Introduction to Satisfiability Modulo Theories

Martin Jonáš

IA072 – Seminar on Concurrency

March 3, 2017

First-Order Logic

Propositional logic speaks only about truth value – true or false.

First-order logic speaks about objects, their properties and relations among them.

First-Order Logic

Propositional logic speaks only about truth value – true or false.

First-order logic speaks about objects, their properties and relations among them.

Examples

- $\exists s.\, \mathrm{Human}(s) \,\wedge\, \mathrm{Mortal}(s)$.
- $\forall s.\, \mathrm{Human}(s) \,\rightarrow\, \mathrm{Mortal}(s)$.
- $\exists x \exists y.\, x < 5 \,\wedge\, y < 3 \,\wedge\, 2 \cdot (x + y) > 20$.

First-Order Logic

Propositional logic speaks only about truth value – true or false.

First-order logic speaks about objects, their properties and relations among them.

Examples

- $\exists s.\, \mathrm{Human}(s) \wedge \mathrm{Mortal}(s)$.
- $\forall s.\, \mathrm{Human}(s) \rightarrow \mathrm{Mortal}(s)$.
- $\exists x \exists y.\, x < 5 \wedge y < 3 \wedge 2 \cdot (x + y) > 20$.

First-Order Logic

Propositional logic speaks only about truth value – true or false.

First-order logic speaks about objects, their properties and relations among them.

Examples

- $\exists s.\, \mathrm{Human}(s) \;\wedge\; \mathrm{Mortal}(s).$
- $\forall s.\, \mathrm{Human}(s) \;\rightarrow\; \mathrm{Mortal}(s).$
- $\exists x \exists y.\, x < 5 \;\wedge\; y < 3 \;\wedge\; 2 \cdot (x + y) > 20.$

In addition to logical symbols, first-order formulas contain variables, constant symbols, function symbols, and predicate symbols.

First-Order Logic

Propositional logic speaks only about truth value – true or false.

First-order logic speaks about objects, their properties and relations among them.

Examples

- $\exists s.\, \mathrm{Human}(s) \ \land\ \mathrm{Mortal}(s).$
- $\forall s.\, \mathrm{Human}(s) \ \rightarrow\ \mathrm{Mortal}(s).$
- $\exists x \exists y.\, x < 5 \ \land\ y < 3 \ \land\ 2 \cdot (x + y) > 20.$

In addition to logical symbols, first-order formulas contain variables, function symbols, and predicate symbols.

First-Order Logic – Syntax

Suppose we have a set $\Sigma^F = \{f, g, \ldots\}$ of function symbols and a set $\Sigma^P = \{R, S, \ldots\}$ of predicate symbols.

First-Order Logic – Syntax

Suppose we have a set $\Sigma^F = \{f, g, \ldots\}$ of function symbols and a set $\Sigma^P = \{R, S, \ldots\}$ of predicate symbols.

(Σ-)Term

1. a variable – $x, y, z, \ldots$
2. a function symbol applied to terms – $f(x)$, $g(f(x), y)$, …

First-Order Logic – Syntax

Suppose we have a set $\Sigma^F = \{f, g, \ldots\}$ of function symbols and a set $\Sigma^P = \{R, S, \ldots\}$ of predicate symbols.

(Σ-)Term

1. a variable – $x, y, z, \ldots$
2. a function symbol applied to terms – $f(x)$, $g(f(x), y)$, ...

(Σ)**-Literal**

1. a predicate symbol applied to terms – $R(x), S(f(x), y), \ldots$
2. a negation of predicate symbol applied to terms – $\neg R(x), \neg S(f(x), y), \ldots$

First-Order Logic – Syntax

Suppose we have a set $\Sigma^F = \{f, g, \dots\}$ of function symbols and a set $\Sigma^P = \{R, S, \dots\}$ of predicate symbols.

(Σ-)Term

1. a variable – $x, y, z, \dots$
2. a function symbol applied to terms – $f(x)$, $g(f(x), y)$, …

(Σ)**-Literal**

1. a predicate symbol applied to terms – $R(x), S(f(x), y), \dots$
2. a negation of predicate symbol applied to terms – $\neg R(x), \neg S(f(x), y), \dots$

(Σ)**-Formula**

1. a Boolean combination of literals – $(R(x) \vee \neg R(y)) \wedge S(f(x), y), \dots$
2. a quantifier applied to a formula – $\forall x.\, R(x), \dots$.

