assumption variables
  - $\neg\mathcal{A} = \{\neg \mathrm{a_i}\}$ negation of all assumption variables
- MaxSat core: a subset of the assumptions variables
  - Invoke sat-assume$(\mathcal{F}_{\mathrm{H}} \land \mathcal{F}_{\mathrm{S}}^{\mathrm{E}}, \neg\mathcal{A})$
  - If UNSAT, obtain subset $\kappa_{\mathrm{a}} \subset \mathcal{A}$
  - Map to core $\kappa = \{\mathrm{C_i} \mid \mathrm{a_i} \in \kappa_{\mathrm{a}}\}$

Extracting cores via Assumptions

- Instrument each soft clause C_i with a new "assumption" variable a_i
  $\rightsquigarrow$ replace C_i with $(C_i \vee a_i)$ for each soft clause C_i
  - $a_i = 0$ switches C_i "on",
    $a_i = 1$ switches C_i "off"
  - $\mathcal{F}_S^E$: soft clauses extended with assumption variables
  - $\neg\mathcal{A} = \{\neg a_i\}$ negation of all assumption variables
- MaxSat core: a subset of the assumptions variables
  - Invoke sat-assume$(\mathcal{F}_H \wedge \mathcal{F}_S^E, \neg\mathcal{A})$
  - If UNSAT, obtain subset $\kappa_a \subset \mathcal{A}$
  - Map to core $\kappa = \{C_i \mid a_i \in \kappa_a\}$
  - Used by all core-based MaxSAT algorithms.

Core Extraction - Example

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2), (\neg \mathrm{b}_3), (\neg \mathrm{b}_4)\}$$

Core Extraction - Example

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$
$$\mathcal{F}_{\mathrm{S}}^{\mathrm{E}} = \{(\neg \mathrm{b}_1 \vee \mathrm{a}_1), (\neg \mathrm{b}_2 \vee \mathrm{a}_2), (\neg \mathrm{b}_3 \vee \mathrm{a}_3), (\neg \mathrm{b}_4 \vee \mathrm{a}_4)\}$$

1. Extend Soft Clauses

Core Extraction - Example

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$

$$\mathcal{F}_{\mathrm{S}}^{\mathrm{E}} = \{(\neg \mathrm{b}_1 \vee \mathrm{a}_1), (\neg \mathrm{b}_2 \vee \mathrm{a}_2), (\neg \mathrm{b}_3 \vee \mathrm{a}_3), (\neg \mathrm{b}_4 \vee \mathrm{a}_4)\}$$

$$\text{sat-assume}(\mathcal{F}_{\mathrm{H}} \wedge \mathcal{F}_{\mathrm{S}}^{\mathrm{E}}, \{\neg \mathrm{a}_1, \neg \mathrm{a}_2, \neg \mathrm{a}_3, \neg \mathrm{a}_4\})$$

1. Extend Soft Clauses
2. Invoke SAT-solver under assumptions

Core Extraction - Example

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$
$$\mathcal{F}_{\mathrm{S}}^{\mathrm{E}} = \{(\neg \mathrm{b}_1 \vee \mathrm{a}_1), (\neg \mathrm{b}_2 \vee \mathrm{a}_2), (\neg \mathrm{b}_3 \vee \mathrm{a}_3), (\neg \mathrm{b}_4 \vee \mathrm{a}_4)\}$$

$$\text{sat-assume}(\mathcal{F}_{\mathrm{H}} \wedge \mathcal{F}_{\mathrm{S}}^{\mathrm{E}}, \{\neg \mathrm{a}_1, \neg \mathrm{a}_2, \neg \mathrm{a}_3, \neg \mathrm{a}_4\})$$

Results: UNSAT

1. Extend Soft Clauses
2. Invoke SAT-solver under assumptions

Core Extraction - Example

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$
$\mathcal{F}_{\mathrm{S}}^{\mathrm{E}} = \{(\neg \mathrm{b}_1 \vee \mathrm{a}_1), (\neg \mathrm{b}_2 \vee \mathrm{a}_2), (\neg \mathrm{b}_3 \vee \mathrm{a}_3), (\neg \mathrm{b}_4 \vee \mathrm{a}_4)\}$

sat-assume$(\mathcal{F}_{\mathrm{H}} \wedge \mathcal{F}_{\mathrm{S}}^{\mathrm{E}}, \{\neg \mathrm{a}_1, \neg \mathrm{a}_2, \neg \mathrm{a}_3, \neg \mathrm{a}_4\})$

Results: UNSAT
$\kappa_{\mathrm{a}} = \{\mathrm{a}_1, \mathrm{a}_2\}$

1. Extend Soft Clauses
2. Invoke SAT-solver under assumptions
3. Obtain subset of negated assumptions

Core Extraction - Example

$$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$$
$$\mathcal{F}_{\mathrm{S}}^{\mathrm{E}} = \{(\neg \mathrm{b}_1 \vee \mathrm{a}_1), (\neg \mathrm{b}_2 \vee \mathrm{a}_2), (\neg \mathrm{b}_3 \vee \mathrm{a}_3), (\neg \mathrm{b}_4 \vee \mathrm{a}_4)\}$$

$$\text{sat-assume}(\mathcal{F}_{\mathrm{H}} \wedge \mathcal{F}_{\mathrm{S}}^{\mathrm{E}}, \{\neg \mathrm{a}_1, \neg \mathrm{a}_2, \neg \mathrm{a}_3, \neg \mathrm{a}_4\})$$

Results: UNSAT
$$\kappa_{\mathrm{a}} = \{\mathrm{a}_1, \mathrm{a}_2\} \longrightarrow \kappa = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}$$

1. Extend Soft Clauses
2. Invoke SAT-solver under assumptions
3. Obtain subset of negated assumptions
4. Obtain core.

Core Extraction - Example

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{S}}^{\mathrm{E}} = \{(\neg \mathrm{b}_1 \vee \mathrm{a}_1), (\neg \mathrm{b}_2 \vee \mathrm{a}_2), (\neg \mathrm{b}_3 \vee \mathrm{a}_3), (\neg \mathrm{b}_4 \vee \mathrm{a}_4)\}$

sa

Observation:
Unit soft clauses do not need assumption variables

Results: UNSAT

$\kappa_{\mathrm{a}} = \{\mathrm{a}_1, \mathrm{a}_2\} \longrightarrow \kappa = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}$

1. Extend Soft Clauses
2. Invoke SAT-solver under assumptions
3. Obtain subset of negated assumptions
4. Obtain core.

MaxSat via Blocking Variables

Clauses

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{x})(\neg\mathrm{x}, \mathrm{b}_2)\}$

$\mathcal{F}_{\mathrm{S}} = \{(\neg\mathrm{b}_1), (\neg\mathrm{b}_2)\}$

Assume all soft clauses are unit negative literals.

MaxSat via Blocking Variables

Clauses

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{x})(\neg \mathrm{x}, \mathrm{b}_2)\}$

$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}$

Blocking Vars.

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{x})(\neg \mathrm{x}, \mathrm{b}_2)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2\}$

Assume all soft clauses are unit negative literals.

Blocking Variable: a variable that appears in a ”soft clauses”

MaxSat via Blocking Variables

Clauses

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

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

Find $\tau(\mathcal{F}_H) = 1$ minimizing
$\text{cost}(\tau) = \sum_{C \in \mathcal{F}_S} (1 - \tau(C))$

Blocking Vars.

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

$\mathcal{F}_B = \{b_1, b_2\}$

Find $\tau(\mathcal{F}_H) = 1$ minimizing
$\text{cost}(\tau) = \sum_{b \in \mathcal{F}_B} \tau(b)$

Assign weight to blocking variables instead

MaxSat via Blocking Variables

Clauses

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{x})(\neg \mathrm{x}, \mathrm{b}_2)\}$

$\mathcal{F}_{\mathrm{S}} = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\}$

Find $\tau(\mathcal{F}_{\mathrm{H}}) = 1$ minimizing
$\mathrm{cost}(\tau) = \sum_{\mathrm{C} \in \mathcal{F}_{\mathrm{S}}} (1 - \tau(\mathrm{C}))$

$\kappa = \{(\neg \mathrm{b}_1), (\neg \mathrm{b}_2)\} \subset \mathcal{F}_{\mathrm{S}}$
$\mathcal{F}_{\mathrm{H}} \wedge (\neg \mathrm{b}_1) \wedge (\neg \mathrm{b}_2)$ UNSAT

Blocking Vars.

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{x})(\neg \mathrm{x}, \mathrm{b}_2)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2\}$

Find $\tau(\mathcal{F}_{\mathrm{H}}) = 1$ minimizing
$\mathrm{cost}(\tau) = \sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \tau(\mathrm{b})$

$\kappa = (\mathrm{b}_1, \mathrm{b}_2)$
$\mathcal{F}_{\mathrm{H}} \models (\mathrm{b}_1, \mathrm{b}_2)$

Core: a clause over (set of) blocking variables entailed by $\mathcal{F}_{\mathrm{H}}$.

MaxSat via Blocking Variables

Clauses

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{x})(\neg\mathrm{x}, \mathrm{b}_2)\}$

$\mathcal{F}_{\mathrm{S}} = \{(\neg\mathrm{b}_1), (\neg\mathrm{b}_2)\}$

Find $\tau(\mathcal{F}_{\mathrm{H}}) = 1$ minimizing
$\mathrm{cost}(\tau) = \sum_{\mathrm{C}\in\mathcal{F}_{\mathrm{S}}}(1 - \tau(\mathrm{C}))$

$\kappa = \{(\neg\mathrm{b}_1), (\neg\mathrm{b}_2)\} \subset \mathcal{F}_{\mathrm{S}}$
$\mathcal{F}_{\mathrm{H}} \wedge (\neg\mathrm{b}_1) \wedge (\neg\mathrm{b}_2)$ UNSAT

Blocking Vars.

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{x})(\neg\mathrm{x}, \mathrm{b}_2)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2\}$

Find $\tau(\mathcal{F}_{\mathrm{H}}) = 1$ minimizing
$\mathrm{cost}(\tau) = \sum_{\mathrm{b}\in\mathcal{F}_{\mathrm{B}}} \tau(\mathrm{b})$

$\kappa = (\mathrm{b}_1, \mathrm{b}_2)$
$\mathcal{F}_{\mathrm{H}} \models (\mathrm{b}_1, \mathrm{b}_2)$

Core: a clause over (set of) blocking variables entailed by $\mathcal{F}_{\mathrm{H}}$.

Hitting Set: a subset of blocking variables with non-empty intersection with cores.

Core Extraction with Blocking Variables

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

sat-assume$(\mathcal{F}_{\mathrm{H}}, \{\neg \mathrm{b} \mid \mathrm{b} \in \mathcal{F}_{\mathrm{B}}\})$
Results: UNSAT
$\kappa = \{\mathrm{b}_1, \mathrm{b}_2\}$

1. Core extraction

Core Extraction with Blocking Variables

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{C} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3)\}$ $\quad\quad$ $\mathrm{hs} = \{\mathrm{b}_2\}$

$\text{sat-assume}(\mathcal{F}_{\mathrm{H}}, \{\neg \mathrm{b} \mid \mathrm{b} \in \mathcal{F}_{\mathrm{B}} \setminus \mathrm{hs}\})$

Results: UNSAT

$\kappa = \{\mathrm{b}_3, \mathrm{b}_4\}$

2. Hitting set test:

IHS with bounds

Upper bounds from Core Extraction

IHS with bounds

Upper bounds from Core Extraction

- A new hitting set is not needed after every core.
- Instead, keep extracting cores until solver reports SAT

IHS with bounds

Upper bounds from Core Extraction

- A new hitting set is not needed after every core.
- Instead, keep extracting cores until solver reports SAT
- Obtain model τ of $\mathcal{F}_{\mathrm{H}}$ for which $\mathrm{cost}(\tau)$ is an upper bound on the optimal cost.

