

Satisfiability Modulo Theories and DPLL(T)

Andrew Reynolds

March 18, 2015

Overview

SAT : Satisfiability for Propositional Logic

$$(A \vee B) \wedge (C \vee D) \wedge \neg B$$

- Does there exist truth values for A, B, C, D that make this formula true?

SMT : Satisfiability Modulo Theories

$$(x+1>0 \vee x+y>0) \wedge (x<0 \vee x+y>4) \wedge \neg x+y>0$$

- Does there exist integer values for x, y that make this formula true?

Theories:

- Linear Integer Arithmetic (LIA)
- Inductive Datatypes (DT)

Additional Topics:

- Combination of Theories (DT+LIA)
- Quantified Formulas

SAT

$$(A \vee B) \wedge (C \vee D) \wedge \neg B$$

DPLL algorithm

- Input : clauses in Clausal Normal Form (CNF)
- Alternates between:
 - Propagations : assign values to atoms whose value is forced
 - Decisions : choose an arbitrary value for an unassigned atom
- Answers SAT when all clauses have one literal \rightarrow true
- Answer UNSAT when made no decisions, one clause has all of its literals \rightarrow false

SAT

$$(A \vee B) \wedge (C \vee D) \wedge \neg B$$

DPLL algorithm

SAT

$$(A \vee B) \wedge (C \vee D) \wedge \neg B$$

DPLL algorithm

- Propagate : $B \rightarrow \text{false}$

SAT

$$(A \vee B) \wedge (C \vee D) \wedge \neg B$$

DPLL algorithm

- Propagate : $B \rightarrow \text{false}$
- Propagate : $A \rightarrow \text{true}$

SAT

$$(A \vee B) \wedge (C \vee D) \wedge \neg B$$

DPLL algorithm

- Propagate : $B \rightarrow \text{false}$
- Propagate : $A \rightarrow \text{true}$
- Decide : $C \rightarrow \text{true}$

SAT

$$(A \vee B) \wedge (C \vee D) \wedge \neg B$$

DPLL algorithm

- Propagate : $B \rightarrow \text{false}$
- Propagate : $A \rightarrow \text{true}$
- Decide : $C \rightarrow \text{true}$
 \Rightarrow Input is satisfiable

SAT

Optimizations:

- Two watched literals
- Conflict-driven clause learning (CDCL)
- Inprocessing

Using an encoding of problems into propositional SAT:

- Pro : Very efficient methods available
- Con : Not expressive

⇒ Motivation for Satisfiability *Modulo Theories*

Satisfiability Modulo Theories (SMT)

Extend SAT problems with reasoning about *theories*

- E.g. linear integer arithmetic (LIA) : $(x+1>0 \vee x+y>0) \wedge (x<0 \vee x+y>4)$

Formally, a theory T is a pair (Σ_T, I_T) , where:

- Σ_T is set of function symbols, the *signature* of T
 - E.g. $\Sigma_{LIA} = \{ +, -, <, \leq, >, \geq, 0, 1, 2, 3, \dots \}$
- I_T is a set of *interpretations* for T
 - E.g. each $I \in I_{LIA}$ interpret functions in Σ_{LIA} in standard way:
 - $1+1 = 2, 1+2 = 3, \dots$
 - $1 > 0 = \text{true}, 0 > 1 = \text{false}, \dots$
 - ...

SMT

$$(x+1>0 \vee x+y>0) \wedge (x<0 \vee x+y>4) \wedge \neg x+y>0$$

- DPLL(**T**) algorithm
 - Extends DPLL algorithm to incorporate reasoning about a theory **T**
 - Idea:
 - Use DPLL algorithm to find assignments for propositional abstraction of formula
 - Use off-the-shelf **SAT solver**
 - Check the *T-satisfiability* of assignments found by SAT solver
 - Use *Theory Solver for T*

SMT

$$(x+1 > 0 \vee x+y > 0) \wedge (x < 0 \vee x+y > 4) \wedge \neg x+y > 0$$

- DPLL(LIA) algorithm

Invoke DPLL(T) for theory T = LIA (linear integer arithmetic)

SMT

$$(x+1>0 \vee x+y>0) \wedge (x<0 \vee x+y>4) \wedge \neg x+y>0$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$

SMT

$$(A \vee B) \wedge (C \vee D) \wedge \neg B$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$
- Invoke SAT solver:

SMT

$$(A \vee B) \wedge (C \vee D) \wedge \neg B$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$
- Invoke SAT solver:
 - Propagate : $B \rightarrow \text{false}$, Propagate : $A \rightarrow \text{true}$, Decide : $C \rightarrow \text{true}$

SMT

$$(A \vee B) \wedge (C \vee D) \wedge \neg B$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$
- Invoke SAT solver:
 - Propagate : $B \rightarrow \text{false}$, Propagate : $A \rightarrow \text{true}$, Decide : $C \rightarrow \text{true}$
 - \Rightarrow Are LIA literals corresponding to $\{A, \neg B, C\}$ LIA-satisfiable?

SMT

$$(A \vee B) \wedge (C \vee D) \wedge \neg B$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$
- Invoke SAT solver:
 - Propagate : $B \rightarrow \text{false}$, Propagate : $A \rightarrow \text{true}$, Decide : $C \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0 \}$

SMT

$$(A \vee B) \wedge (C \vee D) \wedge \neg B$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$
- Invoke SAT solver:
 - Propagate : $B \rightarrow \text{false}$, Propagate : $A \rightarrow \text{true}$, Decide : $C \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0 \}$
 - $x+1>0 \wedge x<0$ is *LIA-unsatisfiable!*

SMT

$$\begin{aligned} & (A \vee B) \wedge (C \vee D) \wedge \neg B \wedge \\ & (\neg A \vee \neg C) \end{aligned}$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$
- Invoke SAT solver:
 - Propagate : $B \rightarrow \text{false}$, Propagate : $A \rightarrow \text{true}$, Decide : $C \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0 \}$
 - $(\neg A \vee \neg C)$ added to list of clauses
 - \Rightarrow Since one of $x+1>0, x<0$ must be false

SMT

$$\begin{aligned} & (A \vee B) \wedge (C \vee D) \wedge \neg B \wedge \\ & (\neg A \vee \neg C) \end{aligned}$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$
- Invoke SAT solver:
 - Propagate : $B \rightarrow \text{false}$, Propagate : $A \rightarrow \text{true}$, Decide : $C \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0 \}$
 - $(\neg A \vee \neg C)$ added to list of clauses
- Invoke SAT solver:
 - Backtrack decision on C

SMT

$$\begin{aligned} & (A \vee B) \wedge (C \vee D) \wedge \neg B \wedge \\ & (\neg A \vee \neg C) \end{aligned}$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$
- Invoke SAT solver:
 - Propagate : $B \rightarrow \text{false}$, Propagate : $A \rightarrow \text{true}$, Decide : $C \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0 \}$
 - $(\neg A \vee \neg C)$ added to list of clauses
- Invoke SAT solver:
 - Propagate : $C \rightarrow \text{false}$

SMT

$$\begin{aligned} & (\text{A} \vee \text{B}) \wedge (\text{C} \vee \text{D}) \wedge \neg \text{B} \wedge \\ & (\neg \text{A} \vee \neg \text{C}) \end{aligned}$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$
- Invoke SAT solver:
 - Propagate : $B \rightarrow \text{false}$, Propagate : $A \rightarrow \text{true}$, Decide : $C \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0 \}$
 - $(\neg A \vee \neg C)$ added to list of clauses
- Invoke SAT solver:
 - Propagate : $C \rightarrow \text{false}$, Propagate : $D \rightarrow \text{true}$