First-Order Logic – Syntax

Suppose we have a set $\Sigma^F = \{f, g, \ldots\}$ of function symbols and a set $\Sigma^P = \{R, S, \ldots\}$ of predicate symbols.

(Σ-)Term

1. a variable – $x, y, z, \ldots$
2. a function symbol applied to terms – $f(x)$, $g(f(x), y)$, ...

(Σ)**-Literal**

1. a predicate symbol applied to terms – $R(x), S(f(x), y), \ldots$
2. a negation of predicate symbol applied to terms – $\neg R(x), \neg S(f(x), y), \ldots$

(Σ)**-Formula**

1. a Boolean combination of literals – $(R(x) \vee \neg R(y)) \wedge S(f(x), y), \ldots$
2. a quantifier applied to a formula – $\forall x.\, R(x), \ldots$.

The set $\Sigma = \Sigma^F \cup \Sigma^P$ is called a signature.

First-Order Logic – Semantics

Is the following formula true?

$$\forall x \exists y. <(x, y) \ \land \ <(y, +(x, 1))$$

First-Order Logic – Semantics

Is the following formula true?

$$\forall x \exists y.\, x < y \;\wedge\; y < x + 1$$

First-Order Logic – Semantics

Is the following formula true?

$$\forall x \exists y.\, x < y \;\wedge\; y < x + 1$$

It depends.

First-Order Logic – Semantics

Is the following formula true?

$$\forall x \exists y.\, x < y \ \wedge \ y < x + 1$$

It depends.

- What are x, y, z?
- What does the function symbol $+$ mean?
- What does the relation symbol $<$ mean?

First-Order Logic – Semantics

Is the following formula true?

$$\forall x \exists y.\, x < y \;\wedge\; y < x + 1$$

It depends.

- What are x, y, z?
- What does the function symbol $+$ mean?
- What does the relation symbol $<$ mean?

Meaning of these three things is given by a Σ-structure.

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$,
   - we suppose that $=^{A}$ is the identity relation.

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$,
   - we suppose that $=^{A}$ is the identity relation.

Given a Σ-structure and an assignment μ of variables to elements of A, we can evaluate each formula.

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{ar(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{ar(R)}$,
   - we suppose that $=^A$ is the identity relation.

Given a Σ-structure and an assignment μ of variables to elements of A, we can evaluate each formula.

- $A = \mathbb{Z}$
- $+^{\mathcal{A}}(x, y) = x + y$
- $<^{\mathcal{A}} (x, y) \iff x < y$
- $1^{\mathcal{A}} = 1$
- $\mu(x) = 1, \mu(y) = 3$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$,
   - we suppose that $=^{A}$ is the identity relation.

Given a Σ-structure and an assignment μ of variables to elements of A, we can evaluate each formula.

- $A = \mathbb{Z}$
- $+^{\mathcal{A}}(x, y) = x + y$
- $<^{\mathcal{A}}(x, y) \iff x < y$
- $1^{\mathcal{A}} = 1$
- $\mu(x) = 1, \mu(y) = 3$

$$(x < y) \wedge (y + 1 < x)$$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$,
   - we suppose that $=^{\mathcal{A}}$ is the identity relation.

Given a Σ-structure and an assignment μ of variables to elements of A, we can evaluate each formula.

- $A = \mathbb{Z}$
- $+^{\mathcal{A}}(x, y) = x + y$
- $<^{\mathcal{A}} (x, y) \iff x < y$
- $1^{\mathcal{A}} = 1$
- $\mu(x) = 1, \mu(y) = 3$

$$(\mu(x) <^{\mathcal{A}} \mu(y)) \land (\mu(y) +^{\mathcal{A}} 1^{\mathcal{A}} <^{\mathcal{A}} \mu(x))$$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$,
   - we suppose that $=^{A}$ is the identity relation.

Given a Σ-structure and an assignment μ of variables to elements of A, we can evaluate each formula.

- $A = \mathbb{Z}$
- $+^{\mathcal{A}}(x, y) = x + y$
- $<^{\mathcal{A}} (x, y) \iff x < y$
- $1^{\mathcal{A}} = 1$
- $\mu(x) = 1, \mu(y) = 3$

$$(1 <^{\mathcal{A}} 3) \wedge (3 +^{\mathcal{A}} 1^{\mathcal{A}} <^{A} 1)$$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$,
   - we suppose that $=^{A}$ is the identity relation.

