Satisfiability Modulo Theories

Successes and Challenges

Clark Barrett

barrett@cs.nyu.edu

New York University

Motivation

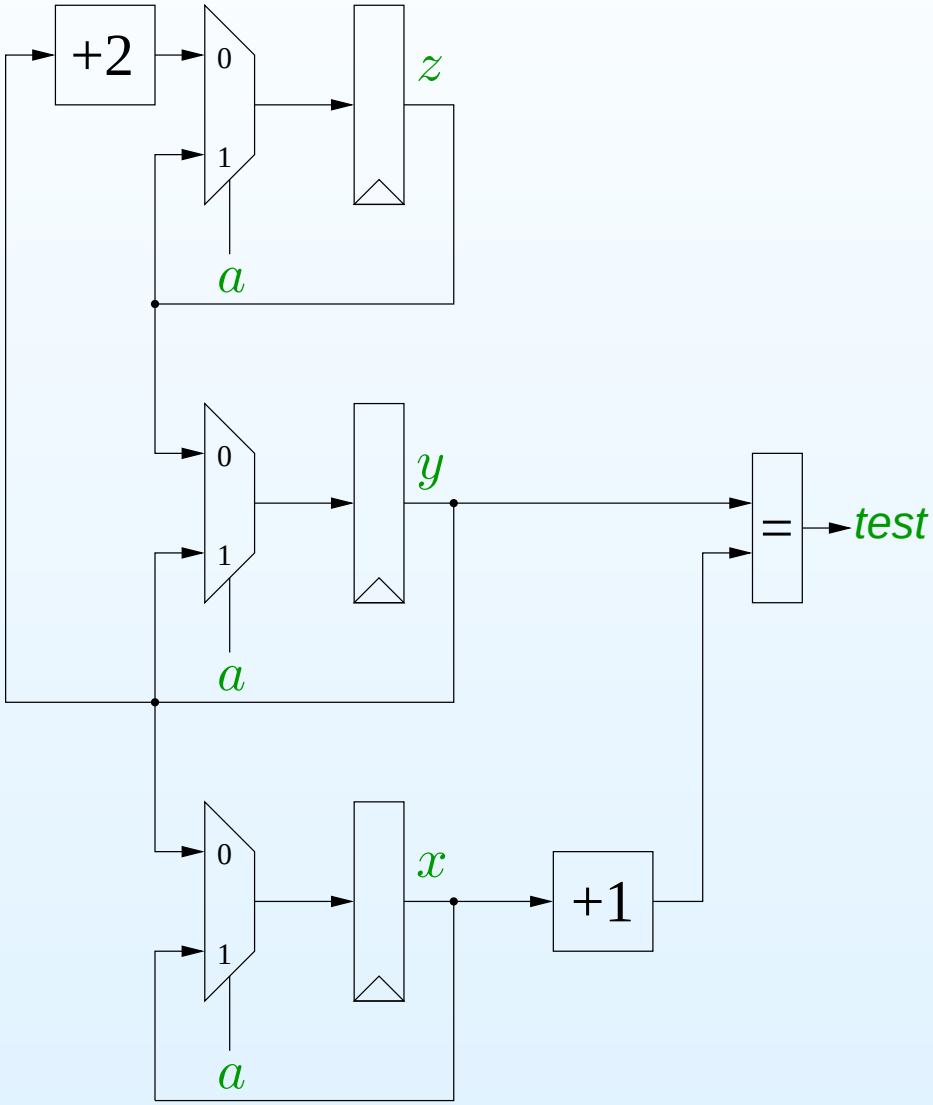
Automatic analysis of computer hardware and software requires engines capable of reasoning efficiently about large and complex systems.

Boolean engines such as Binary Decision Diagrams and SAT solvers are typical engines of choice for today's industrial verification applications.

However, systems are usually designed and modeled at a higher level than the Boolean level and the translation to Boolean logic can be expensive.

A primary goal of research in Satisfiability Modulo Theories (SMT) is to create verification engines that can reason natively at a higher level of abstraction, while still retaining the speed and automation of today's Boolean engines.

Example



Circuit Example

In this example, the value of *test* is always supposed to be *true* .

Circuit Example

In this example, the value of *test* is always supposed to be *true* .

One way to prove this is by induction over the number of clock cycles the circuit has executed.

The inductive step is to prove that if *test* is *true* in the current state, then *test* should be *true* in the next state.

Circuit Example

In this example, the value of *test* is always supposed to be *true* .

One way to prove this is by induction over the number of clock cycles the circuit has executed.

The inductive step is to prove that if *test* is *true* in the current state, then *test* should be *true* in the next state.

Let's compare a proof using SAT to a proof using SMT.

Circuit Example

The example can be modeled intuitively as follows:

```
(y = x + 1 AND z = x + 2 AND
 x′ = IF a THEN x ELSE y AND
 y′ = IF a THEN y ELSE z AND
 z′ = IF a THEN z ELSE y + 2) IMPLIES
 y′ = x′ + 1 AND z′ = x′ + 2
```

We can prove this formula by showing that the negation is unsatisfiable.

We can write this formula in propositional logic by using one propositional variable for each bit in the current and next states.