SMT

$$\begin{aligned} & (\text{A} \vee \text{B}) \wedge (\text{C} \vee \text{D}) \wedge \neg \text{B} \wedge \\ & (\neg \text{A} \vee \neg \text{C}) \end{aligned}$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$
- Invoke SAT solver:
 - Propagate : $B \rightarrow \text{false}$, Propagate : $A \rightarrow \text{true}$, Decide : $C \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0 \}$
 - $(\neg A \vee \neg C)$ added to list of clauses
- Invoke SAT solver:
 - Propagate : $C \rightarrow \text{false}$, Propagate : $D \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0, x+y>4 \}$

SMT

$$\begin{aligned} & (\text{A} \vee \text{B}) \wedge (\text{C} \vee \text{D}) \wedge \neg \text{B} \wedge \\ & (\neg \text{A} \vee \neg \text{C}) \end{aligned}$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$
- Invoke SAT solver:
 - Propagate : $B \rightarrow \text{false}$, Propagate : $A \rightarrow \text{true}$, Decide : $C \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0 \}$
 - $(\neg A \vee \neg C)$ added to list of clauses
- Invoke SAT solver:
 - Propagate : $C \rightarrow \text{false}$, Propagate : $D \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0, x+y>4 \}$
 - $\neg x+y>0 \wedge x+y>4$ is *LIA-unsatisfiable!*

SMT

$$\begin{aligned} & (A \vee B) \wedge (C \vee D) \wedge \neg B \wedge \\ & (\neg A \vee \neg C) \wedge (B \vee \neg D) \end{aligned}$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$
- Invoke SAT solver:
 - Propagate : $B \rightarrow \text{false}$, Propagate : $A \rightarrow \text{true}$, Decide : $C \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0 \}$
 - $(\neg A \vee \neg C)$ added to list of clauses
- Invoke SAT solver:
 - Propagate : $C \rightarrow \text{false}$, Propagate : $D \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0, x+y>4 \}$
 - $(B \vee \neg D)$ added to list of clauses

SMT

$$\begin{aligned} & (\text{A} \vee \text{B}) \wedge (\text{C} \vee \text{D}) \wedge \neg \text{B} \wedge \\ & (\neg \text{A} \vee \neg \text{C}) \wedge (\text{B} \vee \neg \text{D}) \end{aligned}$$

- DPLL(LIA) algorithm
- Map : $\{ A \Leftrightarrow x+1>0, B \Leftrightarrow x+y>0, C \Leftrightarrow x<0, D \Leftrightarrow x+y>4 \}$
- Invoke SAT solver:
 - Propagate : $B \rightarrow \text{false}$, Propagate : $A \rightarrow \text{true}$, Decide : $C \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0 \}$
 - $(\neg A \vee \neg C)$ added to list of clauses
- Invoke SAT solver:
 - Propagate : $C \rightarrow \text{false}$, Propagate : $D \rightarrow \text{true}$
- Invoke theory solver for LIA on: $\{ x+1>0, \neg x+y>0, x<0, x+y>4 \}$
 - $(B \vee \neg D)$ added to list of clauses
- No decisions to backtrack \Rightarrow *input is LIA-unsatisfiable*

SMT : Exercise

$$(x > y \vee x > z) \wedge (x + 1 < y \vee \neg x > y) \wedge (x > y \vee z > y)$$

- Determine if above formula is LIA-satisfiable

SMT : Exercise

$$\Phi \left\{ (A \vee B) \wedge (C \vee \neg A) \wedge (A \vee D) \right.$$

- Determine if Φ is LIA-satisfiable
- Map : $\{ A \Leftrightarrow x > y, B \Leftrightarrow x > z, C \Leftrightarrow x + 1 < y, D \Leftrightarrow z > y \}$

SMT : Exercise

$$\Phi \left\{ (A \vee B) \wedge (C \vee \neg A) \wedge (A \vee D) \right.$$

- Determine if Φ is LIA-satisfiable
- Map : $\{ A \Leftrightarrow x > y, B \Leftrightarrow x > z, C \Leftrightarrow x + 1 < y, D \Leftrightarrow z > y \}$
 - If **SAT**, give values for $\{ x, y, z \}$
 - If **UNSAT**, give a set of clauses C_1, \dots, C_n , where:
 - Φ, C_1, \dots, C_n is UNSAT
 - Each C_i is of the form $(l_1 \vee \dots \vee l_m)$, where:
 - Each l_i is one of $(\neg)A, (\neg)B, (\neg)C, (\neg)D$
 - Negation of formulas mapped to by $l_1 \dots l_m$ are LIA-unsatisfiable

SMT : Exercise

$$\Phi \left\{ (A \vee B) \wedge (C \vee \neg A) \wedge (A \vee D) \right.$$

- Determine if Φ is LIA-satisfiable
- Map : $\{ A \Leftrightarrow x > y, B \Leftrightarrow x > z, C \Leftrightarrow x + 1 < y, D \Leftrightarrow z > y \}$

SMT : Exercise

$$\Phi \left\{ (A \vee B) \wedge (C \vee \neg A) \wedge (A \vee D) \right.$$

- Determine if Φ is LIA-satisfiable
- Map : $\{ A \Leftrightarrow x > y, B \Leftrightarrow x > z, C \Leftrightarrow x + 1 < y, D \Leftrightarrow z > y \}$
- Invoke SAT solver:
 - Decide : $A \rightarrow \text{true}$, Propagate : $C \rightarrow \text{true}$

SMT : Exercise

$$\Phi \left\{ (A \vee B) \wedge (C \vee \neg A) \wedge (A \vee D) \right.$$

- Determine if Φ is LIA-satisfiable
- Map : $\{ A \Leftrightarrow x > y, B \Leftrightarrow x > z, C \Leftrightarrow x + 1 < y, D \Leftrightarrow z > y \}$
- Invoke SAT solver:
 - Decide : $A \rightarrow \text{true}$, Propagate : $C \rightarrow \text{true}$
 - Invoke theory solver for LIA on: $\{ x > y, x + 1 < y \}$

SMT : Exercise

$$\Phi = \left\{ \begin{array}{l} (A \vee B) \wedge (C \vee \neg A) \wedge (A \vee D) \\ (\neg A \vee \neg C) \end{array} \right.$$

- Determine if Φ is LIA-satisfiable
- Map : $\{ A \Leftrightarrow x > y, B \Leftrightarrow x > z, C \Leftrightarrow x + 1 < y, D \Leftrightarrow z > y \}$
- Invoke SAT solver:
 - Decide : $A \rightarrow \text{true}$, Propagate : $C \rightarrow \text{true}$
 - Invoke theory solver for LIA on: $\{ x > y, x + 1 < y \}$
 - $x > y \wedge x + 1 < y$ is LIA-unsatisfiable, add $(\neg A \vee \neg C)$

SMT : Exercise

$$\Phi \left\{ \begin{array}{l} (A \vee B) \wedge (C \vee \neg A) \wedge (A \vee D) \\ (\neg A \vee \neg C) \end{array} \right.$$

- Determine if Φ is LIA-satisfiable
- Map : $\{ A \Leftrightarrow x > y, B \Leftrightarrow x > z, C \Leftrightarrow x + 1 < y, D \Leftrightarrow z > y \}$
- Invoke SAT solver:
 - Decide : $A \rightarrow \text{true}$, Propagate : $C \rightarrow \text{true}$
 - Invoke theory solver for LIA on: $\{ x > y, x + 1 < y \}$
 - $x > y \wedge x + 1 < y$ is LIA-unsatisfiable, add $(\neg A \vee \neg C)$
 - Invoke SAT solver:
 - Backtrack decision on A

SMT : Exercise

$$\Phi = \left\{ \begin{array}{l} (A \vee B) \wedge (C \vee \neg A) \wedge (A \vee D) \\ (\neg A \vee \neg C) \end{array} \right.$$