IHS with bounds

Upper bounds from Core Extraction

- A new hitting set is not needed after every core.
- Instead, keep extracting cores until solver reports SAT
- Obtain model τ of $\mathcal{F}_{\mathrm{H}}$ for which $\mathrm{cost}(\tau)$ is an upper bound on the optimal cost.

Lower bounds from hitting sets

Proposition:

Let C be any set of cores and hs a minimum-cost hitting set.
Then $|\mathrm{hs}|$ is a lower bound on the optimal cost.

IHS with bounds

Upper bounds from Core Extraction

- A new hitting set is not needed after every core.
- Instead, keep extracting cores until solver reports SAT
- Obtain model τ of $\mathcal{F}_{\mathrm{H}}$ for which $\mathrm{cost}(\tau)$ is an upper bound on the optimal cost.

Lower bounds from hitting sets

Proposition:

Let C be any set of cores and hs a minimum-cost hitting set.
Then $|\mathrm{hs}|$ is a lower bound on the optimal cost.

i.e. sizes of minimum-cost hitting sets over cores provide lower bounds.

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

Basic-IHS ($\mathcal{F}$)

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

Basic-IHS ($\mathcal{F}$)

Initialize

$\mathrm{UB} = \infty$

$\mathrm{LB} = 0$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

Basic-IHS ($\mathcal{F}$)

Initialize

while LB < UB

$\mathrm{UB} = \infty$

$\mathrm{LB} = 0$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{hs} = \text{Min-Hs}\ (\mathcal{F}_{\mathrm{B}}, \emptyset)$

$\mathrm{UB} = \infty$

$\mathrm{LB} = 0$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

Basic-IHS ($\mathcal{F}$)

Initialize

while LB < UB

Compute min-cost hitting set hs

Min-Hs ($\mathcal{F}_{\mathrm{B}}$, C):

minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$

subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$

return: $\{\mathrm{b} \mid \mathrm{b} \text{ set to 1 in opt. soln}\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{hs} = \text{Min-Hs}\ (\mathcal{F}_{\mathrm{B}}, \emptyset)$

Basic-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Compute min-cost hitting set hs

$\mathrm{UB} = \infty$

$\mathrm{LB} = 0$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

Weighted Case

Min-Hs ($\mathcal{F}_{\mathrm{B}}$, C):
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{wt}(\mathrm{b})\mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1\ \forall \kappa \in \mathrm{C}$
return: $\{\mathrm{b} \mid \mathrm{b}$ set to 1 in opt. soln$\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{hs} = \emptyset$

$\mathrm{UB} = \infty$

$\mathrm{LB} = |\emptyset|$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

Basic-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Compute min-cost hitting set hs
  Update LB

Min-Hs ($\mathcal{F}_{\mathrm{B}}, \mathrm{C}$):
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$
return: $\{\mathrm{b} \mid \mathrm{b} \text{ set to 1 in opt. soln}\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{hs} = \emptyset$

$\mathcal{A} = \mathcal{F}_{\mathrm{B}} \setminus \mathrm{hs}$

$\mathrm{UB} = \infty$

$\mathrm{LB} = 0$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

Basic-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions

Min-Hs ($\mathcal{F}_{\mathrm{B}}, \mathrm{C}$):
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \; \forall \kappa \in \mathrm{C}$
return: $\{\mathrm{b} \mid \mathrm{b} \text{ set to 1 in opt. soln}\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

sat-assume$(\mathcal{F}_{\mathrm{H}}, \neg\mathcal{A})$

$\mathcal{A} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{K} = \{\}$

$\mathrm{UB} = \infty$

$\mathrm{LB} = 0$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

Basic-IHS $(\mathcal{F})$

Initialize
while LB < UB
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT

Min-Hs $(\mathcal{F}_{\mathrm{B}}, \mathrm{C})$:
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \; \forall \kappa \in \mathrm{C}$
return: $\{\mathrm{b} \mid \mathrm{b} \text{ set to } 1 \text{ in opt. soln}\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\text{sat-assume}(\mathcal{F}_{\mathrm{H}}, \neg\mathcal{A})$

$\mathcal{A} = \{\cancel{\mathrm{b}_1}, \cancel{\mathrm{b}_2}, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{K} = \{(\mathrm{b}_1, \mathrm{b}_2)\}$

$\mathrm{UB} = \infty$

$\mathrm{LB} = 0$

$\mathrm{C} = \emptyset$

$\text{bestsol} = \varnothing$

Basic-IHS ($\mathcal{F}$)

Initialize
while LB < UB
- Compute min-cost hitting set hs
- Update LB
- Set up assumptions
- Extract cores until SAT

Min-Hs ($\mathcal{F}_{\mathrm{B}}, \mathrm{C}$):

minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$

subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$

return: $\{\mathrm{b} \mid \text{b set to 1 in opt. soln}\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

sat-assume$(\mathcal{F}_{\mathrm{H}}, \neg\mathcal{A})$

$\mathcal{A} = \{\mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{K} = \{(\mathrm{b}_1, \mathrm{b}_2)\}$

$\mathrm{UB} = \infty$

$\mathrm{LB} = 0$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

Basic-IHS $(\mathcal{F})$

Initialize
while LB < UB
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT

Min-Hs $(\mathcal{F}_{\mathrm{B}}, \mathrm{C})$:
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$
return: $\{\mathrm{b} \mid \mathrm{b}$ set to 1 in opt. soln$\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

sat-assume$(\mathcal{F}_{\mathrm{H}}, \neg\mathcal{A})$

$\mathcal{A} = \{\cancel{\mathrm{b}_3}, \cancel{\mathrm{b}_4}\}$

$\mathrm{K} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathrm{UB} = \infty$

$\mathrm{LB} = 0$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

Basic-IHS $(\mathcal{F})$

Initialize
while LB < UB
- Compute min-cost hitting set hs
- Update LB
- Set up assumptions
- Extract cores until SAT

Min-Hs $(\mathcal{F}_{\mathrm{B}}, \mathrm{C})$:
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \; \forall \kappa \in \mathrm{C}$
return: $\{\mathrm{b} \mid \mathrm{b}$ set to 1 in opt. soln$\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\text{sat-assume}(\mathcal{F}_{\mathrm{H}}, \neg\mathcal{A})$

$\mathcal{A} = \{\}$

$\mathrm{K} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathrm{UB} = \infty$

$\mathrm{LB} = 0$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

Basic-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT

Min-Hs ($\mathcal{F}_{\mathrm{B}}$, C):
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$
return: $\{\mathrm{b} \mid \mathrm{b} \text{ set to 1 in opt. soln}\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

sat-assume$(\mathcal{F}_{\mathrm{H}}, \neg\mathcal{A})$

$\mathcal{A} = \{\}$

$\mathrm{K} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\tau = \{\neg\mathrm{b}_1, \mathrm{b}_2, \neg\mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{UB} = \infty$

$\mathrm{LB} = 0$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

Basic-IHS $(\mathcal{F})$

Initialize
while LB < UB
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT

Min-Hs $(\mathcal{F}_{\mathrm{B}}, \mathrm{C})$:
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \;\forall \kappa \in \mathrm{C}$
return: $\{\mathrm{b} \mid \mathrm{b}$ set to 1 in opt. soln$\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{K} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\tau = \{\neg \mathrm{b}_1, \mathrm{b}_2, \neg \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{UB} = \mathrm{cost}(\tau)$

$\mathrm{LB} = 0$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \{\neg \mathrm{b}_1, \mathrm{b}_2, \neg \mathrm{b}_3, \mathrm{b}_4\}$

Basic-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB

Min-Hs ($\mathcal{F}_{\mathrm{B}}, \mathrm{C}$):
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \; \forall \kappa \in \mathrm{C}$
return: $\{\mathrm{b} \mid \mathrm{b} \text{ set to 1 in opt. soln}\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{K} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\tau = \{\neg \mathrm{b}_1, \mathrm{b}_2, \neg \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{UB} = 2$

$\mathrm{LB} = 0$

$\mathrm{C} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathrm{bestsol} = \{\neg \mathrm{b}_1, \mathrm{b}_2, \neg \mathrm{b}_3, \mathrm{b}_4\}$

Basic-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

Min-Hs ($\mathcal{F}_{\mathrm{B}}$, C):

minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$

subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$

return: $\{\mathrm{b} \mid \mathrm{b} \text{ set to 1 in opt. soln}\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

Basic-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

$\mathrm{UB} = 2$

$\mathrm{LB} = 0$

$\mathrm{C} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathrm{bestsol} = \{\neg\mathrm{b}_1, \mathrm{b}_2, \neg\mathrm{b}_3, \mathrm{b}_4\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_\mathrm{H} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_\mathrm{B} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{hs} = \text{Min-Hs}\ (\mathcal{F}_\mathrm{B}, \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_3, \mathrm{b}_4)\})$

$\mathrm{UB} = 2$

$\mathrm{LB} = 0$

$\mathrm{C} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathrm{bestsol} = \{\neg \mathrm{b}_1, \mathrm{b}_2, \neg \mathrm{b}_3, \mathrm{b}_4\}$

Basic-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

Min-Hs ($\mathcal{F}_\mathrm{B}$, C):
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_\mathrm{B}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1\ \forall \kappa \in \mathrm{C}$
return: $\{\mathrm{b} \mid \mathrm{b}$ set to 1 in opt. soln$\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{hs} = \{\mathrm{b}_1, \mathrm{b}_4\}$

Basic-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

$\mathrm{UB} = 2$

$\mathrm{LB} = |\{\mathrm{b}_1, \mathrm{b}_4\}|$

$\mathrm{C} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathrm{bestsol} = \{\neg \mathrm{b}_1, \mathrm{b}_2, \neg \mathrm{b}_3, \mathrm{b}_4\}$

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{hs} = \{\mathrm{b}_1, \mathrm{b}_4\}$

$\mathrm{UB} = 2$

$\mathrm{LB} = 2$

$\mathrm{C} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathrm{bestsol} = \{\neg\mathrm{b}_1, \mathrm{b}_2, \neg\mathrm{b}_3, \mathrm{b}_4\}$

Basic-IHS ($\mathcal{F}$)

Initialize
while LB < UB
    Compute min-cost hitting set hs
    Update LB
    Set up assumptions
    Extract cores until SAT
    Update UB
    Add cores to C
return bestsol

Solving (unweighted) MaxSat with IHS

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{hs} = \{\mathrm{b}_1, \mathrm{b}_4\}$

Basic-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C
return bestsol

$\mathrm{UB} = 2$

$\mathrm{LB} = 2$

LB need to be increased
to optimum before termination

$\mathrm{C} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathrm{bestsol} = \{\neg\mathrm{b}_1, \mathrm{b}_2, \neg\mathrm{b}_3, \mathrm{b}_4\}$

Optimizations in Solvers

Solvers implementing the implicit hittings set approach include several optimizations, such as

- a non-optimal hitting sets for extracting several cores before/between hitting set computations, [Davies and Bacchus, 2011, 2013b,a; Saikko, Berg, and Järvisalo, 2016a]
- LP-solving techniques such as reduced cost fixing in the hitting sets [Bacchus, Hyttinen, Järvisalo, and Saikko, 2017]
- ...