Given a Σ-structure and an assignment μ of variables to elements of A, we can evaluate each formula.

- $A = \mathbb{Z}$
- $+^{\mathcal{A}}(x, y) = x + y$
- $<^{\mathcal{A}} (x, y) \iff x < y$
- $1^{\mathcal{A}} = 1$
- $\mu(x) = 1, \mu(y) = 3$

$$(1 <^{\mathcal{A}} 3) \wedge (3 +^{\mathcal{A}} 1 <^{A} 1)$$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$,
   - we suppose that $=^A$ is the identity relation.

Given a Σ-structure and an assignment μ of variables to elements of A, we can evaluate each formula.

- $A = \mathbb{Z}$
- $+^{\mathcal{A}}(x, y) = x + y$
- $<^{\mathcal{A}} (x, y) \iff x < y$
- $1^{\mathcal{A}} = 1$
- $\mu(x) = 1, \mu(y) = 3$

$$(1 <^{\mathcal{A}} 3) \wedge (4 <^{A} 1)$$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$,
   - we suppose that $=^{A}$ is the identity relation.

Given a Σ-structure and an assignment μ of variables to elements of A, we can evaluate each formula.

- $A = \mathbb{Z}$
- $+^{\mathcal{A}}(x, y) = x + y$
- $<^{\mathcal{A}}(x, y) \iff x < y$
- $1^{\mathcal{A}} = 1$
- $\mu(x) = 1, \mu(y) = 3$

$\top \wedge \bot$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$,
   - we suppose that $=^{A}$ is the identity relation.

Given a Σ-structure and an assignment μ of variables to elements of A, we can evaluate each formula.

- $A = \mathbb{Z}$
- $+^{\mathcal{A}}(x, y) = x + y$
- $<^{\mathcal{A}} (x, y) \iff x < y$ $\qquad\qquad\qquad \perp$
- $1^{\mathcal{A}} = 1$
- $\mu(x) = 1, \mu(y) = 3$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$.

Given a Σ-structure and an assignment μ from variables to elements of A, we can evaluate each formula.

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$.

Given a Σ-structure and an assignment μ from variables to elements of A, we can evaluate each formula.

- $A = \{\circ, \bullet\}$
- $+^{\mathcal{A}}(x, y) = y$
- $<^{\mathcal{A}}\ (x, y) = \{(\circ, \circ), (\bullet, \circ)\}$
- $1^{\mathcal{A}} = \bullet$
- $\mu(x) = \circ, \mu(y) = \circ$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$.

Given a Σ-structure and an assignment μ from variables to elements of A, we can evaluate each formula.

- $A = \{\circ, \bullet\}$
- $+^{\mathcal{A}}(x, y) = y$
- $<^{\mathcal{A}}\ (x, y) = \{(\circ, \circ), (\bullet, \circ)\}$
- $1^{\mathcal{A}} = \bullet$
- $\mu(x) = \circ, \mu(y) = \circ$

$$(x < y) \ \wedge \ (y + 1 < x)$$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$.

Given a Σ-structure and an assignment μ from variables to elements of A, we can evaluate each formula.

- $A = \{\circ, \bullet\}$
- $+^{\mathcal{A}}(x, y) = y$
- $<^{\mathcal{A}}(x, y) = \{(\circ, \circ), (\bullet, \circ)\}$
- $1^{\mathcal{A}} = \bullet$
- $\mu(x) = \circ, \mu(y) = \circ$

$$(\mu(x) <^{\mathcal{A}} \mu(y)) \wedge (\mu(y) +^{\mathcal{A}} 1^{\mathcal{A}} <^{A} \mu(x))$$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}}\colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$.

Given a Σ-structure and an assignment μ from variables to elements of A, we can evaluate each formula.

- $A = \{\circ, \bullet\}$
- $+^{\mathcal{A}}(x, y) = y$
- $<^{\mathcal{A}}(x, y) = \{(\circ, \circ), (\bullet, \circ)\}$
- $1^{\mathcal{A}} = \bullet$
- $\mu(x) = \circ, \mu(y) = \circ$

$$(\circ <^{\mathcal{A}} \circ) \wedge (\circ +^{\mathcal{A}} 1^{\mathcal{A}} <^{A} \circ)$$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$.