- Determine if Φ is LIA-satisfiable
- Map : $\{ A \Leftrightarrow x > y, B \Leftrightarrow x > z, C \Leftrightarrow x + 1 < y, D \Leftrightarrow z > y \}$
- Invoke SAT solver:
 - Decide : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{false}$, Propagate : $C \rightarrow \text{true}$
 - Invoke theory solver for LIA on: $\{ x > y, x + 1 < y \}$
 - $x > y \wedge x + 1 < y$ is LIA-unsatisfiable, add $(\neg A \vee \neg C)$
 - Invoke SAT solver:
 - Propagate : $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$

SMT : Exercise

$$\Phi = \left\{ \begin{array}{l} (A \vee B) \wedge (C \vee \neg A) \wedge (A \vee D) \\ (\neg A \vee \neg C) \end{array} \right.$$

- Determine if Φ is LIA-satisfiable
- Map : $\{ A \Leftrightarrow x > y, B \Leftrightarrow x > z, C \Leftrightarrow x + 1 < y, D \Leftrightarrow z > y \}$
- Invoke SAT solver:
 - Decide : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{false}$, Propagate : $C \rightarrow \text{true}$
 - Invoke theory solver for LIA on: $\{ x > y, x + 1 < y \}$
 - $x > y \wedge x + 1 < y$ is LIA-unsatisfiable, add $(\neg A \vee \neg C)$
 - Invoke SAT solver:
 - Propagate : $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$
 - Invoke theory solver for LIA on: $\{ \neg x > y, x > z, z > y \}$

SMT : Exercise

$$\Phi = \left\{ \begin{array}{l} (A \vee B) \wedge (C \vee \neg A) \wedge (A \vee D) \\ (\neg A \vee \neg C) \wedge (A \vee \neg B \vee \neg D) \end{array} \right.$$

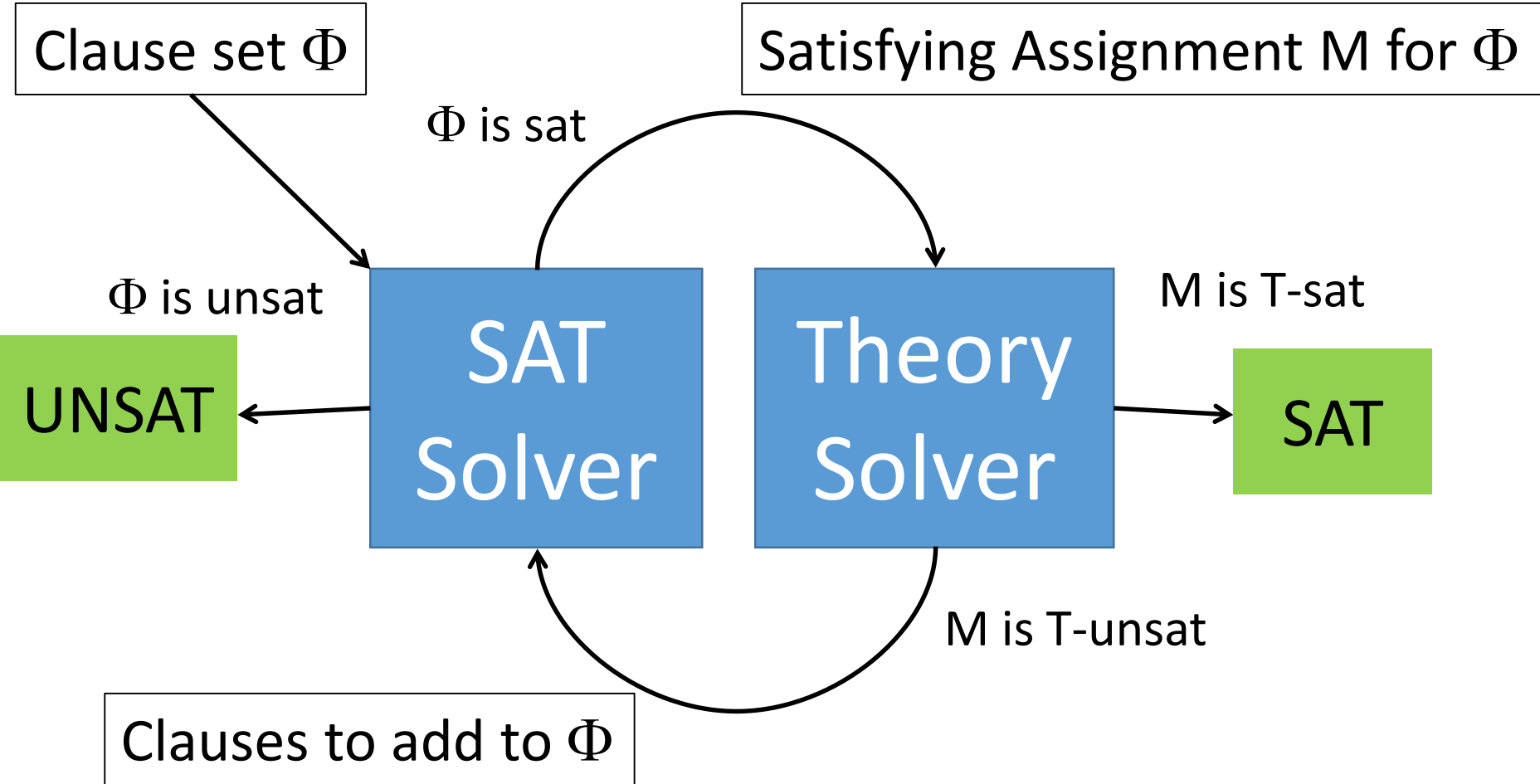
- Determine if Φ is LIA-satisfiable
 - Map : $\{ A \Leftrightarrow x > y, B \Leftrightarrow x > z, C \Leftrightarrow x + 1 < y, D \Leftrightarrow z > y \}$
 - Invoke SAT solver:
 - Decide : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{false}$, Propagate : $C \rightarrow \text{true}$
 - Invoke theory solver for LIA on: $\{ x > y, x + 1 < y \}$
 - $x > y \wedge x + 1 < y$ is LIA-unsatisfiable, add $(\neg A \vee \neg C)$
 - Invoke SAT solver:
 - Propagate : $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$
 - Invoke theory solver for LIA on: $\{ \neg x > y, x > z, z > y \}$
 - $\neg x > y \wedge x > z \wedge z > y$ is LIA-unsatisfiable, add $(A \vee \neg B \vee \neg D)$

SMT : Exercise

$$\Phi = \left\{ \begin{array}{l} (A \vee B) \wedge (C \vee \neg A) \wedge (A \vee D) \\ (\neg A \vee \neg C) \wedge (A \vee \neg B \vee \neg D) \end{array} \right.$$

- Determine if Φ is LIA-satisfiable
- Map : $\{ A \Leftrightarrow x > y, B \Leftrightarrow x > z, C \Leftrightarrow x + 1 < y, D \Leftrightarrow z > y \}$
- Invoke SAT solver:
 - Decide : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{false}$, Propagate : $C \rightarrow \text{true}$
 - Invoke theory solver for LIA on: $\{ x > y, x + 1 < y \}$
 - $x > y \wedge x + 1 < y$ is LIA-unsatisfiable, add $(\neg A \vee \neg C)$
 - Invoke SAT solver:
 - Propagate : $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$
 - Invoke theory solver for LIA on: $\{ \neg x > y, x > z, z > y \}$
 - $\neg x > y \wedge x > z \wedge z > y$ is LIA-unsatisfiable, add $(A \vee \neg B \vee \neg D)$
- No decisions to backtrack \Rightarrow *input is LIA-unsatisfiable*

DPLL(T)