Some of these optimizations are integral for making the solvers competitive.

Implicit Hitting Sets

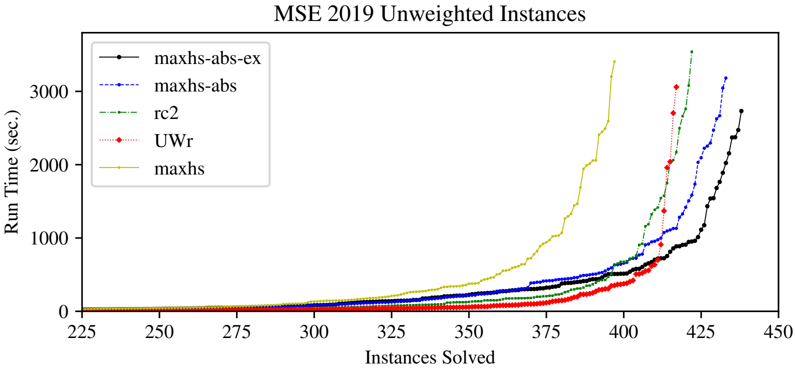
- Effective on range of MaxSat problems including large ones.
- Superior to other methods when there are many distinct weights.
- Usually superior to CPLEX for solving MaxSAT instances.

Abstract Cores

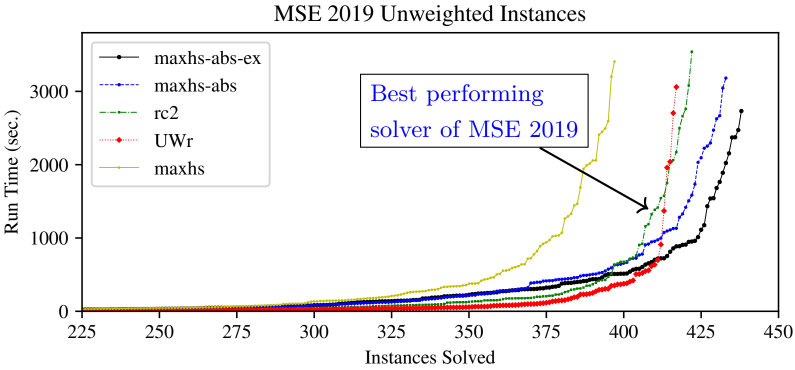
Goals for this section

1. What are the weaknesses of IHS for MaxSAT?
2. What are abstract cores and how do they address the weaknesses?
3. How can abstract core reasoning be incorporated into IHS?
4. What effect does it have?

Goals for this section



Goals for this section



Drawbacks of IHS

Davies [2013]

Main motivation for abstract cores:
There exists MaxSAT instances on which IHS needs an exponential number of cores.

Drawbacks of IHS

Davies [2013]

Main motivation for abstract cores:
There exists MaxSAT instances on which IHS needs an exponential number of cores.

$$\mathcal{F}_{\mathrm{H}} = \left\{ \mathrm{CNF}(\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{b}_{\mathrm{i}} \geq \mathrm{r}) \right\}$$

$$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \ldots, \mathrm{b}_{\mathrm{n}}\}$$

Any solution assigns r b-vars to 1
$\Rightarrow$ any subset of b-vars with at least $(\mathrm{n} - \mathrm{r}) + 1$ elements has one set to 1

Drawbacks of IHS

Davies [2013]

Main motivation for abstract cores:
There exists MaxSAT instances on which IHS needs an exponential number of cores.

$$\mathcal{F}_{\mathrm{H}} = \left\{ \mathrm{CNF}\left(\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{b_i} \geq \mathrm{r}\right) \right\}$$

$$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b_1}, \ldots, \mathrm{b_n}\}$$

$\mathrm{n} = 8, \quad \mathrm{r} = 4$

$\kappa_1 = (\mathrm{b_{i_1}}, \mathrm{b_{i_2}}, \mathrm{b_{i_3}}, \mathrm{b_{i_4}}, \mathrm{b_{i_5}})$

is a core for any $\mathrm{i_1}, \mathrm{i_2}, \mathrm{i_3}, \mathrm{i_4}, \mathrm{i_5}$

Intuition:

Any $\kappa \subset \mathcal{F}_{\mathrm{B}}$ s.t $|\kappa| = (\mathrm{n} - \mathrm{r}) + 1$ is a core.

Drawbacks of IHS

Davies [2013]

Main motivation for abstract cores:
There exists MaxSAT instances on which IHS needs an exponential number of cores.

$$\mathcal{F}_H = \left\{ \mathrm{CNF}(\sum_{i=1}^{n} b_i \geq r) \right\}$$

$$\mathcal{F}_B = \{b_1, \ldots, b_n\}$$

$n = 8, \quad r = 4$

$\kappa_1 = (b_{i_1}, b_{i_2}, b_{i_3}, b_{i_4}, b_{i_5})$

is a core for any i_1, i_2, i_3, i_4, i_5

$\kappa_2 = (b_{i_1}, b_{i_2}, b_{i_3}, b_{i_4})$

is not a core for any i_1, i_2, i_3, i_4

Intuition:

Any $\kappa \subset \mathcal{F}_B$ s.t $|\kappa| = (n - r) + 1$ is a core.

Drawbacks of IHS

Davies [2013]

Main motivation for abstract cores:
There exists MaxSAT instances on which IHS needs an exponential number of cores.

$$\mathcal{F}_\mathrm{H} = \left\{ \mathrm{CNF}(\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{b_i} \geq \mathrm{r}) \right\}$$

$$\mathcal{F}_\mathrm{B} = \{\mathrm{b}_1, \ldots, \mathrm{b_n}\}$$

$\mathrm{n} = 8, \quad \mathrm{r} = 4$

$\kappa_1 = (\mathrm{b}_{\mathrm{i}_1}, \mathrm{b}_{\mathrm{i}_2}, \mathrm{b}_{\mathrm{i}_3}, \mathrm{b}_{\mathrm{i}_4}, \mathrm{b}_{\mathrm{i}_5})$

is a core for any $\mathrm{i}_1, \mathrm{i}_2, \mathrm{i}_3, \mathrm{i}_4, \mathrm{i}_5$

$\kappa_2 = (\mathrm{b}_{\mathrm{i}_1}, \mathrm{b}_{\mathrm{i}_2}, \mathrm{b}_{\mathrm{i}_3}, \mathrm{b}_{\mathrm{i}_4})$

is not a core for any $\mathrm{i}_1, \mathrm{i}_2, \mathrm{i}_3, \mathrm{i}_4$

Intuition:

Any $\kappa \subset \mathcal{F}_\mathrm{B}$ s.t $|\kappa| = (\mathrm{n} - \mathrm{r}) + 1$ is a core.

IHS needs to extract all $\binom{\mathrm{n}}{(\mathrm{n}-\mathrm{r})+1}$ of them.

Drawbacks of IHS

Davies [2013]

Main motivation for abstract cores:
There exists MaxSAT instances on which IHS needs an exponential number of cores.

$$\mathcal{F}_H = \left\{ \mathrm{CNF}(\sum_{i=1}^{n} b_i \geq r) \right\}$$

$$\mathcal{F}_B = \{b_1, \ldots, b_n\}$$

Blocking variables are exchangeable:
cores are defined by the number of them,
not the identity of them

$n = 8, \quad r = 4$

$\kappa_1 = (b_{i_1}, b_{i_2}, b_{i_3}, b_{i_4}, b_{i_5})$
is a core for any i_1, i_2, i_3, i_4, i_5

$\kappa_2 = (b_{i_1}, b_{i_2}, b_{i_3}, b_{i_4})$
is not a core for any i_1, i_2, i_3, i_4

Intuition:
Any $\kappa \subset \mathcal{F}_B$ s.t $|\kappa| = (n - r) + 1$ is a core.
IHS needs to extract all $\binom{n}{(n-r)+1}$ of them.

More Specifically

$$\mathcal{F}_{\mathrm{H}} = \left\{ \mathrm{CNF}\left(\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{b}_{\mathrm{i}} \geq \mathrm{r}\right) \right\}$$

$$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \ldots, \mathrm{b}_{\mathrm{n}}\}$$

Opt. cost $= \mathrm{r}$

Let C be any set of cores.

Assume $\mathrm{S} \notin \mathrm{C}$ for some $|\mathrm{S}| = \mathrm{n} - \mathrm{r} + 1$

More Specifically

$$\mathcal{F}_{\mathrm{H}} = \left\{ \mathrm{CNF}(\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{b_i} \geq \mathrm{r}) \right\}$$

$$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \dots, \mathrm{b_n}\}$$

Opt. cost $= \mathrm{r}$

Let C be any set of cores.

Assume $\mathrm{S} \notin \mathrm{C}$ for some $|\mathrm{S}| = \mathrm{n} - \mathrm{r} + 1$

Then $\mathcal{F}_{\mathrm{B}} \setminus \mathrm{S}$ hits every $\kappa \in \mathrm{C}$

$|\text{Min-Hs}(\mathcal{F}_{\mathrm{B}}, \emptyset)| \leq |\mathcal{F}_{\mathrm{B}} \setminus \mathrm{S}| = \mathrm{r} - 1 < \mathrm{r} = \text{Opt. cost}$

More Specifically

$$\mathcal{F}_{\mathrm{H}} = \left\{ \mathrm{CNF}\left(\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{b}_{\mathrm{i}} \geq \mathrm{r}\right) \right\}$$

$$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \ldots, \mathrm{b}_{\mathrm{n}}\}$$

Opt. cost $= \mathrm{r}$

Let C be any set of cores.

Assume $\mathrm{S} \notin \mathrm{C}$ for some $|\mathrm{S}| = \mathrm{n} - \mathrm{r} + 1$

Then $\mathcal{F}_{\mathrm{B}} \setminus \mathrm{S}$ hits every $\kappa \in \mathrm{C}$

$|\text{Min-Hs}(\mathcal{F}_{\mathrm{B}}, \emptyset)| \leq |\mathcal{F}_{\mathrm{B}} \setminus \mathrm{S}| = \mathrm{r} - 1 < \mathrm{r} = \text{Opt. cost}$

$\Rightarrow$ IHS does not terminate.

Weakness shown in practice

https://maxsat-evaluations.github.io/2019/rankings.html

Benchmarks	MaxHS	MaxHS 4.0	RC2
drmx-atmostk (W) (11)	3	11	11
drmx-atmostk (UW) (17)	3	17	17

- Results from 2019 MSE and our paper.
- MaxHS: an IHS solver w/o abstract cores
- MaxHS 4.0 an IHS solver with abstract cores
- RC2: the best performing solver in the 2020 MaxSat evaluation

Abstract Cores

Research Question

Does there exists a compact representation of large sets of cores that IHS can reason over?

Abstract cores

Idea: What happens if we introduce literals that count the number of blocking variables set to true?