Given a Σ-structure and an assignment μ from variables to elements of A, we can evaluate each formula.

- $A = \{\circ, \bullet\}$
- $+^{\mathcal{A}}(x, y) = y$
- $<^{\mathcal{A}}(x, y) = \{(\circ, \circ), (\bullet, \circ)\}$
- $1^{\mathcal{A}} = \bullet$
- $\mu(x) = \circ, \mu(y) = \circ$

$$(\circ <^{\mathcal{A}} \circ) \land (\circ +^{\mathcal{A}} \bullet <^{A} \circ)$$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$.

Given a Σ-structure and an assignment μ from variables to elements of A, we can evaluate each formula.

- $A = \{\circ, \bullet\}$
- $+^{\mathcal{A}}(x, y) = y$
- $<^{\mathcal{A}} (x, y) = \{(\circ, \circ), (\bullet, \circ)\}$
- $1^{\mathcal{A}} = \bullet$
- $\mu(x) = \circ, \mu(y) = \circ$

$$(\circ <^{\mathcal{A}} \circ) \wedge (\bullet <^{A} \circ)$$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$.

Given a Σ-structure and an assignment μ from variables to elements of A, we can evaluate each formula.

- $A = \{\circ, \bullet\}$
- $+^{\mathcal{A}}(x, y) = y$
- $<^{\mathcal{A}} (x, y) = \{(\circ, \circ), (\bullet, \circ)\}$
- $1^{\mathcal{A}} = \bullet$
- $\mu(x) = \circ, \mu(y) = \circ$

$\top \wedge \top$

First-Order Logic – Semantics

A Σ-structure $\mathcal{A}$ is a pair of

1. a non-empty set A called the universe,
2. a map $(\_)^{\mathcal{A}}$ that
   - to each $f \in \Sigma^F$ assigns a function $f^{\mathcal{A}} \colon A^{\mathrm{ar}(f)} \to A$,
   - to each $R \in \Sigma^P$ assigns a relation $R^{\mathcal{A}} \subseteq A^{\mathrm{ar}(R)}$.

Given a Σ-structure and an assignment μ from variables to elements of A, we can evaluate each formula.

- $A = \{\circ, \bullet\}$
- $+^{\mathcal{A}}(x, y) = y$
- $<^{\mathcal{A}}\ (x, y) = \{(\circ, \circ), (\bullet, \circ)\}$ $\top$
- $1^{\mathcal{A}} = \bullet$
- $\mu(x) = \circ, \mu(y) = \circ$

First-Order Satisfiability

Definition

Formula φ is satisfiable if it evaluates to true for some Σ-structure $\mathcal{A}$ and a variable assignment μ.

First-Order Satisfiability

Definition

Formula φ is satisfiable if it evaluates to true for some Σ-structure $\mathcal{A}$ and a variable assignment μ.

Is formula $(x < y) \ \land \ (y + 1 < x)$ satisfiable?

First-Order Satisfiability

Definition

Formula φ is satisfiable if it evaluates to true for some Σ-structure $\mathcal{A}$ and a variable assignment μ.

Is formula $(x < y) \ \wedge \ (y + 1 < x)$ satisfiable? Yes. ☹

First-Order Satisfiability

Definition

Formula φ is satisfiable if it evaluates to true for some Σ-structure $\mathcal{A}$ and a variable assignment μ.

Is formula $(x < y) \ \wedge \ (y + 1 < x)$ satisfiable? Yes. ☹

Solution

Consider only well-behaved structures

This gives rise to the Satisfiability Modulo Theories

Satisfiability Modulo Theories

Definition

A (Σ-)theory is a set of Σ-structures.

Definition

A formula φ is satisfiable modulo theory T if it evaluates to true for some structure $\mathcal{A} \in T$ and a variable assignment μ.

Satisfiability Modulo Theories – Example

Consider the structure $\mathcal{Z}$ with the universe $\mathbb{Z}$ and the standard interpretation of operations $+$, $<$, and 1.

The formula $(x < y) \;\wedge\; (y + 1 < x)$ is unsatisfiable modulo theory $T = \{\mathcal{Z}\}$.

The formula $(x < y) \;\wedge\; (y < x + 2)$ is satisfiable modulo theory $T = \{\mathcal{Z}\}$.