DPLL(T) Theory Solvers

Input : A set of T-literals M

Output : either

1. M is T-satisfiable
2. $\{l_1, \dots, l_n\} \subseteq M$ is T-unsatisfiable
3. Don't know: return lemma

DPLL(T) Theory Solvers

Input : A set of T-literals M

Output : either

1. M is T-satisfiable

• Return *model*, e.g. $\{ x \rightarrow 2, y \rightarrow 3, z \rightarrow -3, \dots \}$

2. $\{ l_1, \dots, l_n \} \subseteq M$ is T-unsatisfiable

3. Don't know: return lemma

DPLL(T) Theory Solvers

Input : A set of T-literals M

Output : either

1. M is T-satisfiable

• Return *model*, e.g. $\{ x \rightarrow 2, y \rightarrow 3, z \rightarrow -3, \dots \}$

2. $\{ l_1, \dots, l_n \} \subseteq M$ is T-unsatisfiable

• Add T-*conflict* clause $(\neg l_1 \vee \dots \vee \neg l_n)$ to Φ

3. Don't know: return lemma

DPLL(T) Theory Solvers

Input : A set of T-literals M

Output : either

1. M is T-satisfiable

• Return *model*, e.g. $\{ x \rightarrow 2, y \rightarrow 3, z \rightarrow -3, \dots \}$

2. $\{ l_1, \dots, l_n \} \subseteq M$ is T-unsatisfiable

• Add T-*conflict* clause $(\neg l_1 \vee \dots \vee \neg l_n)$ to Φ

3. Don't know: return lemma

• Add clause to Φ , e.g. splitting on demand $(x = y \vee \neg x=y)$

DPLL(T) Theory Solvers

Input : A set of T-literals M

Output : either

1. M is T-satisfiable

- Return *model*, e.g. $\{ x \rightarrow 2, y \rightarrow 3, z \rightarrow -3, \dots \}$

⇒ Should be *solution-sound*

- Answers “ M is T-satisfiable” only if M is T-satisfiable

2. $\{ l_1, \dots, l_n \} \subseteq M$ is T-unsatisfiable

- Add T-*conflict* clause $(\neg l_1 \vee \dots \vee \neg l_n)$ to Φ

3. Don't know: return lemma

- Add clause to Φ , e.g. splitting on demand $(x = y \vee \neg x=y)$

DPLL(T) Theory Solvers

Input : A set of T-literals M

Output : either

1. M is T-satisfiable

- Return *model*, e.g. $\{ x \rightarrow 2, y \rightarrow 3, z \rightarrow -3, \dots \}$

⇒ Should be *solution-sound*

- Answers “ M is T-satisfiable” only if M is T-satisfiable

2. $\{ l_1, \dots, l_n \} \subseteq M$ is T-unsatisfiable

- Add T-*conflict* clause $(\neg l_1 \vee \dots \vee \neg l_n)$ to Φ

⇒ Should be *refutation-sound*

- Answers “ $\{ l_1, \dots, l_n \}$ is T-unsatisfiable” only if $\{ l_1, \dots, l_n \}$ is T-unsatisfiable

3. Don't know: return lemma

- Add clause to Φ , e.g. splitting on demand $(x = y \vee \neg x=y)$

DPLL(T) Theory Solvers

Input : A set of T-literals M

Output : either

1. M is T-satisfiable

- Return *model*, e.g. $\{ x \rightarrow 2, y \rightarrow 3, z \rightarrow -3, \dots \}$

⇒ Should be *solution-sound*

- Answers “ M is T-satisfiable” only if M is T-satisfiable

2. $\{ l_1, \dots, l_n \} \subseteq M$ is T-unsatisfiable

- Add T-*conflict* clause $(\neg l_1 \vee \dots \vee \neg l_n)$ to Φ

⇒ Should be *refutation-sound*

- Answers “ $\{ l_1, \dots, l_n \}$ is T-unsatisfiable” only if $\{ l_1, \dots, l_n \}$ is T-unsatisfiable

3. Don't know: return lemma

- Add clause to Φ , e.g. splitting on demand $(x = y \vee \neg x=y)$

⇒ If solver is solution-sound, refutation-sound, and *terminating*,

- Then it is a *decision procedure* for T

Design of DPLL(T) Theory Solvers

A DPLL(T) theory solver:

- Should be **solution-sound**, **refutation-sound**, **terminating**
- Should produce **models** when M is T-satisfiable
- Should produce **T-conflicts of minimal size** when M is T-unsatisfiable
- Should be designed to work *incrementally*
 - M is constantly being appended to/backtracked upon
- Can be designed to check T-satisfiability either:
 - **Eagerly**: Check if M is T-satisfiable immediately when any literal is added to M
 - **Lazily**: Check if M is T-satisfiable only when M is complete
- Should **cooperate** with other theory solvers when combining theories
 - (see later)

DPLL(T) Theory Solvers : Examples

SMT solvers incorporate:

- Theory solvers that are *decision procedures* for e.g.:
 - Theory of Equality and Uninterpreted Functions (EUF)
 - Congruence closure algorithm
 - Theory of Linear Integer/Real Arithmetic
 - Simplex algorithm
 - Theory of Arrays
 - Theory of Bit Vectors
 - Theory of Inductive Datatypes
 - ...
- Theory solvers that are *incomplete procedures* for e.g.:
 - Theory of Non-Linear Arithmetic
 - Theory of Strings + Length constraints
 - Quantified formulas

DPLL(T) Theory Solvers : Examples

SMT solvers incorporate:

- Theory solvers that are *decision procedures* for e.g.:
 - Theory of Equality and Uninterpreted Functions (EUF)
 - Congruence closure algorithm
 - Theory of Linear Integer/Real Arithmetic
 - Simplex algorithm
 - Theory of Arrays
 - Theory of Bit Vectors
 - *Theory of Inductive Datatypes*
 - ...
- Theory solvers that are *incomplete procedures* for e.g.:
 - Theory of Non-Linear Arithmetic
 - Theory of Strings + Length constraints
 - Quantified formulas

} Focus of the next part

Theory of Inductive Datatypes

Family of theories specified by a set of *types* with *constructors*:

$$D_1 := C_1(s_1 : T_1, \dots, s_i : T_i) \mid \dots \mid C_j(\dots)$$

...

$$D_m := C_{m1}(\dots) \mid \dots \mid C_{mk}(\dots)$$

- D_1, \dots, D_m are *datatypes*
- C_1, \dots, C_j are the *constructors* of datatype type D_1
- C_1 has subfields of type $T_1 \dots T_i$
 - s_1, \dots, s_i are the *selectors* for constructor C_1

Theory of Inductive Datatypes : Example

```
ClrList := cons( head : Clr, tail : ClrList ) | nil
```

```
Clr := red | green | blue
```

Theory of Inductive Datatypes : Example

```
ClrList := cons( head : Clr, tail : ClrList ) | nil
Clr := red | green | blue
```

Theory of Inductive Datatypes (DT) for ClrList and Clr

- $\Sigma_{DT} : \{ \text{cons}, \text{head}, \text{tail}, \text{nil}, \text{red}, \text{green}, \text{blue} \}$
- Interpretations I_{DT} are such that:
 - Terms with different constructors are distinct
 - $\text{red} \neq \text{green}$
 - Constructors are injective
 - If $\text{cons}(c_1, l_1) = \text{cons}(c_2, l_2)$, then $c_1 = c_2$ and $l_1 = l_2$
 - Terms of a datatype must have one of its constructors as its topmost symbol
 - Each c is such that $c = \text{red}$ or $c = \text{green}$ or $c = \text{blue}$
 - Selectors access subfields
 - $\text{head}(\text{cons}(c, l)) = c$
 - Terms do not contain themselves as subterms
 - $l \neq \text{cons}(c, l)$

Theory of Inductive Datatypes : Example

$\text{cons}(x, \text{nil}) = \text{cons}(y, z) \wedge (x = \text{red} \vee x = \text{blue}) \wedge y = \text{green}$