(similar to variables that have been successfully used in core-guided solvers)

Abstract cores

Idea: What happens if we introduce literals that count the number of blocking variables set to true?

(similar to variables that have been successfully used in core-guided solvers)

$$\mathrm{AB} = \{\mathrm{b}_1, \ldots, \mathrm{b}_5\} \subset \mathcal{F}_\mathrm{B}$$

$$\mathcal{F}_\mathrm{H} = \left\{ \mathrm{CNF}(\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{b}_\mathrm{i} \geq \mathrm{r}) \right\}$$

$$\mathcal{F}_\mathrm{B} = \{\mathrm{b}_1, \ldots, \mathrm{b}_\mathrm{n}\}$$

Abstract cores

Idea: What happens if we introduce literals that count the number of blocking variables set to true?

(similar to variables that have been successfully used in core-guided solvers)

$$\mathcal{F}_H = \left\{ \mathrm{CNF}\Big(\sum_{i=1}^{n} b_i \geq r\Big) \right\}$$

$$\mathcal{F}_B = \{b_1, \ldots, b_n\}$$

$$s^{AB}[i] \leftrightarrow \left(\sum_{b \in AB} b \geq i\right)$$

$AB = \{b_1, \ldots, b_5\} \subset \mathcal{F}_B$

Define $s^{AB}[i]$

Abstract cores

Idea: What happens if we introduce literals that count the number of blocking variables set to true?

(similar to variables that have been successfully used in core-guided solvers)

$$\mathcal{F}_H = \left\{ \mathrm{CNF}\left(\sum_{i=1}^{n} b_i \geq r\right) \right\}$$

$$\mathcal{F}_B = \{b_1, \ldots, b_n\}$$

$$s^{AB}[i] \leftrightarrow \left(\sum_{b \in AB} b \geq i\right)$$

$\uparrow$

Note: Can be encoded as CNF

$$AB = \{b_1, \ldots, b_5\} \subset \mathcal{F}_B$$

Define $s^{AB}[i]$

Abstract cores

Idea: What happens if we introduce literals that count the number of blocking variables set to true?

(similar to variables that have been successfully used in core-guided solvers)

$$\mathcal{F}_H = \left\{ \mathrm{CNF}\left(\sum_{i=1}^{n} b_i \geq r\right) \right\}$$

$$\mathcal{F}_B = \{b_1, \ldots, b_n\}$$

$$s^{AB}[i] \leftrightarrow \left(\sum_{b \in AB} b \geq i\right)$$

$AB = \{b_1, \ldots, b_5\} \subset \mathcal{F}_B$

Define $s^{AB}[i]$

Consider: $(b_7, s^{AB}[3], b_n)$

$s^{AB}[3] = 1$ means
$\sum_{b \in S} b \geq 1 = (\bigvee_{b \in S} b)$
for any $S \subset AB$ with
$|S| = 3 (= 5 - 3 + 1)$

Abstract cores

Idea: What happens if we introduce literals that count the number of blocking variables set to true?

(similar to variables that have been successfully used in core-guided solvers)

$$\mathcal{F}_H = \left\{ \mathrm{CNF}\left(\sum_{i=1}^{n} b_i \geq r\right) \right\}$$

$$\mathcal{F}_B = \{b_1, \ldots, b_n\}$$

$$s^{AB}[i] \leftrightarrow \left(\sum_{b \in AB} b \geq i\right)$$

$AB = \{b_1, \ldots, b_5\} \subset \mathcal{F}_B$

Define $s^{AB}[i]$

Consider: $(b_7, s^{AB}[3], b_n)$

$(b_7, b_1, b_2, b_3, b_n)$

$s^{AB}[3] = 1$ means
$\sum_{b \in S} b \geq 1 = (\bigvee_{b \in S} b)$
for any $S \subset AB$ with
$|S| = 3 (= 5 - 3 + 1)$

Abstract cores

Idea: What happens if we introduce literals that count the number of blocking variables set to true?

(similar to variables that have been successfully used in core-guided solvers)

$$\mathcal{F}_H = \left\{ \mathrm{CNF}\left(\sum_{i=1}^{n} b_i \geq r\right) \right\}$$

$$\mathcal{F}_B = \{b_1, \ldots, b_n\}$$

$$s^{AB}[i] \leftrightarrow \left(\sum_{b \in AB} b \geq i\right)$$

$AB = \{b_1, \ldots, b_5\} \subset \mathcal{F}_B$

Define $s^{AB}[i]$

Consider: $(b_7, s^{AB}[3], b_n)$

$(b_7, b_1, b_2, b_3, b_n)$

$(b_7, b_3, b_4, b_2, b_n)$

Abstract cores

Idea: What happens if we introduce literals that count the number of blocking variables set to true?

(similar to variables that have been successfully used in core-guided solvers)

$$\mathcal{F}_{\mathrm{H}} = \left\{ \mathrm{CNF}\left(\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{b_i} \geq \mathrm{r}\right) \right\}$$

$$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b_1}, \ldots, \mathrm{b_n}\}$$

$$\mathrm{s^{AB}[i]} \leftrightarrow \left(\textstyle\sum_{\mathrm{b} \in \mathrm{AB}} \mathrm{b} \geq \mathrm{i}\right)$$

$$\mathrm{AB} = \{\mathrm{b_1}, \ldots, \mathrm{b_5}\} \subset \mathcal{F}_{\mathrm{B}}$$

Define $\mathrm{s^{AB}[i]}$

Consider: $(\mathrm{b_7}, \mathrm{s^{AB}[3]}, \mathrm{b_n})$

$(\mathrm{b_7}, \mathrm{b_1}, \mathrm{b_2}, \mathrm{b_3}, \mathrm{b_n})$

$(\mathrm{b_7}, \mathrm{b_3}, \mathrm{b_4}, \mathrm{b_2}, \mathrm{b_n})$

$(\mathrm{b_7}, \mathrm{b_4}, \mathrm{b_2}, \mathrm{b_1}, \mathrm{b_n})$

Abstract cores

Idea: What happens if we introduce literals that count the number of blocking variables set to true?

(similar to variables that have been successfully used in core-guided solvers)

$$\mathrm{AB} = \{b_1, \ldots, b_5\} \subset \mathcal{F}_B$$

$$\mathcal{F}_H = \left\{ \mathrm{CNF}\left(\sum_{i=1}^{n} b_i \geq r\right) \right\}$$

Define $s^{\mathrm{AB}}[i]$

$$\mathcal{F}_B = \{b_1, \ldots, b_n\}$$

Consider: $(b_7, s^{\mathrm{AB}}[3], b_n)$

$$s^{\mathrm{AB}}[i] \leftrightarrow \left(\sum_{b \in \mathrm{AB}} b \geq i\right)$$

$(b_7, b_1, b_2, b_3, b_n)$

$(b_7, b_4, b_2, b_1, b_n)$

$(b_7, b_3, b_4, b_2, b_n)$

Abstract cores

Idea: What happens if we introduce literals that count the number of blocking variables set to true?

(similar to variables that have been successfully used in core-guided solvers)

Terminology:
AB is an abstraction set

$s^{AB}[i]$ is an abstraction variable

The definition of $s^{AB}[i]$ is
$s^{AB}[i] \leftrightarrow \left(\sum_{b \in AB} b \geq i\right)$

$AB = \{b_1, \ldots, b_5\} \subset \mathcal{F}_B$

Define $s^{AB}[i]$

Consider: $(b_7, s^{AB}[3], b_n)$

Abstract cores

Idea: What happens if we introduce literals that count the number of blocking variables set to true?

(similar to variables that have been successfully used in core-guided solvers)

Terminology:
AB is an abstraction set

$s^{AB}[i]$ is an abstraction variable

The definition of $s^{AB}[i]$ is
$s^{AB}[i] \leftrightarrow \left(\sum_{b \in AB} b \geq i\right)$

$AB = \{b_1, \ldots, b_5\} \subset \mathcal{F}_B$

Define $s^{AB}[i]$

Consider: $(b_7, s^{AB}[3], b_n)$

Abstract Core:
a clause over abstraction and blocking variables that is entailed by $\mathcal{F}_H$ and the definitions of abstraction variables

Abstract cores

Idea: What happens if we introduce literals that count the number of blocking variables set to true?

(similar to variables that have been successfully used in core-guided solvers)

Terminology:
AB is an abstraction set

$s^{AB}[i]$ is an abstraction variable

The definition of $s^{AB}[i]$ is
$s^{AB}[i] \leftrightarrow \left(\sum_{b \in AB} b \geq i\right)$

$AB = \{b_1, \ldots, b_5\} \subset \mathcal{F}_B$

Define $s^{AB}[i]$

Could use other definitions
Summations successfull
in core-guided solvers.

Abstract Core:
a clause over abstraction and blocking variables that is entailed by $\mathcal{F}_H$ and the definitions of abstraction variables

Abstract cores are expressive

Proposition

An abstract core containing the abstraction variables $\{s^{AB^1}[j_1], \ldots, s^{AB^k}[j_k]\}$ is equivalent to the conjunction of

$$\prod_{i=1}^{k} \binom{|AB^i|}{|AB^i| - j_i + 1}$$

regular cores.

Abstract cores are expressive

Proposition

An abstract core containing the abstraction variables $\{s^{AB^1}[j_1], \ldots, s^{AB^k}[j_k]\}$ is equivalent to the conjunction of

$$\prod_{i=1}^{k} \binom{|AB^i|}{|AB^i| - j_i + 1}$$

regular cores.

Two Questions remain:

1. How to compute abstraction sets?
2. How to extract and reason over abstract cores in IHS?

Computing Abstraction Sets

Ideally

Identify a set $\mathrm{S} \subset \mathcal{F}_{\mathrm{B}}$ of exchangeable blocking variables.

Computing Abstraction Sets

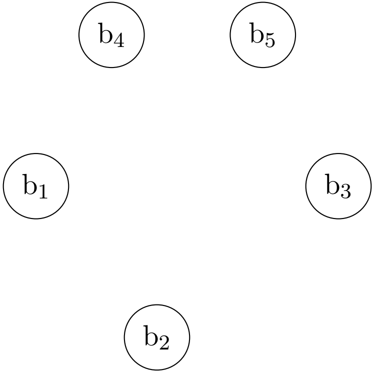
LB: 0

Ideally

Identify a set $S \subset \mathcal{F}_B$ of exchangeable blocking variables.

In practice

Form abstraction sets over blocking variables that appear frequently in cores together.



Computing Abstraction Sets

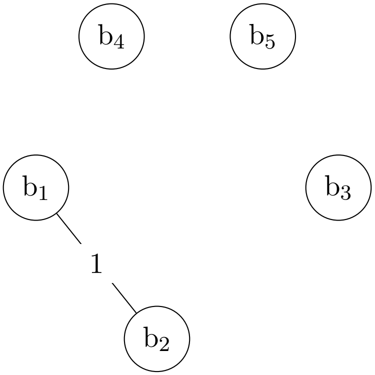
Core: (b_1, b_2) LB: 0

Ideally

Identify a set $S \subset \mathcal{F}_B$ of exchangeable blocking variables.

In practice

Form abstraction sets over blocking variables that appear frequently in cores together.