Theories – Tour d’horizon

Theory of equality and uninterpreted functions

Theories – Tour d'horizon

Theory of equality and uninterpreted functions

- $\Sigma = \{=, f, g, h, \ldots\}$
- $T_=$ is a set of all Σ-structures

Theories – Tour d'horizon

Theory of equality and uninterpreted functions

- $\Sigma = \{=, f, g, h, \ldots\}$
- $T_=$ is a set of all Σ-structures

$$x = v \land y = g(z) \;\land\; f(g(x)) \neq f(y) \;\land\; z = v$$

Theories – Tour d’horizon

Theory of equality and uninterpreted functions

- $\Sigma = \{=, f, g, h, \ldots\}$
- $T_=$ is a set of all Σ-structures

$$x = v \land y = g(z) \;\land\; f(g(x)) \neq f(y) \;\land\; z = v$$

- satisfiability of arbitrary formulas is undecidable

Theories – Tour d'horizon

Theory of equality and uninterpreted functions

- $\Sigma = \{=, f, g, h, \ldots\}$
- $T_=$ is a set of all Σ-structures

$$x = v \land y = g(z) \ \land \ f(g(x)) \neq f(y) \ \land \ z = v$$

- satisfiability of arbitrary formulas is undecidable
- satisfiability of quantifier-free formulas is decidable (Ackermann, 1954)

Theories – Tour d'horizon

Theory of equality and uninterpreted functions

- $\Sigma = \{=, f, g, h, \ldots\}$
- $T_=$ is a set of all Σ-structures

$$x = v \land y = g(z) \;\land\; f(g(x)) \neq f(y) \;\land\; z = v$$

- satisfiability of arbitrary formulas is undecidable
- satisfiability of quantifier-free formulas is decidable (Ackermann, 1954)
- satisfiability of quantifier-free formulas is NP-complete

Theories – Tour d'horizon

Theory of equality and uninterpreted functions

- $\Sigma = \{=, f, g, h, \ldots\}$
- $T_=$ is a set of all Σ-structures

$$x = v \land y = g(z) \;\land\; f(g(x)) \neq f(y) \;\land\; z = v$$

- satisfiability of arbitrary formulas is undecidable
- satisfiability of quantifier-free formulas is decidable (Ackermann, 1954)
- satisfiability of quantifier-free formulas is NP-complete
- satisfiability of conjunctions of literals is in $\mathcal{O}(n \cdot \log(n))$

Theories – Tour d’horizon

Theory of linear integer arithmetic

Theories – Tour d’horizon

Theory of linear integer arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- T_{LIA} is a set of a single structure with $A = \mathbb{Z}$ and the standard interpretation of operations

Theories – Tour d'horizon

Theory of linear integer arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- T_{LIA} is a set of a single structure with $A = \mathbb{Z}$ and the standard interpretation of operations

$$1 \leqslant x \;\wedge\; (3 \leqslant x + y) \;\wedge\; (1 \leqslant y)$$

Theories – Tour d'horizon

Theory of linear integer arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- T_{LIA} is a set of a single structure with $A = \mathbb{Z}$ and the standard interpretation of operations

$$1 \leqslant x \ \wedge \ (3 \leqslant x + y) \ \wedge \ (1 \leqslant y)$$

- satisfiability of arbitrary formulas is decidable

Theories – Tour d'horizon

Theory of linear integer arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- T_{LIA} is a set of a single structure with $A = \mathbb{Z}$ and the standard interpretation of operations

$$1 \leqslant x \;\wedge\; (3 \leqslant x + y) \;\wedge\; (1 \leqslant y)$$

- satisfiability of arbitrary formulas is decidable
- complexity of satisfiability of arbitrary formulas is in $\Omega(2^{2^n})$ (Fischer, Rabin, 1974)

Theories – Tour d'horizon

Theory of linear integer arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- T_{LIA} is a set of a single structure with $A = \mathbb{Z}$ and the standard interpretation of operations

$$1 \leqslant x \ \wedge \ (3 \leqslant x + y) \ \wedge \ (1 \leqslant y)$$

- satisfiability of arbitrary formulas is decidable
- complexity of satisfiability of arbitrary formulas is in $\Omega(2^{2^n})$ (Fischer, Rabin, 1974)
- complexity of satisfiability of arbitrary formulas is in $\mathcal{O}(2^{2^{2^{kn}}})$ (Oppen, 1978)