DPLL(DT) algorithm

Theory of Inductive Datatypes : Example

$$A \wedge (B \vee C) \wedge D$$

DPLL(DT) algorithm

- Map : { $A \Leftrightarrow \text{cons}(x, \text{nil}) = \text{cons}(y, z)$, $B \Leftrightarrow x = \text{red}$, $C \Leftrightarrow x = \text{blue}$, $D \Leftrightarrow y = \text{green}$ }

Theory of Inductive Datatypes : Example

$$\mathbf{A} \wedge (\mathbf{B} \vee \mathbf{C}) \wedge \mathbf{D}$$

DPLL(DT) algorithm

- Map : $\{ A \Leftrightarrow \text{cons}(x, \text{nil}) = \text{cons}(y, z), B \Leftrightarrow x = \text{red}, C \Leftrightarrow x = \text{blue}, D \Leftrightarrow y = \text{green} \}$
- Invoke SAT solver
 - Propagate : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$, Decide : $B \rightarrow \text{true}$

Theory of Inductive Datatypes : Example

$$\mathbf{A} \quad \wedge \quad (\mathbf{B} \quad \vee \quad \mathbf{C} \quad) \wedge \mathbf{D}$$

DPLL(DT) algorithm

- Map : $\{ A \Leftrightarrow \text{cons}(x, \text{nil}) = \text{cons}(y, z), B \Leftrightarrow x = \text{red}, C \Leftrightarrow x = \text{blue}, D \Leftrightarrow y = \text{green} \}$
- Invoke SAT solver
 - Propagate : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$, Decide : $B \rightarrow \text{true}$
- Invoke theory solver for DT on: $\{ \text{cons}(x, \text{nil}) = \text{cons}(y, z), x = \text{red}, y = \text{green} \}$

Theory of Inductive Datatypes : Example

$$A \wedge (B \vee C) \wedge D$$

DPLL(DT) algorithm

- Map : $\{ A \Leftrightarrow \text{cons}(x, \text{nil}) = \text{cons}(y, z), B \Leftrightarrow x = \text{red}, C \Leftrightarrow x = \text{blue}, D \Leftrightarrow y = \text{green} \}$
- Invoke SAT solver
 - Propagate : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$, Decide : $B \rightarrow \text{true}$
- Invoke theory solver for DT on: $\{ \text{cons}(x, \text{nil}) = \text{cons}(y, z), x = \text{red}, y = \text{green} \}$
 - $\text{cons}(x, \text{nil}) = \text{cons}(y, z) \wedge x = \text{red} \wedge y = \text{green}$ is *DT-unsatisfiable!*
 - \Rightarrow Since $\text{cons}(x, \text{nil}) = \text{cons}(y, \text{nil})$, we have $x = y$, but $x = \text{red}$ and $y = \text{green}$ and $\text{red} \neq \text{green}$

Theory of Inductive Datatypes : Example

$$\begin{array}{c} A \wedge (B \vee C) \wedge D \\ (\neg A \vee \neg B \vee \neg D) \end{array}$$

DPLL(DT) algorithm

- Map : $\{ A \Leftrightarrow \text{cons}(x, \text{nil}) = \text{cons}(y, z), B \Leftrightarrow x = \text{red}, C \Leftrightarrow x = \text{blue}, D \Leftrightarrow y = \text{green} \}$
- Invoke SAT solver
 - Propagate : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$, Decide : $B \rightarrow \text{true}$
- Invoke theory solver for DT on: $\{ \text{cons}(x, \text{nil}) = \text{cons}(y, z), x = \text{red}, y = \text{green} \}$
 - $(\neg A \vee \neg B \vee \neg D)$ added to list of clauses

Theory of Inductive Datatypes : Example

$$\begin{array}{c} A \wedge (B \vee C) \wedge D \\ (\neg A \vee \neg B \vee \neg D) \end{array}$$

DPLL(DT) algorithm

- Map : $\{ A \Leftrightarrow \text{cons}(x, \text{nil}) = \text{cons}(y, z), B \Leftrightarrow x = \text{red}, C \Leftrightarrow x = \text{blue}, D \Leftrightarrow y = \text{green} \}$
- Invoke SAT solver
 - Propagate : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$, Decide : $B \rightarrow \text{true}$
- Invoke theory solver for DT on: $\{ \text{cons}(x, \text{nil}) = \text{cons}(y, z), x = \text{red}, y = \text{green} \}$
 - $(\neg A \vee \neg B \vee \neg D)$ added to list of clauses
- Invoke SAT solver
 - Backtrack decision on B

Theory of Inductive Datatypes : Example

$$\begin{array}{c} A \wedge (B \vee C) \wedge D \\ (\neg A \vee \neg B \vee \neg D) \end{array}$$

DPLL(DT) algorithm

- Map : { $A \Leftrightarrow \text{cons}(x, \text{nil}) = \text{cons}(y, z)$, $B \Leftrightarrow x = \text{red}$, $C \Leftrightarrow x = \text{blue}$, $D \Leftrightarrow y = \text{green}$ }
- Invoke SAT solver
 - Propagate : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$, Decide : $B \rightarrow \text{true}$
- Invoke theory solver for DT on: { $\text{cons}(x, \text{nil}) = \text{cons}(y, z)$, $x = \text{red}$, $y = \text{green}$ }
 - $(\neg A \vee \neg B \vee \neg D)$ added to list of clauses
- Invoke SAT solver
 - Propagate : $B \rightarrow \text{false}$, Propagate : $C \rightarrow \text{true}$

Theory of Inductive Datatypes : Example

$$\begin{array}{c} A \wedge (B \vee C) \wedge D \\ (\neg A \vee \neg B \vee \neg D) \end{array}$$

DPLL(DT) algorithm

- Map : $\{ A \Leftrightarrow \text{cons}(x, \text{nil}) = \text{cons}(y, z), B \Leftrightarrow x = \text{red}, C \Leftrightarrow x = \text{blue}, D \Leftrightarrow y = \text{green} \}$
- Invoke SAT solver
 - Propagate : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$, Decide : $B \rightarrow \text{true}$
- Invoke theory solver for DT on: $\{ \text{cons}(x, \text{nil}) = \text{cons}(y, z), x = \text{red}, y = \text{green} \}$
 - $(\neg A \vee \neg B \vee \neg D)$ added to list of clauses
- Invoke SAT solver
 - Propagate : $B \rightarrow \text{false}$, Propagate : $C \rightarrow \text{true}$
- Invoke theory solver for DT on: $\{ \text{cons}(x, \text{nil}) = \text{cons}(y, z), \neg x = \text{red}, x = \text{blue}, y = \text{green} \}$

Theory of Inductive Datatypes : Example

$$\begin{array}{c} A \wedge (B \vee C) \wedge D \\ (\neg A \vee \neg B \vee \neg D) \end{array}$$

DPLL(DT) algorithm

- Map : $\{ A \Leftrightarrow \text{cons}(x, \text{nil}) = \text{cons}(y, z), B \Leftrightarrow x = \text{red}, C \Leftrightarrow x = \text{blue}, D \Leftrightarrow y = \text{green} \}$
- Invoke SAT solver
 - Propagate : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$, Decide : $B \rightarrow \text{true}$
- Invoke theory solver for DT on: $\{ \text{cons}(x, \text{nil}) = \text{cons}(y, z), x = \text{red}, y = \text{green} \}$
 - $(\neg A \vee \neg B \vee \neg D)$ added to list of clauses
- Invoke SAT solver
 - Propagate : $B \rightarrow \text{false}$, Propagate : $C \rightarrow \text{true}$
- Invoke theory solver for DT on: $\{ \text{cons}(x, \text{nil}) = \text{cons}(y, z), \neg x = \text{red}, x = \text{blue}, y = \text{green} \}$
 - $\text{cons}(x, \text{nil}) = \text{cons}(y, z) \wedge x = \text{blue} \wedge y = \text{green}$ is *DT-unsatisfiable!*
 - \Rightarrow Since $\text{cons}(x, \text{nil}) = \text{cons}(y, \text{nil})$, we have $x = y$, but $x = \text{red}$ and $y = \text{green}$ and $\text{red} \neq \text{green}$