Computing Abstraction Sets

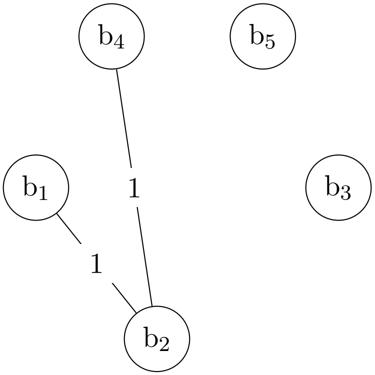
Ideally

Identify a set $S \subset \mathcal{F}_B$ of exchangeable blocking variables.

In practice

Form abstraction sets over blocking variables that appear frequently in cores together.

Core: (b_2, b_4) LB: 0



Computing Abstraction Sets

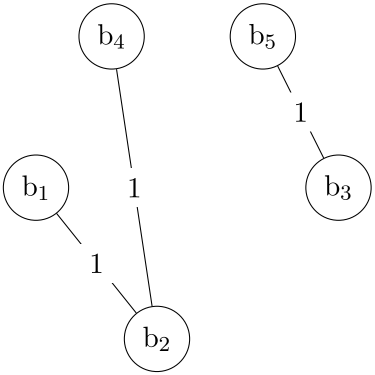
Core: (b_3, b_5) LB: 0

Ideally

Identify a set $S \subset \mathcal{F}_B$ of exchangeable blocking variables.

In practice

Form abstraction sets over blocking variables that appear frequently in cores together.



Computing Abstraction Sets

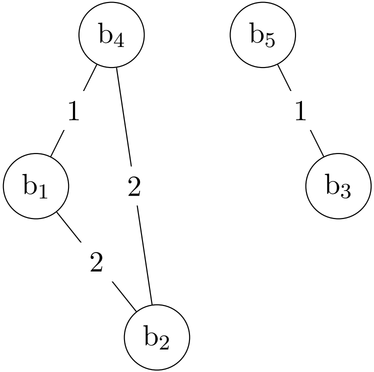
Core: (b_1, b_2, b_4) LB: 0

Ideally

Identify a set $S \subset \mathcal{F}_B$ of exchangeable blocking variables.

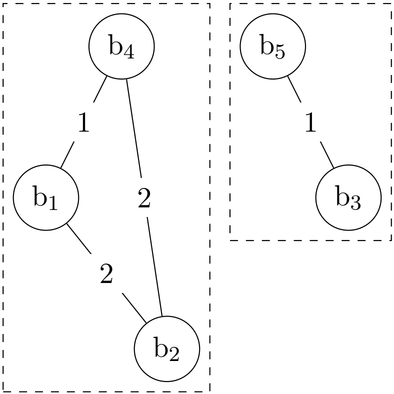
In practice

Form abstraction sets over blocking variables that appear frequently in cores together.



Computing Abstraction Sets

Clustering LB: 0



Ideally

Identify a set $S \subset \mathcal{F}_B$ of exchangeable blocking variables.

In practice

Form abstraction sets over blocking variables that appear frequently in cores together.

Recall: IHS needs to increase LB to optimum

Computing Abstraction Sets

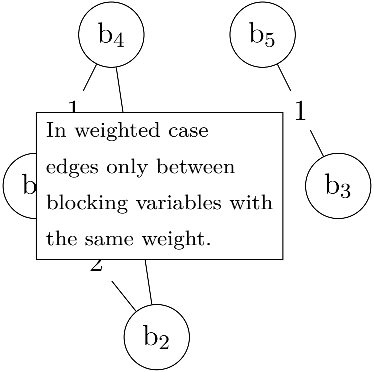
Ideally

Identify a set $S \subset \mathcal{F}_B$ of exchangeable blocking variables.

In practice

Form abstraction sets over blocking variables that appear frequently in cores together.

Recall: IHS needs to increase LB to optimum



IHS with abstract core reasoning

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$ Abstract-IHS ($\mathcal{F}$)

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

IHS with abstract core reasoning

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

Abstract-IHS ($\mathcal{F}$)

Initialize

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{UB} = \infty$ $\mathrm{LB} = 0$

$\mathcal{AB} = \emptyset$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

IHS with abstract core reasoning

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

Abstract-IHS ($\mathcal{F}$)

Initialize

while LB < UB

$\mathrm{UB} = \infty$ $\quad$ $\mathrm{LB} = 0$

$\mathcal{AB} = \emptyset$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

IHS with abstract core reasoning

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4),$
$\bigwedge_{\mathrm{i}=1}^{2} \mathrm{CNF}\left((\mathrm{b}_2 + \mathrm{b}_3 \geq \mathrm{i}) \to \mathrm{s}^{\mathrm{AB}}[\mathrm{i}]\right)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

Abstract-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Update $\mathcal{AB}$

$\mathrm{UB} = \infty$ $\mathrm{LB} = 0$

$\mathcal{AB} = \{\mathrm{AB} = \{\mathrm{b}_2, \mathrm{b}_3\}\}$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

IHS with abstract core reasoning

$\mathcal{F}_H = \{(b_1, b_2), (b_2, b_3), (b_3, b_4),$
$\bigwedge_{i=1}^{2} \mathrm{CNF}\left((b_2 + b_3 \geq i) \rightarrow s^{AB}[i]\right)\}$

$\mathcal{F}_B = \{b_1, b_2, b_3, b_4\}$

$hs = \text{Min-Abs}\ (\mathcal{F}_B, \emptyset, \mathcal{AB})$

Abstract-IHS $(\mathcal{F})$

Initialize
while LB < UB
Update $\mathcal{AB}$
Compute min-cost hitting set hs

$\sum_{b \in AB} b - k \cdot s^{AB}[k] \geq 0$

$\sum_{b \in AB} b - |AB| \cdot s^{AB}[k] < k$

Min-Abs $(\mathcal{F}_B, C, \mathcal{AB})$:
minimize: $\sum_{b \in \mathcal{F}_B} b$
subject to: $\sum_{b \in \kappa} b \geq 1\ \forall \kappa \in C$
$\left(\sum_{b \in AB} b \geq k\right) \leftrightarrow s^{AB}[k]\ \forall AB \in \mathcal{AB}$
return: $\{b \mid b \text{ set to 1 in opt. soln}\}$

IHS with abstract core reasoning

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4),$
$\bigwedge_{\mathrm{i}=1}^{2} \mathrm{CNF}\left((\mathrm{b}_2 + \mathrm{b}_3 \geq \mathrm{i}) \rightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{i}]\right)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{hs} = \emptyset$

Abstract-IHS $(\mathcal{F})$

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB

$\mathrm{UB} = \infty$

$\mathrm{LB} = |\emptyset|$

$\mathcal{AB} = \{\mathrm{AB} = \{\mathrm{b}_2, \mathrm{b}_3\}\}$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

Min-Abs $(\mathcal{F}_{\mathrm{B}}, \mathrm{C}, \mathcal{AB})$:
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$
$\left(\sum_{\mathrm{b} \in \mathrm{AB}} \mathrm{b} \geq \mathrm{k}\right) \leftrightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{k}] \ \forall \mathrm{AB} \in \mathcal{AB}$
return: $\{\mathrm{b} \mid \mathrm{b} \text{ set to 1 in opt. soln}\}$

IHS with abstract core reasoning

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4),$
$\bigwedge_{\mathrm{i}=1}^{2} \mathrm{CNF}\left((\mathrm{b}_2 + \mathrm{b}_3 \geq \mathrm{i}) \rightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{i}]\right)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{hs} = \emptyset$

$\mathcal{A} = \mathrm{ABSTRACT}(\mathcal{F}_{\mathrm{B}}, \mathrm{hs}, \mathcal{AB})$
$= \{\mathrm{b}_1, \mathrm{s}^{\mathrm{AB}}[1], \mathrm{b}_4\}$

$\mathrm{AB} = \{\mathrm{b}_2, \mathrm{b}_3\}$

Abstract-IHS ($\mathcal{F}$)

Initialize
while LB < UB
- Update $\mathcal{AB}$
- Compute min-cost hitting set hs
- Update LB
- Set up assumptions

```
ABSTRACT(F_B, hs, AB)
    A ← {b | b ∈ F_B − hs}
    foreach AB ∈ AB do
        A ← A − {b | b ∈ AB}
        A ← A ∪ {s^AB[|AB ∩ hs| + 1]}
    return A
```

$\mathrm{ABSTRACT}(\mathcal{F}_{\mathrm{B}}, \mathrm{hs}, \mathcal{AB})$
- $\mathcal{A} \leftarrow \{\mathrm{b} \mid \mathrm{b} \in \mathcal{F}_{\mathrm{B}} - \mathrm{hs}\}$
- foreach $\mathrm{AB} \in \mathcal{AB}$ do
  - $\mathcal{A} \leftarrow \mathcal{A} - \{\mathrm{b} \mid \mathrm{b} \in \mathrm{AB}\}$
  - $\mathcal{A} \leftarrow \mathcal{A} \cup \{\mathrm{s}^{\mathrm{AB}}[|\mathrm{AB} \cap \mathrm{hs}| + 1]\}$
- return $\mathcal{A}$

IHS with abstract core reasoning

$\mathcal{F}_H = \{(b_1, b_2), (b_2, b_3), (b_3, b_4),$
$\bigwedge_{i=1}^{2} \mathrm{CNF}\left((b_2 + b_3 \geq i) \to s^{AB}[i]\right)\}$

$\mathcal{F}_B = \{b_1, b_2, b_3, b_4\}$

sat-assume$(\mathcal{F}_H, \neg\mathcal{A})$

$\mathcal{A} = \{b_1, s^{AB}[1], b_4\}$

$K = \{\}$

Abstract-IHS $(\mathcal{F})$

Initialize
while LB < UB
- Update $\mathcal{AB}$
- Compute min-cost hitting set hs
- Update LB
- Set up assumptions
- Extract cores until SAT

$UB = \infty \qquad LB = 0$

$\mathcal{AB} = \{AB = \{b_2, b_3\}\}$

$C = \emptyset$

$\text{bestsol} = \varnothing$

Min-Abs $(\mathcal{F}_B, C, \mathcal{AB})$:
minimize: $\sum_{b \in \mathcal{F}_B} b$
subject to: $\sum_{b \in \kappa} b \geq 1 \ \forall \kappa \in C$
$\left(\sum_{b \in AB} b \geq k\right) \leftrightarrow s^{AB}[k] \ \forall AB \in \mathcal{AB}$
return: $\{b \mid b \text{ set to 1 in opt. soln}\}$

IHS with abstract core reasoning

$\mathcal{F}_H = \{(b_1, b_2), (b_2, b_3), (b_3, b_4),$
$\bigwedge_{i=1}^{2} \mathrm{CNF}\left((b_2 + b_3 \geq i) \rightarrow s^{AB}[i]\right)\}$

$\mathcal{F}_B = \{b_1, b_2, b_3, b_4\}$

sat-assume$(\mathcal{F}_H, \neg\mathcal{A})$

$\mathcal{A} = \{b_1, \cancel{s^{AB}[1]}, b_4\}$

$K = \{(s^{AB}[1])\}$

Abstract-IHS $(\mathcal{F})$

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT

$UB = \infty$ $LB = 0$

$\mathcal{AB} = \{AB = \{b_2, b_3\}\}$

$C = \emptyset$

bestsol $= \varnothing$

Min-Abs $(\mathcal{F}_B, C, \mathcal{AB})$:
minimize: $\sum_{b \in \mathcal{F}_B} b$
subject to: $\sum_{b \in \kappa} b \geq 1 \ \forall \kappa \in C$
$\left(\sum_{b \in AB} b \geq k\right) \leftrightarrow s^{AB}[k] \ \forall AB \in \mathcal{AB}$
return: $\{b \mid b$ set to 1 in opt. soln$\}$