Circuit Example

Assuming a bit-width of 2 for simplicity and skipping the details, we get the following formula:

$$
\begin{aligned}
&(z1 \leftrightarrow \neg x1) \wedge (z0 \leftrightarrow x0) \wedge \\
&(y1 \leftrightarrow (x1 \oplus x0)) \wedge (y0 \leftrightarrow \neg x0) \wedge \\
&(a \rightarrow ((xp1 \leftrightarrow x1) \wedge (xp0 \leftrightarrow x0))) \wedge \\
&(\neg a \rightarrow ((xp1 \leftrightarrow y1) \wedge (xp0 \leftrightarrow y0))) \wedge \\
&(a \rightarrow ((yp1 \leftrightarrow y1) \wedge (yp0 \leftrightarrow y0))) \wedge \\
&(\neg a \rightarrow ((yp1 \leftrightarrow z1) \wedge (yp0 \leftrightarrow z0))) \wedge \\
&(a \rightarrow ((zp1 \leftrightarrow z1) \wedge (zp0 \leftrightarrow z0))) \wedge \\
&(\neg a \rightarrow ((zp1 \leftrightarrow \neg y1) \wedge (zp0 \leftrightarrow y0))) \wedge \\
&(\neg(zp1 \leftrightarrow \neg xp1) \vee \neg(zp0 \leftrightarrow xp0) \vee \\
&\neg(yp1 \leftrightarrow (xp1 \oplus xp0)) \wedge (yp0 \leftrightarrow \neg xp0)
\end{aligned}
$$

Circuit Example

Recall that the invariant of the circuit is captured by the following formula:

```
(y = x + 1 AND z = x + 2 AND
 x′ = IF a THEN x ELSE y AND
 y′ = IF a THEN y ELSE z AND
 z′ = IF a THEN z ELSE y + 2) IMPLIES
 y′ = x′ + 1 AND z′ = x′ + 2
```

When using a SAT solver, this formula must be encoded into propositional logic

Using an SMT solver, the formula can be solved as it is

SMT vs First-Order Satisfiability

It is important to make a distinction between SMT and first-order satisfiability. For example, is the following sentence satisfiable?

$$\mathit{read}\left(\mathit{write}\left(a, i, v\right), i\right) \neq v$$

SMT vs First-Order Satisfiability

It is important to make a distinction between SMT and first-order satisfiability. For example, is the following sentence satisfiable?

$$\mathit{read}\left(\mathit{write}\left(a, i, v\right), i\right) \neq v$$

If the set of allowable models is unrestricted, then the answer is yes.

SMT vs First-Order Satisfiability

It is important to make a distinction between SMT and first-order satisfiability. For example, is the following sentence satisfiable?

$$\mathit{read}\,(\mathit{write}\,(a, i, v), i) \neq v$$

If the set of allowable models is unrestricted, then the answer is yes.

However, if we only consider models that obey the axioms for *read* and *write* then the answer is no.

Satisfiability Modulo Theories

For a theory T, the T-satisfiability problem consists of deciding whether there exists a model $\mathcal{A}$ and variable assignment α such that $(\mathcal{A}, \alpha) \models T \cup \varphi$ for an given formula φ.

Another way to think of this is as a restriction on the models we are willing to consider when trying to satisfy φ.

Some recent work in SMT considers a theory to be a collection of models rather than a set of sentences.

Theories

In principle, SMT can be applied to any theory T.

In practice, SMT solvers support a small but growing set of theories.

Common Theories for SMT:

- Empty theory (equality with uninterpreted functions)
- Real and/or integer arithmetic
- Arrays
- Inductive (Recursive) Data Types
- Records and Tuples
- Bit-vectors
- Pointers and reachability

Language

Given a theory, there is also a question of whether we are checking satisfiability of the full first-order language or some subset of it.

A typical restriction is that the formula be quantifier-free.

For arithmetic, other common restrictions include:

- Formulas with only linear atoms
- Formulas with only difference logic atoms (i.e. $x - y \bowtie c$, where x and y are variables, c is a constant, and $\bowtie \in \{<, \leq, >, \geq, =\}$)

Example: Theory $T_{\mathcal{A}}$ of Arrays

Let $\Sigma_{\mathcal{A}}$ be the signature $(\mathit{read}, \mathit{write})$.

Let $\Lambda_{\mathcal{A}}$ be the following axioms:

$$\forall\, a\, \forall\, i\, \forall\, v\, (\mathit{read}\,(\mathit{write}\,(a, i, v), i) = v)$$
$$\forall\, a\, \forall\, i\, \forall\, j\, \forall\, v\, (i \neq j \rightarrow \mathit{read}\,(\mathit{write}\,(a, i, v), j) = \mathit{read}\,(a, j))$$
$$\forall\, a\, \forall\, b\, ((\forall\, i\, (\mathit{read}\,(a, i) = \mathit{read}\,(b, i))) \rightarrow a = b)$$

Then $T_{\mathcal{A}} =$ the set of all consequences of $\Lambda_{\mathcal{A}}$.

The satisfiability problem for $T_{\mathcal{A}}$ is undecidable, but the quantifier-free satisfiability problem for $T_{\mathcal{A}}$ is decidable (the problem is NP-complete).

Combining Theories

We are usually interested in a combination of theories. The standard technique for this is the Nelson-Oppen method.

Suppose that $T_1, \ldots, T_n$ are stably-infinite theories with disjoint signatures $\Sigma_1, \ldots, \Sigma_n$ and Sat_i decides T_i-satisfiability of $\Sigma_i(C)$ literals.

We wish to determine the satisfiability of a ground conjunction Γ of $\Sigma(C)$-literals.

1. Purify Γ to obtain an equisatisfiable set $\bigwedge \varphi_i$, where each φ_i is i-pure.
2. Let S be the set of shared variables (i.e. appearing in more than one φ_i).
3. For each arrangement Δ of S,

   Check $\mathit{Sat}_i(\varphi_i \wedge \Delta)$ for each i.

Combining SAT and SMT

Theory solvers check the satisfiability of conjunctions of literals.

What about more general Boolean combinations of literals?