Theory of Inductive Datatypes : Example

$$\begin{aligned} & A \wedge (B \vee C) \wedge D \\ & (\neg A \vee \neg B \vee \neg D) \wedge (\neg A \vee \neg C \vee \neg D) \end{aligned}$$

DPLL(DT) algorithm

- Map : $\{ A \Leftrightarrow \text{cons}(x, \text{nil}) = \text{cons}(y, z), B \Leftrightarrow x = \text{red}, C \Leftrightarrow x = \text{blue}, D \Leftrightarrow y = \text{green} \}$
- Invoke SAT solver
 - Propagate : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$, Decide : $B \rightarrow \text{true}$
- Invoke theory solver for DT on: $\{ \text{cons}(x, \text{nil}) = \text{cons}(y, z), x = \text{red}, y = \text{green} \}$
 - $(\neg A \vee \neg B \vee \neg D)$ added to list of clauses
- Invoke SAT solver
 - Propagate : $B \rightarrow \text{false}$, Propagate : $C \rightarrow \text{true}$
- Invoke theory solver for DT on: $\{ \text{cons}(x, \text{nil}) = \text{cons}(y, z), \neg x = \text{red}, x = \text{blue}, y = \text{green} \}$
 - $(\neg A \vee \neg C \vee \neg D)$ added to list of clauses

Theory of Inductive Datatypes : Example

$$\begin{aligned} & A \wedge (B \vee C) \wedge D \\ & (\neg A \vee \neg B \vee \neg D) \wedge (\neg A \vee \neg C \vee \neg D) \end{aligned}$$

DPLL(DT) algorithm

- Map : $\{ A \Leftrightarrow \text{cons}(x, \text{nil}) = \text{cons}(y, z), B \Leftrightarrow x = \text{red}, C \Leftrightarrow x = \text{blue}, D \Leftrightarrow y = \text{green} \}$
- Invoke SAT solver
 - Propagate : $A \rightarrow \text{true}$, Propagate : $D \rightarrow \text{true}$, Decide : $B \rightarrow \text{true}$
- Invoke theory solver for DT on: $\{ \text{cons}(x, \text{nil}) = \text{cons}(y, z), x = \text{red}, y = \text{green} \}$
 - $(\neg A \vee \neg B \vee \neg D)$ added to list of clauses
- Invoke SAT solver
 - Propagate : $B \rightarrow \text{false}$, Propagate : $C \rightarrow \text{true}$
- Invoke theory solver for DT on: $\{ \text{cons}(x, \text{nil}) = \text{cons}(y, z), \neg x = \text{red}, x = \text{blue}, y = \text{green} \}$
 - $(\neg A \vee \neg C \vee \neg D)$ added to list of clauses
- No decisions to backtrack \Rightarrow *input is DT-unsatisfiable*

Combination of Theories

What if we have:

$$\text{IntList} := \text{cons}(\text{head} : \text{Int}, \text{tail} : \text{IntList}) \mid \text{nil}$$

Example input:

$$(\text{head}(x) + 3 = y \vee x = \text{cons}(y + 1, \text{nil})) \wedge \text{head}(x) > y + 1$$

⇒ Requires reasoning about **datatypes and integers**

Combination of Theories

What if we have:

```
IntList := cons( head : Int, tail : IntList ) | nil
```

Example input:

$$(\text{head}(x) + 3 = y \vee x = \text{cons}(y + 1, \text{nil})) \wedge \text{head}(x) > y + 1$$

Idea:

- *Purify* the literals in the input
- Use DPLL(LIA+DT): find satisfying assignments $M = M_{\text{LIA}} \cup M_{\text{DT}}$
 - Use **existing solver for LIA** to check if M_{LIA} is LIA-satisfiable
 - Use **existing solver for DT** to check if M_{DT} is DT-satisfiable
- If either of $\{ M_{\text{LIA}}, M_{\text{DT}} \}$ is T-unsatisfiable, then M is T-unsatisfiable
- If both $\{ M_{\text{LIA}}, M_{\text{DT}} \}$ are T-satisfiable, then solvers must combine models
 - Communicate equalities between *shared terms*

Combination of Theories : Example

$$(\text{head}(x)+3 = y \vee x = \text{cons}(y+1, \text{nil})) \wedge \text{head}(x) > y+1$$

DPLL(LIA+DT) algorithm

Combination of Theories : Example

(A \vee B) \wedge C

DPLL(LIA+DT) algorithm

- Map : { A \Leftrightarrow head(x)+3 =y, B \Leftrightarrow x=cons(y+1, nil), C \Leftrightarrow head(x) > y+1 }

Combination of Theories : Example

(A \vee B) \wedge C

DPLL(LIA+DT) algorithm

- Map : { A \Leftrightarrow `head(x)+3=y`, B \Leftrightarrow `x=cons(y+1, nil)`, C \Leftrightarrow `head(x) > y+1` }
- Purify A, B, C, e.g. introduce fresh “shared” variables such that:
 - Each literal contains function symbols only belonging to one theory (DT or LIA)

Combination of Theories : Example

(A \vee B) \wedge C

DPLL(LIA+DT) algorithm

- Map : { A \Leftrightarrow $u_1 = y$, B \Leftrightarrow $x = \text{cons}(u_2, \text{nil})$, C \Leftrightarrow $u_2 = y + 1$ }
- Map shared variables : { $u_1 \Leftrightarrow \text{head}(x)$, $u_2 \Leftrightarrow y + 1$ }

Combination of Theories : Example

(**A** \vee B) \wedge **C**

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow \exists y, B \Leftrightarrow x = \text{cons}(u_1, \text{nil}), C \Leftrightarrow u_1 = y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$

Combination of Theories : Example

(**A** \vee B) \wedge **C**

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow u_1 + 3 = y, B \Leftrightarrow x = \text{cons}(u_2, \text{nil}), C \Leftrightarrow u_1 > y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers on : $\{ u_1 + 3 = y, u_1 > y + 1 \} \cup \{ u_1 = \text{head}(x), u_2 = y + 1 \}$

Combination of Theories : Example

$$(A \vee B) \wedge C$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow u_1 + 3 = y, B \Leftrightarrow x = \text{cons}(u_2, \text{nil}), C \Leftrightarrow u_1 > y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$

Combination of Theories : Example

(**A** \vee B) \wedge **C**

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow u_1 + 3 = y, B \Leftrightarrow x = \text{cons}(u_2, \text{nil}), C \Leftrightarrow u_1 > y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$

Combination of Theories : Example

$$\left(A \vee B \right) \wedge C$$
$$\left(\neg A \vee \neg C \right)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow u_1 + 3 = y, B \Leftrightarrow x = \text{cons}(u_2, \text{nil}), C \Leftrightarrow u_2 = y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$

Combination of Theories : Example

$$(A \vee B) \wedge C$$
$$(\neg A \vee \neg C)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow u_1 + 3 = y, B \Leftrightarrow x = \text{cons}(u_2, \text{nil}), C \Leftrightarrow u_1 > y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
- Invoke SAT solver : Backtrack decision on A

Combination of Theories : Example

$$\left(\boxed{A} \vee \boxed{B} \right) \wedge \boxed{C}$$
$$\left(\boxed{\neg A} \vee \boxed{\neg C} \right)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow u_1 + 3 = y, B \Leftrightarrow x = \text{cons}(u_2, \text{nil}), C \Leftrightarrow u_2 = y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
- Invoke SAT solver : Propagate $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$