IHS with abstract core reasoning

$\mathcal{F}_\mathrm{H} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4),$
$\bigwedge_{\mathrm{i}=1}^{2} \mathrm{CNF}\left((\mathrm{b}_2 + \mathrm{b}_3 \geq \mathrm{i}) \to \mathrm{s}^{\mathrm{AB}}[\mathrm{i}]\right)\}$

$\mathcal{F}_\mathrm{B} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

sat-assume$(\mathcal{F}_\mathrm{H}, \neg\mathcal{A})$

$\mathcal{A} = \{\mathrm{b}_1, \mathrm{b}_4\}$

$\mathrm{K} = \{(\mathrm{s}^{\mathrm{AB}}[1])\}$

Abstract-IHS $(\mathcal{F})$

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT

$\mathrm{UB} = \infty$ $\mathrm{LB} = 0$

$\mathcal{AB} = \{\mathrm{AB} = \{\mathrm{b}_2, \mathrm{b}_3\}\}$

$\mathrm{C} = \emptyset$

$\mathrm{bestsol} = \varnothing$

Min-Abs $(\mathcal{F}_\mathrm{B}, \mathrm{C}, \mathcal{AB})$:
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_\mathrm{B}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$
$\left(\sum_{\mathrm{b} \in \mathrm{AB}} \mathrm{b} \geq \mathrm{k}\right) \leftrightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{k}] \ \forall \mathrm{AB} \in \mathcal{AB}$
return: $\{\mathrm{b} \mid \mathrm{b} \text{ set to 1 in opt. soln}\}$

IHS with abstract core reasoning

$\mathcal{F}_{\mathrm{H}} = \{(b_1, b_2), (b_2, b_3), (b_3, b_4),$
$\bigwedge_{i=1}^{2} \mathrm{CNF}\left((b_2 + b_3 \geq i) \rightarrow s^{AB}[i]\right)\}$

$\mathcal{F}_{\mathrm{B}} = \{b_1, b_2, b_3, b_4\}$

$\text{sat-assume}(\mathcal{F}_{\mathrm{H}}, \neg\mathcal{A})$

$\mathcal{A} = \{b_1, b_4\}$

$K = \{(s^{AB}[1])\}$

$\tau = \{\neg b_1, b_2, b_3, \neg b_4\}$

$UB = \infty$ $LB = 0$

$\mathcal{AB} = \{AB = \{b_2, b_3\}\}$

$C = \emptyset$

$\text{bestsol} = \varnothing$

Abstract-IHS ($\mathcal{F}$)

Initialize
while LB < UB
 Update $\mathcal{AB}$
 Compute min-cost hitting set hs
 Update LB
 Set up assumptions
 Extract cores until SAT

Min-Abs ($\mathcal{F}_{\mathrm{B}}, \mathrm{C}, \mathcal{AB}$):
minimize: $\sum_{b \in \mathcal{F}_{\mathrm{B}}} b$
subject to: $\sum_{b \in \kappa} b \geq 1 \ \forall \kappa \in C$
$\left(\sum_{b \in AB} b \geq k\right) \leftrightarrow s^{AB}[k] \ \forall AB \in \mathcal{AB}$
return: $\{b \mid b$ set to 1 in opt. soln$\}$

IHS with abstract core reasoning

$\mathcal{F}_{\mathrm{H}} = \{(b_1, b_2), (b_2, b_3), (b_3, b_4),$
$\bigwedge_{i=1}^{2} \mathrm{CNF}\left((b_2 + b_3 \geq i) \rightarrow s^{AB}[i]\right)\}$

$\mathcal{F}_{\mathrm{B}} = \{b_1, b_2, b_3, b_4\}$

$K = \{(s^{AB}[1])\}$

$\tau = \{\neg b_1, b_2, b_3, \neg b_4\}$

$\mathrm{UB} = \mathrm{cost}(\tau)$ $\qquad$ $\mathrm{LB} = 0$

$\mathcal{AB} = \{AB = \{b_2, b_3\}\}$

$C = \emptyset$

$\mathrm{bestsol} = \{\neg b_1, b_2, b_3, \neg b_4\}$

Abstract-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB

Min-Abs ($\mathcal{F}_{\mathrm{B}}, C, \mathcal{AB}$):
minimize: $\sum_{b \in \mathcal{F}_{\mathrm{B}}} b$
subject to: $\sum_{b \in \kappa} b \geq 1 \; \forall \kappa \in C$
$\left(\sum_{b \in AB} b \geq k\right) \leftrightarrow s^{AB}[k] \; \forall AB \in \mathcal{AB}$
return: $\{b \mid b$ set to 1 in opt. soln$\}$

IHS with abstract core reasoning

$\mathcal{F}_H = \{(b_1, b_2), (b_2, b_3), (b_3, b_4),$
$\bigwedge_{i=1}^{2} \text{CNF}\left((b_2 + b_3 \geq i) \to s^{AB}[i]\right)\}$

$\mathcal{F}_B = \{b_1, b_2, b_3, b_4\}$

$K = \{(s^{AB}[1])\}$

$\tau = \{\neg b_1, b_2, b_3, \neg b_4\}$

$UB = 2$ $\quad$ $LB = 0$

$\mathcal{AB} = \{AB = \{b_2, b_3\}\}$

$C = \{(s^{AB}[1])\}$

$\text{bestsol} = \{\neg b_1, b_2, b_3, \neg b_4\}$

Abstract-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

Min-Abs ($\mathcal{F}_B, C, \mathcal{AB}$):

minimize: $\sum_{b \in \mathcal{F}_B} b$
subject to: $\sum_{b \in \kappa} b \geq 1 \; \forall \kappa \in C$
$\left(\sum_{b \in AB} b \geq k\right) \leftrightarrow s^{AB}[k] \; \forall AB \in \mathcal{AB}$
return: $\{b \mid b \text{ set to 1 in opt. soln}\}$

IHS with abstract core reasoning

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4),$
$\bigwedge_{\mathrm{i}=1}^{2} \mathrm{CNF}\left((\mathrm{b}_2 + \mathrm{b}_3 \geq \mathrm{i}) \rightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{i}]\right)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

Abstract-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

$\mathrm{UB} = 2$ $\mathrm{LB} = 0$

$\mathcal{AB} = \{\mathrm{AB} = \{\mathrm{b}_2, \mathrm{b}_3\}\}$

$\mathrm{C} = \{(\mathrm{s}^{\mathrm{AB}}[1])\}$

$\mathrm{bestsol} = \{\neg \mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \neg \mathrm{b}_4\}$

Min-Abs ($\mathcal{F}_{\mathrm{B}}, \mathrm{C}, \mathcal{AB}$):
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$
$\left(\sum_{\mathrm{b} \in \mathrm{AB}} \mathrm{b} \geq \mathrm{k}\right) \leftrightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{k}] \ \forall \mathrm{AB} \in \mathcal{AB}$
return: $\{\mathrm{b} \mid \mathrm{b} \text{ set to 1 in opt. soln}\}$

IHS with abstract core reasoning

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4),$
$\bigwedge_{\mathrm{i}=1}^{2} \mathrm{CNF}\left((\mathrm{b}_2 + \mathrm{b}_3 \geq \mathrm{i}) \rightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{i}]\right)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

Abstract-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

$\mathrm{UB} = 2$ $\mathrm{LB} = 0$

$\mathcal{AB} = \{\mathrm{AB} = \{\mathrm{b}_2, \mathrm{b}_3\}\}$

$\mathrm{C} = \{(\mathrm{s}^{\mathrm{AB}}[1])\}$

$\mathrm{bestsol} = \{\neg \mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \neg \mathrm{b}_4\}$

Min-Abs ($\mathcal{F}_{\mathrm{B}}, \mathrm{C}, \mathcal{AB}$):
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$
$\left(\sum_{\mathrm{b} \in \mathrm{AB}} \mathrm{b} \geq \mathrm{k}\right) \leftrightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{k}] \ \forall \mathrm{AB} \in \mathcal{AB}$
return: $\{\mathrm{b} \mid \mathrm{b}$ set to 1 in opt. soln$\}$

IHS with abstract core reasoning

$\mathcal{F}_H = \{(b_1, b_2), (b_2, b_3), (b_3, b_4),$
$\bigwedge_{i=1}^{2} \mathrm{CNF}\left((b_2 + b_3 \geq i) \rightarrow s^{AB}[i]\right)\}$

$\mathcal{F}_B = \{b_1, b_2, b_3, b_4\}$

$hs = \text{Min-Abs}\left(\mathcal{F}_B, \{(s^{AB}[1])\}, \mathcal{AB}\right)$

Abstract-IHS $(\mathcal{F})$

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

$UB = 2$ $LB = 0$

$\mathcal{AB} = \{AB = \{b_2, b_3\}\}$

$C = \{(s^{AB}[1])\}$

$\text{bestsol} = \{\neg b_1, b_2, b_3, \neg b_4\}$

Min-Abs $(\mathcal{F}_B, C, \mathcal{AB})$:
minimize: $\sum_{b \in \mathcal{F}_B} b$
subject to: $\sum_{b \in \kappa} b \geq 1 \ \forall \kappa \in C$
$\left(\sum_{b \in AB} b \geq k\right) \leftrightarrow s^{AB}[k] \ \forall AB \in \mathcal{AB}$
return: $\{b \mid b \text{ set to 1 in opt. soln}\}$

IHS with abstract core reasoning

$\mathcal{F}_H = \{(b_1, b_2), (b_2, b_3), (b_3, b_4),$
$\bigwedge_{i=1}^{2} \mathrm{CNF}\left((b_2 + b_3 \geq i) \rightarrow s^{AB}[i]\right)\}$

$\mathcal{F}_B = \{b_1, b_2, b_3, b_4\}$

$hs = \{b_2\}$

Abstract-IHS $(\mathcal{F})$

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

$UB = 2$ $LB = |\{b_2\}|$

$\mathcal{AB} = \{AB = \{b_2, b_3\}\}$

$C = \{(s^{AB}[1])\}$

$bestsol = \{\neg b_1, b_2, b_3, \neg b_4\}$

Min-Abs $(\mathcal{F}_B, C, \mathcal{AB})$:
minimize: $\sum_{b \in \mathcal{F}_B} b$
subject to: $\sum_{b \in \kappa} b \geq 1 \; \forall \kappa \in C$
$\left(\sum_{b \in AB} b \geq k\right) \leftrightarrow s^{AB}[k] \; \forall AB \in \mathcal{AB}$
return: $\{b \mid b$ set to 1 in opt. soln$\}$

IHS with abstract core reasoning

$\mathcal{F}_\mathrm{H} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4),$
$\bigwedge_{\mathrm{i}=1}^{2} \mathrm{CNF}\left((\mathrm{b}_2 + \mathrm{b}_3 \geq \mathrm{i}) \rightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{i}]\right)\}$

$\mathcal{F}_\mathrm{B} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{hs} = \{\mathrm{b}_2\}$