What is needed is a combination of SAT reasoning and theory reasoning.

The so-called eager approach to SMT tries to find ways of encoding everything into SAT. There are a variety of techniques, and for some theories, this works quite well.

Here, I will focus on the lazy combination of SAT and theory reasoning. The lazy approach is the basis for most modern SMT solvers.

Abstract DPLL

We start with an abstract description of DPLL, the algorithm used by most SAT solvers.

Abstract DPLL

We start with an abstract description of DPLL, the algorithm used by most SAT solvers.

- Abstract DPLL uses states and transitions to model the progress of the algorithm.

Abstract DPLL

We start with an abstract description of DPLL, the algorithm used by most SAT solvers.

- Abstract DPLL uses states and transitions to model the progress of the algorithm.
- Most states are of the form $M \parallel F$, where
  - M is a sequence of annotated literals denoting a partial truth assignment, and
  - F is the CNF formula being checked, represented as a set of clauses.

Abstract DPLL

We start with an abstract description of DPLL, the algorithm used by most SAT solvers.

- Abstract DPLL uses states and transitions to model the progress of the algorithm.
- Most states are of the form $M \parallel F$, where
  - M is a sequence of annotated literals denoting a partial truth assignment, and
  - F is the CNF formula being checked, represented as a set of clauses.
- The initial state is $\emptyset \parallel F$, where F is to be checked for satisfiability.

Abstract DPLL

We start with an abstract description of DPLL, the algorithm used by most SAT solvers.

- Abstract DPLL uses states and transitions to model the progress of the algorithm.
- Most states are of the form $M \parallel F$, where
  - M is a sequence of annotated literals denoting a partial truth assignment, and
  - F is the CNF formula being checked, represented as a set of clauses.
- The initial state is $\emptyset \parallel F$, where F is to be checked for satisfiability.
- Transitions between states are defined by a set of conditional transition rules.

Abstract DPLL

The final state is either:

- a special fail state: $fail$, if F is unsatisfiable, or
- $M \parallel G$, where G is a CNF formula equisatisfiable with the original formula F, and M satisfies G

We write $M \models C$ to mean that for every truth assignment v, $v(M) = \mathit{true}$ implies $v(C) = \mathit{true}$.

Abstract DPLL Rules

UnitProp :

$$M \parallel F, C \vee l \implies M\ l \parallel F, C \vee l \quad \textbf{if} \begin{cases} M \models \neg C \\ l \text{ is undefined in } M \end{cases}$$

PureLiteral :

$$M \parallel F \implies M\ l \parallel F \quad \textbf{if} \begin{cases} l \text{ occurs in some clause of } F \\ -l \text{ occurs in no clause of } F \\ l \text{ is undefined in } M \end{cases}$$

Decide :

$$M \parallel F \implies M\ l^{\mathsf{d}} \parallel F \quad \textbf{if} \begin{cases} l \text{ or } \neg l \text{ occurs in a clause of } F \\ l \text{ is undefined in } M \end{cases}$$

Fail :

$$M \parallel F, C \implies fail \quad \textbf{if} \begin{cases} M \models \neg C \\ M \text{ contains no decision literals} \end{cases}$$

Abstract DPLL Rules

Backjump :

$$M\ l^{\mathsf{d}}\ N \parallel F, C \implies M\ l' \parallel F, C \quad \textbf{if} \begin{cases} M\ l^{\mathsf{d}}\ N \models \neg C, \text{ and there is} \\ \text{some clause } C' \vee l' \text{ such that:} \\ \quad F, C \models C' \vee l' \text{ and } M \models \neg C', \\ \quad l' \text{ is undefined in } M, \text{ and} \\ \quad l' \text{ or } \neg l' \text{ occurs in } F \text{ or in } M\ l^{\mathsf{d}}\ N \end{cases}$$

Learn :

$$M \parallel F \implies M \parallel F, C \quad \textbf{if} \begin{cases} \text{all atoms of } C \text{ occur in } F \\ F \models C \end{cases}$$

Forget :

$$M \parallel F, C \implies M \parallel F \quad \textbf{if} \begin{cases} F \models C \end{cases}$$

Restart :

$$M \parallel F \implies \emptyset \parallel F$$

Example

$$\emptyset \parallel \quad 1 \vee \overline{2},\ \ \overline{1} \vee \overline{2},\ \ 2 \vee 3,\ \ \overline{3} \vee 2,\ \ 1 \vee 4 \quad \Longrightarrow \quad \text{(PureLiteral)}$$

$$4\ 1^{\mathrm{d}}\ \overline{2}\ 3 \parallel$$

Example

$$\emptyset \parallel \quad 1\vee\overline{2},\ \ \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{3}\vee 2,\ \ 1\vee 4 \quad \Longrightarrow \quad (\text{PureLiteral})$$

$$4 \parallel \quad 1\vee\overline{2},\ \ \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{3}\vee 2,\ \ 1\vee 4$$

$$4\ 1^{\mathsf{d}}\ \overline{2}\ 3 \parallel$$

Example

$$
\begin{array}{rcl}
\emptyset & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \quad \Longrightarrow \quad (\text{PureLiteral}) \\
4 & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \quad \Longrightarrow \quad (\text{Decide}) \\
4\ 1^{\mathsf{d}} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \\
4\ 1^{\mathsf{d}}\ \overline{2}\ 3 & \| &
\end{array}
$$

Example

$$
\begin{array}{rcl}
\emptyset & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \quad \Longrightarrow \quad (\text{PureLiteral}) \\
4 & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \quad \Longrightarrow \quad (\text{Decide}) \\
4\ 1^{\mathsf{d}} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \quad \Longrightarrow \quad (\text{UnitProp}) \\
4\ 1^{\mathsf{d}}\ \overline{2} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \\
4\ 1^{\mathsf{d}}\ \overline{2}\ 3 & \| &
\end{array}
$$