Combination of Theories : Example

$$(A \vee B) \wedge C$$

$$(\neg A \vee \neg C)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow u_1 + 3 = y, B \Leftrightarrow x = \text{cons}(u_2, \text{nil}), C \Leftrightarrow u_1 > y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
- Invoke SAT solver : Propagate $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$
- Invoke theory solvers on : $\{ \neg u_1 + 3 = y, x = \text{cons}(u_2, \text{nil}), u_1 > y + 1 \} \cup \{ u_1 = \text{head}(x), u_2 = y + 1 \}$

Combination of Theories : Example

$$(A \vee B) \wedge C$$

$$(\neg A \vee \neg C)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow u_1 + 3 = y, B \Leftrightarrow x = \text{cons}(u_2, \text{nil}), C \Leftrightarrow u_1 > y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
- Invoke SAT solver : Propagate $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ x = \text{cons}(u_2, \text{nil}), u_1 = \text{head}(x) \}$
 - Solver for LIA on : $\{ \neg u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$

Combination of Theories : Example

$$((A \vee B) \wedge C) \wedge (\neg A \vee \neg C)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow \exists y, B \Leftrightarrow x = \text{cons}(u_1, \text{nil}), C \Leftrightarrow u_1 = y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
- Invoke SAT solver : Propagate $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ x = \text{cons}(u_2, \text{nil}), u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ \neg u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$

Combination of Theories : Example

$$((A \vee B) \wedge C) \wedge (\neg A \vee \neg C)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow \exists y, B \Leftrightarrow x = \text{cons}(u_1, \text{nil}), C \Leftrightarrow u_1 = y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
- Invoke SAT solver : Propagate $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ x = \text{cons}(u_2, \text{nil}), u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ \neg u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... LIA-satisfiable

Combination of Theories : Example

$$\left(\boxed{A} \vee \boxed{B} \right) \wedge \boxed{C}$$
$$\left(\boxed{\neg A} \vee \boxed{\neg C} \right)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow \exists z = y, B \Leftrightarrow x = \text{cons}(z, \text{nil}), C \Leftrightarrow z = y + 1 \}$
 - Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
 - Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
 - Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
 - Invoke SAT solver : Propagate $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$
 - Invoke theory solvers
 - Solver for DT on : $\{ x = \text{cons}(u_2, \text{nil}), u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ \neg u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... LIA-satisfiable
- \Rightarrow To answer "satisfiable", theory solvers must agree on equalities between shared variables u_1, u_2

Combination of Theories : Example

$$((A \vee B) \wedge C) \wedge (\neg A \vee \neg C)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow \exists y, B \Leftrightarrow x = \text{cons}(u_1, \text{nil}), C \Leftrightarrow u_1 = y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
- Invoke SAT solver : Propagate $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ x = \text{cons}(u_2, \text{nil}), u_1 = \text{head}(x) \}$... DT-satisfiable. Is $u_1 = u_2$?
 - Solver for LIA on : $\{ \neg u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... LIA-satisfiable. Is $u_1 = u_2$?

Combination of Theories : Example

$$(A \vee B) \wedge C$$

$$(\neg A \vee \neg C)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow \exists y, B \Leftrightarrow x = \text{cons}(u_1, \text{nil}), C \Leftrightarrow u_1 = y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
- Invoke SAT solver : Propagate $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ x = \text{cons}(u_2, \text{nil}), u_1 = \text{head}(x) \}$... DT-satisfiable. Is $u_1 = u_2$? YES: $u_1 = \text{head}(x) = u_2$
 - Solver for LIA on : $\{ \neg u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... LIA-satisfiable. Is $u_1 = u_2$? NO: $u_2 = y + 1 < u_1$

Combination of Theories : Example

$$((A \vee B) \wedge C) \wedge (\neg A \vee \neg C)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow u_1 + 3 = y, B \Leftrightarrow x = \text{cons}(u_2, \text{nil}), C \Leftrightarrow u_1 > y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
- Invoke SAT solver : Propagate $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ x = \text{cons}(u_2, \text{nil}), u_1 = \text{head}(x) \}$
 - Solver for LIA on : $\{ \neg u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1, u_1 = u_2 \}$

} DT-solver tells LIA-solver $u_1 = u_2$
 ...since $x = \text{cons}(u_2, \text{nil}) \wedge u_1 = \text{head}(x)$

Combination of Theories : Example

$$(A \vee B) \wedge C$$

$$(\neg A \vee \neg C)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow u_1 + 3 = y, B \Leftrightarrow x = \text{cons}(u_2, \text{nil}), C \Leftrightarrow u_1 = y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
- Invoke SAT solver : Propagate $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ x = \text{cons}(u_2, \text{nil}), u_1 = \text{head}(x) \}$
 - Solver for LIA on : $\{ \neg u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1, u_1 = u_2 \}$
 - $u_1 > y + 1 \wedge u_2 = y + 1 \wedge u_1 = u_2$ is LIA-unsatisfiable!

$u_1 = u_2$
...since $x = \text{cons}(u_2, \text{nil}) \wedge u_1 = \text{head}(x)$

Combination of Theories : Example

$$((A \vee B) \wedge C) \wedge (\neg A \vee \neg C)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow u_1 + 3 = y, B \Leftrightarrow x = \text{cons}(u_2, \text{nil}), C \Leftrightarrow u_1 = y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
- Invoke SAT solver : Propagate $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ x = \text{cons}(u_2, \text{nil}), u_1 = \text{head}(x) \}$
 - Solver for LIA on : $\{ \neg u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1, u_1 = u_2 \}$
 - Construct conflict based on *explanation* of $u_1 > y + 1, u_2 = y + 1, u_1 = u_2$

$$u_1 = u_2$$

...since $x = \text{cons}(u_2, \text{nil}) \wedge u_1 = \text{head}(x)$

Combination of Theories : Example

$$(A \vee B) \wedge C$$

$$(\neg A \vee \neg C) \wedge (\neg B \vee \neg C)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow \exists z = y, B \Leftrightarrow x = \text{cons}(z, \text{nil}), C \Leftrightarrow z = y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
- Invoke SAT solver : Propagate $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ x = \text{cons}(u_2, \text{nil}), u_1 = \text{head}(x) \}$
 - Solver for LIA on : $\{ \neg u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1, u_1 = u_2 \}$
 - Add $(\neg B \vee \neg C)$

$u_1 = u_2$
 ...since $x = \text{cons}(u_2, \text{nil}) \wedge u_1 = \text{head}(x)$

Combination of Theories : Example

$$(A \vee B) \wedge C$$

$$(\neg A \vee \neg C) \wedge (\neg B \vee \neg C)$$

DPLL(LIA+DT) algorithm

- Map : $\{ A \Leftrightarrow u_1 + 3 = y, B \Leftrightarrow x = \text{cons}(u_2, \text{nil}), C \Leftrightarrow u_1 = y + 1 \}$
- Map shared variables : $\{ u_1 \Leftrightarrow \text{head}(x), u_2 \Leftrightarrow y + 1 \}$
- Invoke SAT solver: Propagate : $C \rightarrow \text{true}$, Decide : $A \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ u_1 = \text{head}(x) \}$... DT-satisfiable
 - Solver for LIA on : $\{ u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1 \}$... $u_1 + 3 = y \wedge u_1 > y + 1$ is LIA-unsatisfiable! Add $(\neg A \vee \neg C)$
- Invoke SAT solver : Propagate $A \rightarrow \text{false}$, Propagate : $B \rightarrow \text{true}$
- Invoke theory solvers
 - Solver for DT on : $\{ x = \text{cons}(u_2, \text{nil}), u_1 = \text{head}(x) \}$
 - Solver for LIA on : $\{ \neg u_1 + 3 = y, u_1 > y + 1, u_2 = y + 1, u_1 = u_2 \}$
 - Add $(\neg B \vee \neg C)$