$\mathcal{A} = \{\mathrm{b}_1, \mathrm{s}^{\mathrm{AB}}[2], \mathrm{b}_4\}$

Abstract-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

ABSTRACT($\mathcal{F}_\mathrm{B}$, hs, $\mathcal{AB}$)
  $\mathcal{A} \leftarrow \{\mathrm{b} \mid \mathrm{b} \in \mathcal{F}_\mathrm{B} - \mathrm{hs}\}$
  foreach AB $\in \mathcal{AB}$ do
    $\mathcal{A} \leftarrow \mathcal{A} - \{\mathrm{b} \mid \mathrm{b} \in \mathrm{AB}\}$
    $\mathcal{A} \leftarrow \mathcal{A} \cup \{\mathrm{s}^{\mathrm{AB}}[|\mathrm{AB} \cap \mathrm{hs}| + 1]\}$
  return $\mathcal{A}$

IHS with abstract core reasoning

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4),$
$\bigwedge_{\mathrm{i}=1}^{2} \mathrm{CNF}\left((\mathrm{b}_2 + \mathrm{b}_3 \geq \mathrm{i}) \rightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{i}]\right)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

sat-assume$(\mathcal{F}_{\mathrm{H}}, \neg\mathcal{A})$

$\mathcal{A} = \{\mathrm{b}_1, \mathrm{s}^{\mathrm{AB}}[2], \mathrm{b}_4\}$

$\mathrm{K} = \{\}$

Abstract-IHS $(\mathcal{F})$

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

$\mathrm{UB} = 2$ $\mathrm{LB} = 1$

$\mathcal{AB} = \{\mathrm{AB} = \{\mathrm{b}_2, \mathrm{b}_3\}\}$

$\mathrm{C} = \{(\mathrm{s}^{\mathrm{AB}}[1])\}$

bestsol $= \{\neg\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \neg\mathrm{b}_4\}$

Min-Abs $(\mathcal{F}_{\mathrm{B}}, \mathrm{C}, \mathcal{AB})$:
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$
$\left(\sum_{\mathrm{b} \in \mathrm{AB}} \mathrm{b} \geq \mathrm{k}\right) \leftrightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{k}] \ \forall \mathrm{AB} \in \mathcal{AB}$
return: $\{\mathrm{b} \mid \mathrm{b}$ set to 1 in opt. soln$\}$

IHS with abstract core reasoning

$\mathcal{F}_H = \{(b_1, b_2), (b_2, b_3), (b_3, b_4),$
$\bigwedge_{i=1}^{2} \text{CNF}\left((b_2 + b_3 \geq i) \rightarrow s^{AB}[i]\right)\}$

$\mathcal{F}_B = \{b_1, b_2, b_3, b_4\}$

sat-assume$(\mathcal{F}_H, \neg\mathcal{A})$

$\mathcal{A} = \{\cancel{b_1}, \cancel{s^{AB}[2]}, \cancel{b_4}\}$

$K = \{(b_1, s^{AB}[2], b_4)\}$

$UB = 2$ $LB = 1$

$\mathcal{AB} = \{AB = \{b_2, b_3\}\}$

$C = \{(s^{AB}[1])\}$

bestsol $= \{\neg b_1, b_2, b_3, \neg b_4\}$

Abstract-IHS $(\mathcal{F})$

Initialize
while LB < UB
- Update $\mathcal{AB}$
- Compute min-cost hitting set hs
- Update LB
- Set up assumptions
- Extract cores until SAT
- Update UB
- Add cores to C

Min-Abs $(\mathcal{F}_B, C, \mathcal{AB})$:
minimize: $\sum_{b \in \mathcal{F}_B} b$
subject to: $\sum_{b \in \kappa} b \geq 1 \; \forall \kappa \in C$
$\left(\sum_{b \in AB} b \geq k\right) \leftrightarrow s^{AB}[k] \; \forall AB \in \mathcal{AB}$
return: $\{b \mid b \text{ set to 1 in opt. soln}\}$

IHS with abstract core reasoning

$\mathcal{F}_H = \{(b_1, b_2), (b_2, b_3), (b_3, b_4),$
$\bigwedge_{i=1}^{2} \text{CNF}\left((b_2 + b_3 \geq i) \to s^{AB}[i]\right)\}$

$\mathcal{F}_B = \{b_1, b_2, b_3, b_4\}$

$K = \{(b_1, s^{AB}[2], b_4)\}$

$\tau = \{\neg b_1, b_2, b_3, \neg b_4\}$

$UB = 2 \qquad LB = 1$

$\mathcal{AB} = \{AB = \{b_2, b_3\}\}$

$C = \{(s^{AB}[1])\}$

$\text{bestsol} = \{\neg b_1, b_2, b_3, \neg b_4\}$

Abstract-IHS ($\mathcal{F}$)

Initialize
while LB < UB
- Update $\mathcal{AB}$
- Compute min-cost hitting set hs
- Update LB
- Set up assumptions
- Extract cores until SAT
- Update UB
- Add cores to C

Min-Abs ($\mathcal{F}_B$, C, $\mathcal{AB}$):

minimize: $\sum_{b \in \mathcal{F}_B} b$

subject to: $\sum_{b \in \kappa} b \geq 1 \; \forall \kappa \in C$

$\left(\sum_{b \in AB} b \geq k\right) \leftrightarrow s^{AB}[k] \; \forall AB \in \mathcal{AB}$

return: $\{b \mid b \text{ set to 1 in opt. soln}\}$

IHS with abstract core reasoning

$\mathcal{F}_H = \{(b_1, b_2), (b_2, b_3), (b_3, b_4),$
$\bigwedge_{i=1}^{2} \text{CNF}\left((b_2 + b_3 \geq i) \to s^{AB}[i]\right)\}$

$\mathcal{F}_B = \{b_1, b_2, b_3, b_4\}$

$K = \{(b_1, s^{AB}[2], b_4)\}$

$\tau = \{\neg b_1, b_2, b_3, \neg b_4\}$

$UB = 2$ $LB = 1$

$\mathcal{AB} = \{AB = \{b_2, b_3\}\}$

$C = \{(s^{AB}[1]), (b_1, s^{AB}[2], b_4)\}$

$\text{bestsol} = \{\neg b_1, b_2, b_3, \neg b_4\}$

Abstract-IHS ($\mathcal{F}$)

Initialize
while LB < UB
- Update $\mathcal{AB}$
- Compute min-cost hitting set hs
- Update LB
- Set up assumptions
- Extract cores until SAT
- Update UB
- Add cores to C

Min-Abs ($\mathcal{F}_B, C, \mathcal{AB}$):

minimize: $\sum_{b \in \mathcal{F}_B} b$

subject to: $\sum_{b \in \kappa} b \geq 1 \ \forall \kappa \in C$

$\left(\sum_{b \in AB} b \geq k\right) \leftrightarrow s^{AB}[k] \ \forall AB \in \mathcal{AB}$

return: $\{b \mid b \text{ set to 1 in opt. soln}\}$

IHS with abstract core reasoning

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4),$
$\bigwedge_{\mathrm{i}=1}^{2} \mathrm{CNF}\left((\mathrm{b}_2 + \mathrm{b}_3 \geq \mathrm{i}) \rightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{i}]\right)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

Abstract-IHS $(\mathcal{F})$

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

$\mathrm{UB} = 2$ $\mathrm{LB} = 1$

$\mathcal{AB} = \{\mathrm{AB} = \{\mathrm{b}_2, \mathrm{b}_3\}\}$

$\mathrm{C} = \{(\mathrm{s}^{\mathrm{AB}}[1]), (\mathrm{b}_1, \mathrm{s}^{\mathrm{AB}}[2], \mathrm{b}_4)\}$

$\mathrm{bestsol} = \{\neg \mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \neg \mathrm{b}_4\}$

Min-Abs $(\mathcal{F}_{\mathrm{B}}, \mathrm{C}, \mathcal{AB})$:
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$
$\left(\sum_{\mathrm{b} \in \mathrm{AB}} \mathrm{b} \geq \mathrm{k}\right) \leftrightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{k}] \ \forall \mathrm{AB} \in \mathcal{AB}$
return: $\{\mathrm{b} \mid \mathrm{b} \text{ set to 1 in opt. soln}\}$

IHS with abstract core reasoning

$\mathcal{F}_\mathrm{H} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4),$
$\bigwedge_{\mathrm{i}=1}^{2} \mathrm{CNF}\left((\mathrm{b}_2 + \mathrm{b}_3 \geq \mathrm{i}) \rightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{i}]\right)\}$

$\mathcal{F}_\mathrm{B} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{hs} = \mathrm{Min\text{-}Abs}\left(\mathcal{F}_\mathrm{B}, \mathrm{C}, \mathcal{AB}\right)$

Abstract-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

$\mathrm{UB} = 2$ $\mathrm{LB} = 1$

$\mathcal{AB} = \{\mathrm{AB} = \{\mathrm{b}_2, \mathrm{b}_3\}\}$

$\mathrm{C} = \{(\mathrm{s}^{\mathrm{AB}}[1]), (\mathrm{b}_1, \mathrm{s}^{\mathrm{AB}}[2], \mathrm{b}_4)\}$

$\mathrm{bestsol} = \{\neg\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \neg\mathrm{b}_4\}$

Min-Abs $(\mathcal{F}_\mathrm{B}, \mathrm{C}, \mathcal{AB})$:
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_\mathrm{B}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$
$\left(\sum_{\mathrm{b} \in \mathrm{AB}} \mathrm{b} \geq \mathrm{k}\right) \leftrightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{k}] \ \forall \mathrm{AB} \in \mathcal{AB}$
return: $\{\mathrm{b} \mid \mathrm{b} \text{ set to 1 in opt. soln}\}$

IHS with abstract core reasoning

$\mathcal{F}_H = \{(b_1, b_2), (b_2, b_3), (b_3, b_4),$
$\bigwedge_{i=1}^{2} \mathrm{CNF}\left((b_2 + b_3 \geq i) \rightarrow s^{AB}[i]\right)\}$

$\mathcal{F}_B = \{b_1, b_2, b_3, b_4\}$

$hs = \{b_2, b_3\}$

Abstract-IHS ($\mathcal{F}$)

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C

$UB = 2$ $LB = |\{b_2, b_3\}|$

$\mathcal{AB} = \{AB = \{b_2, b_3\}\}$

$C = \{(s^{AB}[1]), (b_1, s^{AB}[2], b_4)\}$

$\mathrm{bestsol} = \{\neg b_1, b_2, b_3, \neg b_4\}$

Min-Abs ($\mathcal{F}_B, C, \mathcal{AB}$):
minimize: $\sum_{b \in \mathcal{F}_B} b$
subject to: $\sum_{b \in \kappa} b \geq 1 \ \forall \kappa \in C$
$\left(\sum_{b \in AB} b \geq k\right) \leftrightarrow s^{AB}[k] \ \forall AB \in \mathcal{AB}$
return: $\{b \mid b \text{ set to 1 in opt. soln}\}$

IHS with abstract core reasoning

$\mathcal{F}_{\mathrm{H}} = \{(\mathrm{b}_1, \mathrm{b}_2), (\mathrm{b}_2, \mathrm{b}_3), (\mathrm{b}_3, \mathrm{b}_4),$
$\bigwedge_{\mathrm{i}=1}^{2} \mathrm{CNF}\left((\mathrm{b}_2 + \mathrm{b}_3 \geq \mathrm{i}) \rightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{i}]\right)\}$