Example

$$
\begin{array}{rcll}
\emptyset & \| & 1\vee\overline{2},\ \ \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{3}\vee 2,\ \ 1\vee 4 & \Longrightarrow \ (\text{PureLiteral}) \\
4 & \| & 1\vee\overline{2},\ \ \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{3}\vee 2,\ \ 1\vee 4 & \Longrightarrow \ (\text{Decide}) \\
4\ 1^{\mathsf{d}} & \| & 1\vee\overline{2},\ \ \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{3}\vee 2,\ \ 1\vee 4 & \Longrightarrow \ (\text{UnitProp}) \\
4\ 1^{\mathsf{d}}\ \overline{2} & \| & 1\vee\overline{2},\ \ \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{3}\vee 2,\ \ 1\vee 4 & \Longrightarrow \ (\text{UnitProp}) \\
4\ 1^{\mathsf{d}}\ \overline{2}\ 3 & \| & 1\vee\overline{2},\ \ \overline{1}\vee\overline{2},\ \ 2\vee 3,\ \ \overline{3}\vee 2,\ \ 1\vee 4 &
\end{array}
$$

Example

$$
\begin{array}{rcl}
\emptyset & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \\
4 & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \\
4\ 1^{\mathsf{d}} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \\
4\ 1^{\mathsf{d}}\ \overline{2} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \\
4\ 1^{\mathsf{d}}\ \overline{2}\ 3 & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \\
4\ \overline{1} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4
\end{array}
\quad
\begin{array}{c}
\Longrightarrow \\ \Longrightarrow \\ \Longrightarrow \\ \Longrightarrow \\ \Longrightarrow \\ \
\end{array}
\quad
\begin{array}{l}
(\text{PureLiteral}) \\ (\text{Decide}) \\ (\text{UnitProp}) \\ (\text{UnitProp}) \\ (\text{Backtrack}) \\ \
\end{array}
$$

Example

$$
\begin{array}{rcl}
\emptyset & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \\
4 & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \\
4\ 1^{\mathsf{d}} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \\
4\ 1^{\mathsf{d}}\ \overline{2} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \\
4\ 1^{\mathsf{d}}\ \overline{2}\ 3 & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \\
4\ \overline{1} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \\
4\ \overline{1}\ \overline{2}\ \overline{3} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4
\end{array}
\begin{array}{cl}
\Longrightarrow & (\text{PureLiteral}) \\
\Longrightarrow & (\text{Decide}) \\
\Longrightarrow & (\text{UnitProp}) \\
\Longrightarrow & (\text{UnitProp}) \\
\Longrightarrow & (\text{Backtrack}) \\
\Longrightarrow & (\text{UnitProp}) \\
 &
\end{array}
$$

Example

$$
\begin{array}{rcl}
\emptyset & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \quad \Longrightarrow \quad (\text{PureLiteral}) \\
4 & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \quad \Longrightarrow \quad (\text{Decide}) \\
4\ 1^{\mathsf{d}} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \quad \Longrightarrow \quad (\text{UnitProp}) \\
4\ 1^{\mathsf{d}}\ \overline{2} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \quad \Longrightarrow \quad (\text{UnitProp}) \\
4\ 1^{\mathsf{d}}\ \overline{2}\ 3 & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \quad \Longrightarrow \quad (\text{Backtrack}) \\
4\ \overline{1} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \quad \Longrightarrow \quad (\text{UnitProp}) \\
4\ \overline{1}\ \overline{2}\ \overline{3} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee 3,\ \overline{3}\vee 2,\ 1\vee 4 \quad \Longrightarrow \quad (\text{Fail}) \\
\multicolumn{3}{c}{\mathit{fail}}
\end{array}
$$

Example

$$
\begin{array}{rcl}
\emptyset & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee3,\ \overline{3}\vee2,\ 1\vee4 \quad \Longrightarrow \quad (\text{PureLiteral}) \\
4 & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee3,\ \overline{3}\vee2,\ 1\vee4 \quad \Longrightarrow \quad (\text{Decide}) \\
4\ 1^{\mathsf{d}} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee3,\ \overline{3}\vee2,\ 1\vee4 \quad \Longrightarrow \quad (\text{UnitProp}) \\
4\ 1^{\mathsf{d}}\ \overline{2} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee3,\ \overline{3}\vee2,\ 1\vee4 \quad \Longrightarrow \quad (\text{UnitProp}) \\
4\ 1^{\mathsf{d}}\ \overline{2}\ 3 & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee3,\ \overline{3}\vee2,\ 1\vee4 \quad \Longrightarrow \quad (\text{Backtrack}) \\
4\ \overline{1} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee3,\ \overline{3}\vee2,\ 1\vee4 \quad \Longrightarrow \quad (\text{UnitProp}) \\
4\ \overline{1}\ \overline{2}\ \overline{3} & \| & 1\vee\overline{2},\ \overline{1}\vee\overline{2},\ 2\vee3,\ \overline{3}\vee2,\ 1\vee4 \quad \Longrightarrow \quad (\text{Fail}) \\
\multicolumn{3}{c}{\mathit{fail}}
\end{array}
$$

Result: Unsatisfiable

Abstract DPLL Modulo Theories

The Abstract DPLL Modulo Theories framework extends the Abstract DPLL framework, providing an abstract and formal setting for reasoning about the combination of SAT and theory reasoning.

Assume we have a theory T with signature Σ and a solver Sat_T that can check T-satisfiability of conjunctions of Σ-literals.