Theories – Tour d'horizon

Theory of linear integer arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- T_{LIA} is a set of a single structure with $A = \mathbb{Z}$ and the standard interpretation of operations

$$1 \leqslant x \ \wedge \ (3 \leqslant x + y) \ \wedge \ (1 \leqslant y)$$

- satisfiability of arbitrary formulas is decidable
- complexity of satisfiability of arbitrary formulas is in $\Omega(2^{2^n})$ (Fischer, Rabin, 1974)
- complexity of satisfiability of arbitrary formulas is in $\mathcal{O}(2^{2^{2^{kn}}})$ (Oppen, 1978)
- satisfiability of quantifier-free formulas is NP-complete

Theories – Tour d'horizon

Theory of linear integer arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- T_{LIA} is a set of a single structure with $A = \mathbb{Z}$ and the standard interpretation of operations

$$1 \leqslant x \ \wedge \ (3 \leqslant x + y) \ \wedge \ (1 \leqslant y)$$

- satisfiability of arbitrary formulas is decidable
- complexity of satisfiability of arbitrary formulas is in $\Omega(2^{2^n})$ (Fischer, Rabin, 1974)
- complexity of satisfiability of arbitrary formulas is in $\mathcal{O}(2^{2^{2^{kn}}})$ (Oppen, 1978)
- satisfiability of quantifier-free formulas is NP-complete
- satisfiability of conjunctions of literals is NP-complete (folklore)

Theories – Tour d’horizon

Theory of linear rational arithmetic

Theories – Tour d'horizon

Theory of linear rational arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- $\mathrm{T}_{\mathrm{LRA}}$ is a set of a single structure with $A = \mathbb{Q}$ and the standard interpretation of operations

Theories – Tour d'horizon

Theory of linear rational arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- T_{LRA} is a set of a single structure with $A = \mathbb{Q}$ and the standard interpretation of operations

$$1 \leqslant x \;\wedge\; (3 \leqslant x + y) \;\wedge\; (1 \leqslant y)$$

Theories – Tour d'horizon

Theory of linear rational arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- T_{LRA} is a set of a single structure with $A = \mathbb{Q}$ and the standard interpretation of operations

$$1 \leqslant x \;\wedge\; (3 \leqslant x + y) \;\wedge\; (1 \leqslant y)$$

- satisfiability of arbitrary formulas is decidable

Theories – Tour d'horizon

Theory of linear rational arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- T_{LRA} is a set of a single structure with $A = \mathbb{Q}$ and the standard interpretation of operations

$$1 \leqslant x \ \wedge \ (3 \leqslant x + y) \ \wedge \ (1 \leqslant y)$$

- satisfiability of arbitrary formulas is decidable
- complexity of satisfiability of arbitrary formulas is in $\Omega(2^n)$ (Fischer, Rabin, 1974)

Theories – Tour d'horizon

Theory of linear rational arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- T_{LRA} is a set of a single structure with $A = \mathbb{Q}$ and the standard interpretation of operations

$$1 \leqslant x \ \wedge \ (3 \leqslant x + y) \ \wedge \ (1 \leqslant y)$$

- satisfiability of arbitrary formulas is decidable
- complexity of satisfiability of arbitrary formulas is in $\Omega(2^n)$ (Fischer, Rabin, 1974)
- complexity of satisfiability of arbitrary formulas is in $\mathcal{O}(2^{2^{kn}})$ (Ferrante, Rackoff, 1975)

Theories – Tour d'horizon

Theory of linear rational arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- T_{LRA} is a set of a single structure with $A = \mathbb{Q}$ and the standard interpretation of operations

$$1 \leqslant x \ \wedge \ (3 \leqslant x + y) \ \wedge \ (1 \leqslant y)$$

- satisfiability of arbitrary formulas is decidable
- complexity of satisfiability of arbitrary formulas is in $\Omega(2^n)$ (Fischer, Rabin, 1974)
- complexity of satisfiability of arbitrary formulas is in $\mathcal{O}(2^{2^{kn}})$ (Ferrante, Rackoff, 1975)
- satisfiability of quantifier-free formulas is NP-complete