$\mathcal{F}_{\mathrm{B}} = \{\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \mathrm{b}_4\}$

$\mathrm{hs} = \{\mathrm{b}_2, \mathrm{b}_3\}$

Abstract-IHS $(\mathcal{F})$

Initialize
while LB < UB
  Update $\mathcal{AB}$
  Compute min-cost hitting set hs
  Update LB
  Set up assumptions
  Extract cores until SAT
  Update UB
  Add cores to C
return bestsol

$\mathrm{UB} = 2$ $\mathrm{LB} = 2$

$\mathcal{AB} = \{\mathrm{AB} = \{\mathrm{b}_2, \mathrm{b}_3\}\}$

$\mathrm{C} = \{(\mathrm{s}^{\mathrm{AB}}[1]), (\mathrm{b}_1, \mathrm{s}^{\mathrm{AB}}[2], \mathrm{b}_4)\}$

$\mathrm{bestsol} = \{\neg\mathrm{b}_1, \mathrm{b}_2, \mathrm{b}_3, \neg\mathrm{b}_4\}$

Min-Abs $(\mathcal{F}_{\mathrm{B}}, \mathrm{C}, \mathcal{AB})$:
minimize: $\sum_{\mathrm{b} \in \mathcal{F}_{\mathrm{B}}} \mathrm{b}$
subject to: $\sum_{\mathrm{b} \in \kappa} \mathrm{b} \geq 1 \ \forall \kappa \in \mathrm{C}$
$\left(\sum_{\mathrm{b} \in \mathrm{AB}} \mathrm{b} \geq \mathrm{k}\right) \leftrightarrow \mathrm{s}^{\mathrm{AB}}[\mathrm{k}] \ \forall \mathrm{AB} \in \mathcal{AB}$
return: $\{\mathrm{b} \mid \mathrm{b}$ set to 1 in opt. soln$\}$

Effects of Abstract Cores

Abstract cores improve IHS in theory

In theory

For each (unweighted) MaxSAT instance, there exists an abstraction set with which Abstract-IHS terminates with a polynomial number of cores.

Abstract cores improve IHS in theory

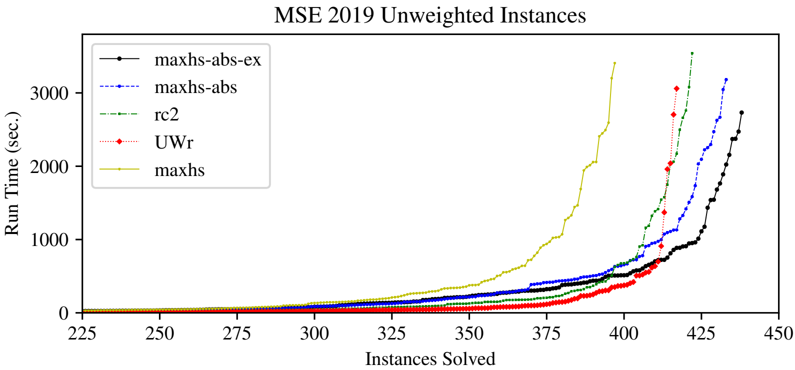
In theory

For each (unweighted) MaxSAT instance, there exists an abstraction set with which Abstract-IHS terminates with a polynomial number of cores.

..however

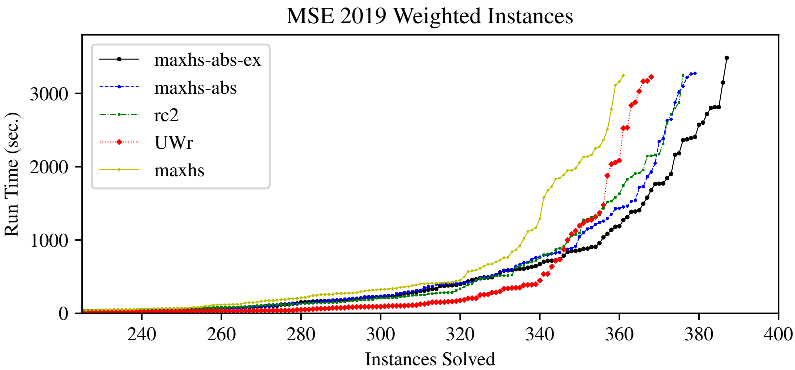
- trade of between expressivity and overhead
- abstraction sets should be large enough to benefit IHS without inducing a lot of overhead.

..and practice

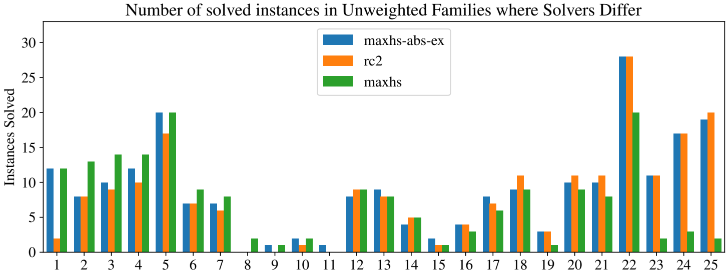


- maxhs: basic IHS (MaxHS [Davies and Bacchus, 2013b, 2011])
- maxhs-abs: maxhs with abstract core reasoning
- maxhs-abs-ex: maxhs-abs with additional heuristics.
- rc2 and UWr best performing solvers in 2019 MSE [Ignatiev, Morgado, and Marques-Silva, 2019; Karpinski and Piotrów, 2019]

Results - Weighted



by benchmark family



Summary

Implicit hitting sets for MaxSat

- MaxSAT - Low-level constraint language:
  weighted Boolean combinations of binary variables
  - Gives tight control over how exactly to encode problem
- Exact optimization: provably optimal solutions
- IHS MaxSat solvers:
  - build on top of highly efficient SAT and IP solver technology
  - one of the most successful approaches to complete MaxSat

Implicit hitting sets for MaxSat

- MaxSAT - Low-level constraint language:
  weighted Boolean combinations of binary variables
  - Gives tight control over how exactly to encode problem
- Exact optimization: provably optimal solutions
- IHS MaxSat solvers:
  - build on top of highly efficient SAT and IP solver technology
  - one of the most successful approaches to complete MaxSat
  - ... even before the addition of abstract cores.

Further Reading and Links

Surveys

- "Maximum Satisfiability" by Bacchus, Järvisalo & Martins
  - Chapter in vol. 2 of Handbook of Satisfiability
  - Now available.

MaxSat Evaluations https://maxsat-evaluations.github.io

Most recent report: [Bacchus, Järvisalo, and Martins, 2019]

Thank you for attending!

Bibliography I

Fahiem Bacchus, Antti Hyttinen, Matti Järvisalo, and Paul Saikko. Reduced cost fixing in MaxSAT. In J. Christopher Beck, editor, Principles and Practice of Constraint Programming - 23rd International Conference, CP 2017, Melbourne, VIC, Australia, August 28 - September 1, 2017, Proceedings, 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. Maxsat evaluation 2018: New developments and detailed results. J. Satisf. Boolean Model. Comput., 11(1):99–131, 2019. doi: 10.3233/SAT190119. URL https://doi.org/10.3233/SAT190119.

Jeremias Berg and Matti Järvisalo. Cost-optimal constrained correlation clustering via weighted partial maximum satisfiability. Artificial Intelligence, 244:110–142, 2017. doi: 10.1016/j.artint.2015.07.001. URL https://doi.org/10.1016/j.artint.2015.07.001.

Jeremias Berg, Matti Järvisalo, and Brandon Malone. Learning optimal bounded treewidth Bayesian networks via maximum satisfiability. In Jukka Corander and Samuel Kaski, editors, Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, April 22-25, 2014, volume 33 of JMLR Workshop and Conference Proceedings, pages 86–95. JMLR.org, 2014.

Jessica Davies. Solving MAXSAT by Decoupling Optimization and Satisfaction. PhD thesis, University of Toronto, 2013.

Jessica Davies and Fahiem Bacchus. Solving MAXSAT by solving a sequence of simpler SAT instances. In Jimmy Ho-Man Lee, editor, Principles and Practice of Constraint Programming - CP 2011 - 17th International Conference, CP 2011, Perugia, Italy, September 12-16, 2011. Proceedings, volume 6876 of Lecture Notes in Computer Science, pages 225–239. Springer, 2011. ISBN 978-3-642-23785-0. doi: 10.1007/978-3-642-23786-7. URL http://dx.doi.org/10.1007/978-3-642-23786-7.

Jessica Davies and Fahiem Bacchus. Postponing optimization to speed up MAXSAT solving. In Christian Schulte, editor, Principles and Practice of Constraint Programming - 19th International Conference, CP 2013, Uppsala, Sweden, September 16-20, 2013. Proceedings, volume 8124 of Lecture Notes in Computer Science, pages 247–262. Springer, 2013a.

Bibliography II

Jessica Davies and Fahiem Bacchus. Exploiting the power of MIP solvers in MaxSAT. In Matti Järvisalo and Allen Van Gelder, editors, Theory and Applications of Satisfiability Testing - SAT 2013 - 16th International Conference, Helsinki, Finland, July 8-12, 2013. Proceedings, volume 7962 of Lecture Notes in Computer Science, pages 166–181. Springer, 2013b.

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.

Alexey Ignatiev, Alessandro Previti, Mark H. Liffiton, and João Marques-Silva. Smallest MUS extraction with minimal hitting set dualization. In Gilles Pesant, editor, Principles and Practice of Constraint Programming - 21st International Conference, CP 2015, Cork, Ireland, August 31 - September 4, 2015, Proceedings, pages 173–182, 2015. doi: 10.1007/978-3-319-23219-5_13. URL https://doi.org/10.1007/978-3-319-23219-5_13.

Alexey Ignatiev, António Morgado, and João Marques-Silva. RC2: an efficient maxsat solver. J. Satisf. Boolean Model. Comput., 11(1):53–64, 2019. URL https://doi.org/10.3233/SAT190116.

Richard M. Karp. Implicit hitting set problems and multi-genome alignment. In Amihood Amir and Laxmi Parida, editors, Combinatorial Pattern Matching, 21st Annual Symposium, CPM 2010, New York, NY, USA, June 21-23, 2010. Proceedings, volume 6129 of Lecture Notes in Computer Science, page 151. Springer, 2010.

Michal Karpinski and Marek Piotrów. Encoding cardinality constraints using multiway merge selection networks. Constraints, 24(3-4):234–251, 2019. URL https://doi.org/10.1007/s10601-019-09302-0.

Paul Saikko, Jeremias Berg, and Matti Järvisalo. LMHS: A SAT-IP hybrid MaxSAT solver. In Nadia Creignou and Daniel Le Berre, editors, Theory and Applications of Satisfiability Testing - SAT 2016 - 19th International Conference, Bordeaux, France, July 5-8, 2016, Proceedings, volume 9710 of Lecture Notes in Computer Science, pages 539–546. Springer, 2016a.

Paul Saikko, Johannes Peter Wallner, and Matti Järvisalo. Implicit hitting set algorithms for reasoning beyond NP. In Chitta Baral, James P. Delgrande, and Frank Wolter, editors, Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016., pages 104–113. AAAI Press, 2016b.