Suppose we want to check the satisfiability of an arbitray (quantifier-free) Σ-formula ϕ.

We start by converting ϕ to CNF.

We can then use the Abstract DPLL rules, allowing any first-order literal where before we had propositional literals.

Abstract DPLL Modulo Theories

The Abstract DPLL Modulo Theories framework extends the Abstract DPLL framework, providing an abstract and formal setting for reasoning about the combination of SAT and theory reasoning.

Assume we have a theory T with signature Σ and a solver Sat_T that can check T-satisfiability of conjunctions of Σ-literals.

Suppose we want to check the satisfiability of an arbitray (quantifier-free) Σ-formula ϕ.

We start by converting ϕ to CNF.

What other changes do we need to make to Abstract DPLL so it will work for SMT?

Abstract DPLL Modulo Theories

The first change is to the definition of a final state. A final state is now:

- the special fail state: $fail$, or
- $M \parallel F$, where $M \models F$, and $\mathit{Sat}_T(M)$ reports satisfiable.

Abstract DPLL Modulo Theories

The first change is to the definition of a final state. A final state is now:

- the special fail state: $fail$, or
- $M \parallel F$, where $M \models F$, and $\mathit{Sat}_T(M)$ reports satisfiable.

What happens if we reach a state in which: $M \parallel F$, $M \models F$, and $\mathit{Sat}_T(M)$ reports unsatisfiable? (call this a pseudo-final state)

Abstract DPLL Modulo Theories

The first change is to the definition of a final state. A final state is now:

- the special fail state: $fail$, or
- $M \parallel F$, where $M \models F$, and $\mathit{Sat}_T(M)$ reports satisfiable.

What happens if we reach a state in which: $M \parallel F$, $M \models F$, and $\mathit{Sat}_T(M)$ reports unsatisfiable? (call this a pseudo-final state)

We need to backtrack. The SAT solver will take care of this automatically if we can add a clause C such that $M \models \neg C$.

Abstract DPLL Modulo Theories

The first change is to the definition of a final state. A final state is now:

- the special fail state: $fail$, or
- $M \parallel F$, where $M \models F$, and $\mathit{Sat}_T(M)$ reports satisfiable.

What happens if we reach a state in which: $M \parallel F$, $M \models F$, and $\mathit{Sat}_T(M)$ reports unsatisfiable? (call this a pseudo-final state)

We need to backtrack. The SAT solver will take care of this automatically if we can add a clause C such that $M \models \neg C$.

What clause should we add?

Abstract DPLL Modulo Theories

The first change is to the definition of a final state. A final state is now:

- the special fail state: $fail$, or
- $M \parallel F$, where $M \models F$, and $\mathit{Sat}_T(M)$ reports satisfiable.

What happens if we reach a state in which: $M \parallel F$, $M \models F$, and $\mathit{Sat}_T(M)$ reports unsatisfiable? (call this a pseudo-final state)

We need to backtrack. The SAT solver will take care of this automatically if we can add a clause C such that $M \models \neg C$.

What clause should we add? How about $\neg M$?

Abstract DPLL Modulo Theories

The justification for adding $\neg M$ is that $T \models \neg M$.

We can generalize this to any clause C such that $T \models C$. The following modified Learn rule allows this (we also modify the Forget rule in an analagous way):

Theory Learn :

$$M \parallel F \quad \Longrightarrow \quad M \parallel F, C \quad \textbf{if} \begin{cases} \text{all atoms of } C \text{ occur in } F \\ F \models_T C \end{cases}$$

Theory Forget :

$$M \parallel F, C \quad \Longrightarrow \quad M \parallel F \quad \textbf{if} \begin{cases} F \models_T C \end{cases}$$

Abstract DPLL Modulo Theories

The resulting set of rules is almost enough to correctly implement an SMT solver. We need one more change.

Abstract DPLL Modulo Theories

The resulting set of rules is almost enough to correctly implement an SMT solver. We need one more change.

A somewhat surprising observation is that the pure literal rule has to be abandoned. Why?

Abstract DPLL Modulo Theories

The resulting set of rules is almost enough to correctly implement an SMT solver. We need one more change.

A somewhat surprising observation is that the pure literal rule has to be abandoned. Why?

Propositional literals are independent of each other, but first order literals may not be.

Abstract DPLL Modulo Theories

The resulting set of rules is almost enough to correctly implement an SMT solver. We need one more change.

A somewhat surprising observation is that the pure literal rule has to be abandoned. Why?

Propositional literals are independent of each other, but first order literals may not be.

The remaining rules form a sound and complete procedure for SMT.

From SAT to SMT

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\emptyset \parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$$

From SAT to SMT

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\emptyset \parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} \qquad \Longrightarrow \quad (\text{UnitProp})$$

$$1 \parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$$

From SAT to SMT

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \quad \vee \quad \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \quad \vee \quad \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll} \emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{UnitProp}) \\ 1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{Decide}) \\ 1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \end{array}$$

From SAT to SMT

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \; \vee \; \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \; \vee \; \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} &
\end{array}$$

From SAT to SMT

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \quad \vee \quad \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \quad \vee \quad \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Theory Learn)} \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4 &
\end{array}$$

From SAT to SMT

$$\underbrace{g(a) = c}_{1} \;\wedge\; \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \;\vee\; \underbrace{g(a) = d}_{3} \;\wedge\; \underbrace{c \neq d}_{\overline{4}} \;\vee\; \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Theory Learn)} \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4 & \Longrightarrow \quad \text{(Backjump)} \\
1\ \overline{2}^{\mathsf{d}}\ 4 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4 &
\end{array}$$