Theories – Tour d'horizon

Theory of linear rational arithmetic

- $\Sigma = \{0, 1, +, -, =, \leqslant\}$
- T_{LRA} is a set of a single structure with $A = \mathbb{Q}$ and the standard interpretation of operations

$$1 \leqslant x \ \wedge \ (3 \leqslant x + y) \ \wedge \ (1 \leqslant y)$$

- satisfiability of arbitrary formulas is decidable
- complexity of satisfiability of arbitrary formulas is in $\Omega(2^n)$ (Fischer, Rabin, 1974)
- complexity of satisfiability of arbitrary formulas is in $\mathcal{O}(2^{2^{kn}})$ (Ferrante, Rackoff, 1975)
- satisfiability of quantifier-free formulas is NP-complete
- satisfiability of conjunctions of literals in P (Khachiyan, 1979)

Theories – Tour d’horizon

Theory of non-linear integer arithmetic

Theories – Tour d'horizon

Theory of non-linear integer arithmetic

- $\Sigma = \{0, 1, +, -, \cdot, =, \leqslant\}$
- $\mathrm{T}_{\mathrm{NIA}}$ is a set of a single structure with $A = \mathbb{Z}$ and the standard interpretation of operations

Theories – Tour d'horizon

Theory of non-linear integer arithmetic

- $\Sigma = \{0, 1, +, -, \cdot, =, \leqslant\}$
- T_{NIA} is a set of a single structure with $A = \mathbb{Z}$ and the standard interpretation of operations

$$1 \leqslant x \;\wedge\; (3 \leqslant x \cdot y) \;\wedge\; (1 \leqslant y)$$

Theories – Tour d'horizon

Theory of non-linear integer arithmetic

- $\Sigma = \{0, 1, +, -, \cdot, =, \leqslant\}$
- T_{NIA} is a set of a single structure with $A = \mathbb{Z}$ and the standard interpretation of operations

$$1 \leqslant x \;\wedge\; (3 \leqslant x \cdot y) \;\wedge\; (1 \leqslant y)$$

- satisfiability of conjunctions of quantifier-free formulas is undecidable (Matiyasevich, 1971)

Theories – Tour d’horizon

Theory of non-linear real arithmetic

Theories – Tour d’horizon

Theory of non-linear real arithmetic

- $\Sigma = \{0, 1, +, -, \cdot, =, \leqslant\}$
- T_{NRA} is a set of a single structure with $A = \mathbb{R}$ and the standard interpretation of operations

Theories – Tour d'horizon

Theory of non-linear real arithmetic

- $\Sigma = \{0, 1, +, -, \cdot, =, \leqslant\}$
- T_{NRA} is a set of a single structure with $A = \mathbb{R}$ and the standard interpretation of operations

$$1 \leqslant x \;\wedge\; (3 \leqslant x + y) \;\wedge\; (1 \leqslant y)$$

Theories – Tour d'horizon

Theory of non-linear real arithmetic

- $\Sigma = \{0, 1, +, -, \cdot, =, \leqslant\}$
- T_{NRA} is a set of a single structure with $A = \mathbb{R}$ and the standard interpretation of operations

$$1 \leqslant x \ \wedge \ (3 \leqslant x + y) \ \wedge \ (1 \leqslant y)$$

- satisfiability of arbitrary formulas is decidable (Tarski, 1951)

Theories – Tour d'horizon

Theory of non-linear real arithmetic

- $\Sigma = \{0, 1, +, -, \cdot, =, \leqslant\}$
- T_{NRA} is a set of a single structure with $A = \mathbb{R}$ and the standard interpretation of operations

$$1 \leqslant x \;\wedge\; (3 \leqslant x + y) \;\wedge\; (1 \leqslant y)$$

- satisfiability of arbitrary formulas is decidable (Tarski, 1951)
- complexity of satisfiability of arbitrary formulas is in $\mathcal{O}(2^{2^{kn}})$ (Collins, 1975)

Theories – Tour d'horizon

Theory of non-linear real arithmetic

- $\Sigma = \{0, 1, +, -, \cdot, =, \leqslant\}$
- T_{NRA} is a set of a single structure with $A = \mathbb{R}$ and the standard interpretation of operations

$$1 \leqslant x \ \wedge \ (3 \leqslant x + y) \ \wedge \ (1 \leqslant y)$$

- satisfiability of arbitrary formulas is decidable (Tarski, 1951)
- complexity of satisfiability of arbitrary formulas is in $\mathcal{O}(2^{2^{kn}})$ (Collins, 1975)
- complexity of satisfiability of conjunctions of literals in $\mathcal{O}(2^{2^{kn}})$