\Rightarrow Input is *DT+LIA-unsatisfiable!*

$u_1 = u_2$
...since $x = \text{cons}(u_2, \text{nil}) \wedge u_1 = \text{head}(x)$

SMT : Theory Combination


Nelson-Oppen Theory Combination

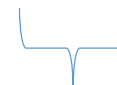
- SMT solvers use preexisting theory solvers for combined theories $T_1 + \dots + T_n$
- Basic idea given purified set of literals M :
 - Partition and distribute M to T_1 -solver, ..., T_n -solver
 - If any T_i -solver says “unsat”, then M is unsatisfiable
 - If each T_i -solver says “sat”, then solvers must agree on equalities between shared variables
- Requires theory solvers to:
 - Have disjoint signatures
 - E.g. arithmetic has functions $\{ +, <, 0, 1, \dots \}$, datatypes has functions $\{ \text{cons}, \text{head}, \text{tail}, \dots \}$
 - Know equalities/disequalities between shared variables
 - E.g. are $u_1 = u_2$ equal?
 - Theories agree on cardinalities for shared types
 - E.g. LIA and DT may agree that Int has infinite cardinality

Quantified Formulas

What if input has *quantifiers*:

$$(\forall x. P(x) \vee \forall x. \neg P(x)) \wedge P(a) \wedge \exists x. \neg P(x)$$


for all Int x


there exists an Int x

Quantified Formulas

What if input has *quantifiers*:

$$(\forall x. P(x) \vee \forall x. \neg P(x)) \wedge P(a) \wedge \exists x. \neg P(x)$$

Problem is generally *undecidable*

- E.g. no procedure for checking T-satisfiability of $\{ \forall x. P(x), P(a), \dots \}$

Quantified Formulas

What if input has *quantifiers*:

$$(\forall x. P(x) \vee \forall x. \neg P(x)) \wedge P(a) \wedge \neg P(k)$$

Problem is generally *undecidable*

- E.g. no procedure for checking T-satisfiability of $\{ \forall x. P(x), P(a), \dots \}$

Witness existential quantification

- Introduce a fresh constant that witnesses the formula, so $\exists x. \neg P(x)$ becomes $\neg P(k)$

Quantified Formulas

What if input has *quantifiers*:

$$\begin{aligned} & (\forall x. P(x) \vee \forall x. \neg P(x)) \wedge P(a) \wedge \neg P(k) \\ & (\neg \forall x. P(x) \vee P(a)) \wedge (\neg \forall x. P(x) \vee P(k)), \dots \end{aligned}$$

Problem is generally *undecidable*

- E.g. no procedure for checking T-satisfiability of $\{ \forall x. P(x), P(a), \dots \}$

Witness existential quantification

- Introduce a fresh constant that witnesses the formula, so $\exists x. \neg P(x)$ becomes $\neg P(k)$

Instantiate universal quantification

- Add clauses of the form $(\neg \forall x. P(x) \vee P(a))$
 - Either P does not hold for all x, or it holds for a

Quantified Formulas

What if input has *quantifiers*:

$$\begin{aligned} & (\forall x. P(x) \vee \forall x. \neg P(x)) \wedge P(a) \wedge \neg P(k) \\ & (\neg \forall x. P(x) \vee P(a)) \wedge (\neg \forall x. P(x) \vee P(k)), \dots \end{aligned}$$

Problem is generally *undecidable*

- E.g. no procedure for checking T-satisfiability of $\{ \forall x. P(x), P(a), \dots \}$

Witness existential quantification

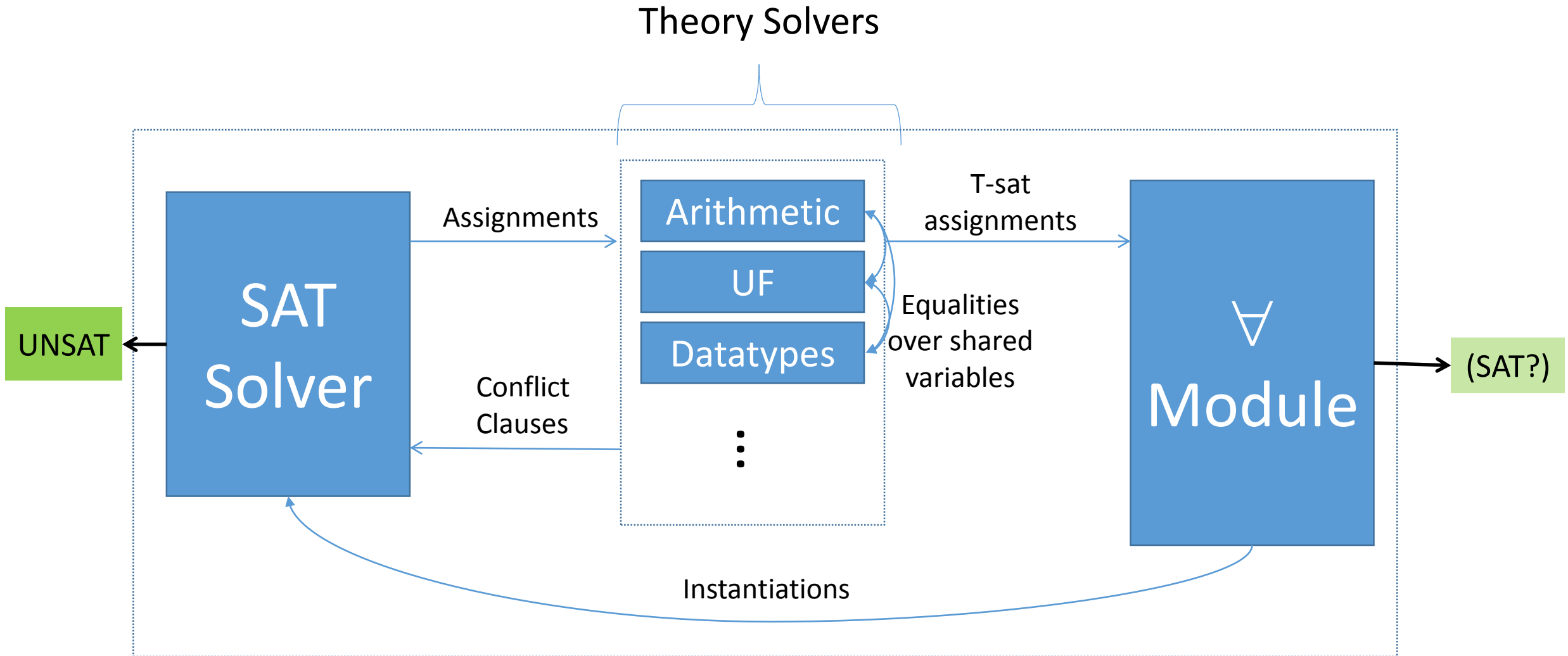
- Introduce a fresh constant that witnesses the formula, so $\exists x. \neg P(x)$ becomes $\neg P(k)$

Instantiate universal quantification

- Add clauses of the form $(\neg \forall x. P(x) \vee P(a))$

\Rightarrow *Sound but incomplete*, thus may be unable to answer “sat”

SMT Solver for Quantifiers : Summary



SMT Summary

SMT solvers use

- DPLL(T) algorithm for theory T, which uses:
 - Off-the-shelf SAT solver
 - Theory solver for T
- Nelson-Oppen theory combination for combined theories $T_1 + T_2$, which uses:
 - Existing theory solvers for T_1 and T_2
- Incomplete methods for quantified formulas
 - Primarily use instantiation-based procedure for (universal) quantification