From SAT to SMT

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \quad \vee \quad \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \quad \vee \quad \underbrace{g(a) \neq d}_{\overline{3}}$$

$\emptyset$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (UnitProp)
1	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Decide)
$1\ \overline{2}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Decide)
$1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Theory Learn)
$1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4$	$\Longrightarrow$ (Backjump)
$1\ \overline{2}^{\mathsf{d}}\ 4$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4$	$\Longrightarrow$ (UnitProp)
$1\ \overline{2}^{\mathsf{d}}\ 4\ \overline{3}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4$	

From SAT to SMT

$$\underbrace{g(a) = c}_{1} \;\wedge\; \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \;\vee\; \underbrace{g(a) = d}_{3} \;\wedge\; \underbrace{c \neq d}_{\overline{4}} \;\vee\; \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{UnitProp}) \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{Decide}) \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{Decide}) \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{Theory Learn}) \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4 & \Longrightarrow \quad (\text{Backjump}) \\
1\ \overline{2}^{\mathsf{d}}\ 4 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4 & \Longrightarrow \quad (\text{UnitProp}) \\
1\ \overline{2}^{\mathsf{d}}\ 4\ \overline{3} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4 & \Longrightarrow \quad (\text{Theory Learn}) \\
1\ \overline{2}^{\mathsf{d}}\ 4\ \overline{3} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3 &
\end{array}$$

From SAT to SMT

$$\underbrace{g(a) = c}_{1} \;\wedge\; \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \;\vee\; \underbrace{g(a) = d}_{3} \;\wedge\; \underbrace{c \neq d}_{\overline{4}} \;\vee\; \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Theory Learn)} \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4 & \Longrightarrow \quad \text{(Backjump)} \\
1\ \overline{2}^{\mathsf{d}}\ 4 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4 & \Longrightarrow \quad \text{(UnitProp)} \\
1\ \overline{2}^{\mathsf{d}}\ 4\ \overline{3} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4 & \Longrightarrow \quad \text{(Theory Learn)} \\
1\ \overline{2}^{\mathsf{d}}\ 4\ \overline{3} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3 & \Longrightarrow \quad \text{(Backjump)} \\
1\ 2 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3 &
\end{array}$$

From SAT to SMT

$$\underbrace{g(a) = c}_{1} \;\wedge\; \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \;\vee\; \underbrace{g(a) = d}_{3} \;\wedge\; \underbrace{c \neq d}_{\overline{4}} \;\vee\; \underbrace{g(a) \neq d}_{\overline{3}}$$

$\emptyset$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (UnitProp)
1	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Decide)
$1\ \overline{2}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Decide)
$1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Theory Learn)
$1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4$	$\Longrightarrow$ (Backjump)
$1\ \overline{2}^{\mathsf{d}}\ 4$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4$	$\Longrightarrow$ (UnitProp)
$1\ \overline{2}^{\mathsf{d}}\ 4\ \overline{3}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4$	$\Longrightarrow$ (Theory Learn)
$1\ \overline{2}^{\mathsf{d}}\ 4\ \overline{3}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3$	$\Longrightarrow$ (Backjump)
$1\ 2$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3$	$\Longrightarrow$ (UnitProp)
$1\ 2\ 3\ \overline{4}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3$	

From SAT to SMT

$$\underbrace{g(a) = c}_{1} \;\wedge\; \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \;\vee\; \underbrace{g(a) = d}_{3} \;\wedge\; \underbrace{c \neq d}_{\overline{4}} \;\vee\; \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Theory Learn)} \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4 & \Longrightarrow \quad \text{(Backjump)} \\
1\ \overline{2}^{\mathsf{d}}\ 4 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4 & \Longrightarrow \quad \text{(UnitProp)} \\
1\ \overline{2}^{\mathsf{d}}\ 4\ \overline{3} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4 & \Longrightarrow \quad \text{(Theory Learn)} \\
1\ \overline{2}^{\mathsf{d}}\ 4\ \overline{3} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3 & \Longrightarrow \quad \text{(Backjump)} \\
1\ 2 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3 & \Longrightarrow \quad \text{(UnitProp)} \\
1\ 2\ 3\ \overline{4} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3 & \Longrightarrow \quad \text{(Theory Learn)} \\
1\ 2\ 3\ \overline{4} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3,\ \overline{1} \vee \overline{2} \vee \overline{3} \vee 4 &
\end{array}$$

From SAT to SMT

$$\underbrace{g(a) = c}_{1} \;\wedge\; \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \;\vee\; \underbrace{g(a) = d}_{3} \;\wedge\; \underbrace{c \neq d}_{\overline{4}} \;\vee\; \underbrace{g(a) \neq d}_{\overline{3}}$$

$\emptyset$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (UnitProp)
1	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Decide)
$1\ \overline{2}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Decide)
$1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Theory Learn)
$1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4$	$\Longrightarrow$ (Backjump)
$1\ \overline{2}^{\mathsf{d}}\ 4$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4$	$\Longrightarrow$ (UnitProp)
$1\ \overline{2}^{\mathsf{d}}\ 4\ \overline{3}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4$	$\Longrightarrow$ (Theory Learn)
$1\ \overline{2}^{\mathsf{d}}\ 4\ \overline{3}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3$	$\Longrightarrow$ (Backjump)
$1\ 2$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3$	$\Longrightarrow$ (UnitProp)
$1\ 2\ 3\ \overline{4}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3$	$\Longrightarrow$ (Theory Learn)
$1\ 2\ 3\ \overline{4}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 \vee 4,\ \overline{1} \vee 2 \vee \overline{4} \vee 3,\ \overline{1} \vee \overline{2} \vee \overline{3} \vee 4$	$\Longrightarrow$ (Fail)
$fail$			

Improving Abstract DPLL Modulo Theories

We will mention three ways to improve the algorithm.

- Minimizing learned clauses
- Eager conflict detection
- Theory propagation

Minimizing Learned Clauses

The main difficulty with the approach as it stands is that learned clauses can be highly redundant.

Suppose that F contains $n + 2$ propositional variables.

When a pseudo-final state is reached, M will determine a value for all $n + 2$ variables.

But what if only 2 of these assignments are already T-unsatisfiable?

If we always learn $\neg M$ in a pseudo-final state, in the worst case, 2^n clauses will be need to be learned when a single clause containing the two offending literals would have sufficed.

Minimizing Learned Clauses

To avoid this kind of redundancy, we can be smarter about the clauses that are learned with Theory Learn.

In particular, when $\mathit{Sat}_T(M)$ is called, we should make an effort to find the smallest possible subset of M which is inconsistent.

We can then learn a clause derived from only these literals.

From SAT to SMT — Minimized Clauses

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\emptyset \parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$$

From SAT to SMT — Minimized Clauses

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\emptyset \parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} \qquad \Longrightarrow \quad \text{(UnitProp)}$$

$$1 \parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$$

From SAT to SMT — Minimized Clauses

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} &
\end{array}$$

From SAT to SMT — Minimized Clauses

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} &
\end{array}$$

From SAT to SMT — Minimized Clauses

$$\underbrace{g(a) = c}_{1} \;\wedge\; \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \;\wedge\; \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{UnitProp}) \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{Decide}) \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{Decide}) \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{Theory Learn}) \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 &
\end{array}$$

From SAT to SMT — Minimized Clauses

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{UnitProp}) \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{Decide}) \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{Decide}) \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{Theory Learn}) \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 & \Longrightarrow \quad (\text{Backjump}) \\
1\ 2 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 &
\end{array}$$

From SAT to SMT — Minimized Clauses

$$\underbrace{g(a) = c}_{1} \;\wedge\; \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \;\wedge\; \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$\emptyset$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (UnitProp)
1	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Decide)
$1\ \overline{2}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Decide)
$1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Theory Learn)
$1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2$	$\Longrightarrow$ (Backjump)
$1\ 2$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2$	$\Longrightarrow$ (UnitProp)
$1\ 2\ 3\ \overline{4}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2$	

From SAT to SMT — Minimized Clauses

$$\underbrace{g(a) = c}_{1} \;\wedge\; \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \;\wedge\; \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$\emptyset$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (UnitProp)
1	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Decide)
$1\ \overline{2}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Decide)
$1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$ (Theory Learn)
$1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2$	$\Longrightarrow$ (Backjump)
$1\ 2$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2$	$\Longrightarrow$ (UnitProp)
$1\ 2\ 3\ \overline{4}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2$	$\Longrightarrow$ (Theory Learn)
$1\ 2\ 3\ \overline{4}$	$\parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2,\ \overline{1} \vee \overline{3} \vee 4$	

From SAT to SMT — Minimized Clauses

$$\underbrace{g(a) = c}_{1} \;\wedge\; \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \;\wedge\; \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$
\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Theory Learn)} \\
1\ \overline{2}^{\mathsf{d}}\ \overline{4}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 & \Longrightarrow \quad \text{(Backjump)} \\
1\ 2 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 & \Longrightarrow \quad \text{(UnitProp)} \\
1\ 2\ 3\ \overline{4} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 & \Longrightarrow \quad \text{(Theory Learn)} \\
1\ 2\ 3\ \overline{4} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2,\ \overline{1} \vee \overline{3} \vee 4 & \Longrightarrow \quad \text{(Fail)} \\
\mathit{fail} & & &
\end{array}
$$

Eager Conflict Detection

Currently, we have indicated that we will check M for T-satisfiability only when a pseudo-final state is reached.

In contrast, a more eager policy would be to check M for T-satisfiability every time M changes.

Experimental results show that this approach is significantly better most of the time.

It requires that Sat_T be online: able quickly to determine the consistency of incrementally more literals or to backtrack to a previous state.

It also requires that the SAT solver be instrumented to call Sat_T every time a variable is assigned a value.

From SAT to SMT — Eager Conflict Detection

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\emptyset \parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$$

From SAT to SMT — Eager Conflict Detection

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{lll} \emptyset \parallel & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{UnitProp}) \\ 1 \parallel & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \end{array}$$

From SAT to SMT — Eager Conflict Detection

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll} \emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\ 1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\ 1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \end{array}$$

From SAT to SMT — Eager Conflict Detection

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Theory Learn)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 &
\end{array}$$

From SAT to SMT — Eager Conflict Detection

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Theory Learn)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 & \Longrightarrow \quad \text{(Backjump)} \\
1\ 2 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 &
\end{array}$$

From SAT to SMT — Eager Conflict Detection

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$\emptyset$	$\parallel$ $1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$	(UnitProp)
1	$\parallel$ $1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$	(Decide)
$1\ \overline{2}^{\mathsf{d}}$	$\parallel$ $1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$	(Theory Learn)
$1\ \overline{2}^{\mathsf{d}}$	$\parallel$ $1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2$	$\Longrightarrow$	(Backjump)
$1\ 2$	$\parallel$ $1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2$	$\Longrightarrow$	(UnitProp)
$1\ 2\ 3\ \overline{4}$	$\parallel$ $1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2$		

From SAT to SMT — Eager Conflict Detection

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Decide)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Theory Learn)} \\
1\ \overline{2}^{\mathsf{d}} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 & \Longrightarrow \quad \text{(Backjump)} \\
1\ 2 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 & \Longrightarrow \quad \text{(UnitProp)} \\
1\ 2\ 3\ \overline{4} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2 & \Longrightarrow \quad \text{(Theory Learn)} \\
1\ 2\ 3\ \overline{4} & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2,\ \overline{1} \vee \overline{3} \vee 4 &
\end{array}$$

From SAT to SMT — Eager Conflict Detection

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$\emptyset$	$\parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$	(UnitProp)
1	$\parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$	(Decide)
$1\ \overline{2}^{\mathsf{d}}$	$\parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$	(Theory Learn)
$1\ \overline{2}^{\mathsf{d}}$	$\parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2$	$\Longrightarrow$	(Backjump)
$1\ 2$	$\parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2$	$\Longrightarrow$	(UnitProp)
$1\ 2\ 3\ \overline{4}$	$\parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2$	$\Longrightarrow$	(Theory Learn)
$1\ 2\ 3\ \overline{4}$	$\parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3},\ \overline{1} \vee 2,\ \overline{1} \vee \overline{3} \vee 4$	$\Longrightarrow$	(Fail)
$fail$			

Theory Propagation

A final improvement is to add the following rule:

Theory Propagate :

$$M \parallel F \quad \Longrightarrow \quad M\ l \parallel F \quad \textbf{if} \begin{cases} M \models_T l \\ l \text{ or } \neg l \text{ occurs in } F \\ l \text{ is undefined in } M \end{cases}$$

This rule allows a theory solver to inform the SAT solver if it happens to know that an unassigned literal is entailed by M.

Experimental results show that this can also be very helpful in practice.

From SAT to SMT — Theory Propagation

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\emptyset \parallel \quad 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$$

From SAT to SMT — Theory Propagation

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{lll} \emptyset \parallel & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad (\text{UnitProp}) \\ 1 \parallel & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \end{array}$$

From SAT to SMT — Theory Propagation

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Theory Propagate)} \\
1\ 2 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} &
\end{array}$$

From SAT to SMT — Theory Propagation

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(Theory Propagate)} \\
1\ 2 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow \quad \text{(UnitProp)} \\
1\ 2\ 3 & \| & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} &
\end{array}$$

From SAT to SMT — Theory Propagation

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$$\begin{array}{rcll}
\emptyset \parallel & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow & (\text{UnitProp}) \\
1 \parallel & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow & (\text{Theory Propagate}) \\
1\ 2 \parallel & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow & (\text{UnitProp}) \\
1\ 2\ 3 \parallel & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} & \Longrightarrow & (\text{Theory Propagate}) \\
1\ 2\ 3\ 4 \parallel & 1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3} &&
\end{array}$$

From SAT to SMT — Theory Propagation

$$\underbrace{g(a) = c}_{1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{2}} \vee \underbrace{g(a) = d}_{3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{4}} \vee \underbrace{g(a) \neq d}_{\overline{3}}$$

$\emptyset \parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$	(UnitProp)
$1 \parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$	(Theory Propagate)
$1\ 2 \parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$	(UnitProp)
$1\ 2\ 3 \parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$	(Theory Propagate)
$1\ 2\ 3\ 4 \parallel$	$1,\ \overline{2} \vee 3,\ \overline{4} \vee \overline{3}$	$\Longrightarrow$	(Fail)
$fail$			

Extensions

We briefly mention two extensions.

The first is to allow the theory solver to use the SAT solver for internal case splitting.

We do this by allowing the learning rule to introduce new variables and terms

Extended T-Learn :

$$M \parallel F \quad \Longrightarrow \quad M \parallel F, C \quad \textbf{if} \begin{cases} \text{each atom of } C \text{ occurs in } F \text{ or in } \mathcal{L}(M) \\ F \models_T \exists^*(C) \end{cases}$$

Quantifiers

The Abstract DPLL Modulo Theories framework can also be extended to include rules for quantifier instantiation.

- We extend the notion of literal to that of an abstract literal which may include quantified formulas in place of atomic formulas.
- Add two additional rules:

Inst_$\exists$:

$$M \parallel F \quad \Longrightarrow \quad M \parallel F, (\neg\exists\, x.\, P \vee P[x/sk]) \quad \textbf{if} \begin{cases} \exists\, x\, P \text{ is an abstract literal in } M \\ sk \text{ is a fresh constant.} \end{cases}$$

Inst_$\forall$:

$$M \parallel F \quad \Longrightarrow \quad M \parallel F, (\neg\forall\, x.\, P \vee P[x/t]) \quad \textbf{if} \begin{cases} \forall\, x\, P \text{ is an abstract literal in } M \\ t \text{ is a ground term.} \end{cases}$$

Quantifiers

The simple technique of quantifier instantiation is remarkably effective on verification benchmarks.

The main difficulty is coming up with the right terms to instantiate.

Matching techniques pioneered by Simplify have recently been adopted and improved by several modern SMT solvers.

SMT Solvers: State of the Art

Building on fast SAT technology, SMT solvers have been improving dramatically.

The winners of this year's SMT competition are orders of magnitude faster than those of just a couple of years ago.

Current leading solvers include:

- Barcelogic (U Barcelona, Spain)
- CVC3 (NYU, U Iowa)
- MathSAT (U Trento, Italy)
- Yices (SRI)
- Z3 (Microsoft)

SMT solvers are becoming the engine of choice for an ever-increasing set of verification applications.

Some Challenges

- More mature tools
- Better integration of SAT in SMT
- More complete techniques for quantifiers
- Parallel SMT
- Improving the SMT-LIB standard
- Producing and Checking Proofs
- Non-linear arithmetic