Theories – Tour d’horizon

Theory of arrays

Theories – Tour d'horizon

Theory of arrays

- $\Sigma = \{\text{read}, \text{write}, =\}$
- T_A is a set of structures, where A is a set of arrays and elements and
  - $\text{read}(a, i)$ is interpreted as an element on index i of array a
  - $\text{write}(a, i, v)$ is interpreted as an array a after replacing element on index i by v
  - equality is defined only for elements

Theories – Tour d'horizon

Theory of arrays

- $\Sigma = \{\mathrm{read}, \mathrm{write}, =\}$
- T_A is a set of structures, where A is a set of arrays and elements and
  - $\mathrm{read}(a, i)$ is interpreted as an element on index i of array a
  - $\mathrm{write}(a, i, v)$ is interpreted as an array a after replacing element on index i by v
  - equality is defined only for elements

$$\mathrm{read}(a, i) = u \ \wedge \ (b = \mathrm{write}(a, i, v)) \ \wedge \ (\mathrm{read}(a, i) = \mathrm{read}(b, i))$$

Theories – Tour d'horizon

Theory of arrays

- $\Sigma = \{\mathrm{read}, \mathrm{write}, =\}$
- T_A is a set of structures, where A is a set of arrays and elements and
  - $\mathrm{read}(a, i)$ is interpreted as an element on index i of array a
  - $\mathrm{write}(a, i, v)$ is interpreted as an array a after replacing element on index i by v
  - equality is defined only for elements

$$\mathrm{read}(a, i) = u \ \wedge \ (b = \mathrm{write}(a, i, v)) \ \wedge \ (\mathrm{read}(a, i) = \mathrm{read}(b, i))$$

- satisfiability of arbitrary formulas is undecidable

Theories – Tour d'horizon

Theory of arrays

- $\Sigma = \{\mathrm{read}, \mathrm{write}, =\}$
- T_A is a set of structures, where A is a set of arrays and elements and
  - $\mathrm{read}(a, i)$ is interpreted as an element on index i of array a
  - $\mathrm{write}(a, i, v)$ is interpreted as an array a after replacing element on index i by v
  - equality is defined only for elements

$$\mathrm{read}(a, i) = u \ \wedge \ (b = \mathrm{write}(a, i, v)) \ \wedge \ (\mathrm{read}(a, i) = \mathrm{read}(b, i))$$

- satisfiability of arbitrary formulas is undecidable
- satisfiability of quantifier-free formulas is NP-complete

Theories – Tour d'horizon

Theory of arrays with extensionality

Theories – Tour d'horizon

Theory of arrays with extensionality

- $\Sigma = \{\mathrm{read}, \mathrm{write}, =\}$
- T_A is a set of structures, where A is a set of arrays and elements and
  - $\mathrm{read}(a, i)$ is interpreted as an element on index i of array a
  - $\mathrm{write}(a, i, v)$ is interpreted as an array a after replacing element on index i by v

Theories – Tour d'horizon

Theory of arrays with extensionality

- $\Sigma = \{\mathrm{read}, \mathrm{write}, =\}$
- T_A is a set of structures, where A is a set of arrays and elements and
  - $\mathrm{read}(a, i)$ is interpreted as an element on index i of array a
  - $\mathrm{write}(a, i, v)$ is interpreted as an array a after replacing element on index i by v

$$\mathrm{read}(a, i) = u \;\wedge\; (b = \mathrm{write}(a, i, v)) \;\wedge\; (a = b)$$

Theories – Tour d’horizon

Theory of arrays with extensionality

- $\Sigma = \{\mathrm{read}, \mathrm{write}, =\}$
- T_A is a set of structures, where A is a set of arrays and elements and
  - $\mathrm{read}(a, i)$ is interpreted as an element on index i of array a
  - $\mathrm{write}(a, i, v)$ is interpreted as an array a after replacing element on index i by v

$$\mathrm{read}(a, i) = u \;\wedge\; (b = \mathrm{write}(a, i, v)) \;\wedge\; (a = b)$$

- satisfiability of arbitrary formulas is undecidable

Theories – Tour d'horizon

Theory of arrays with extensionality

- $\Sigma = \{\mathrm{read}, \mathrm{write}, =\}$
- T_A is a set of structures, where A is a set of arrays and elements and
  - $\mathrm{read}(a, i)$ is interpreted as an element on index i of array a
  - $\mathrm{write}(a, i, v)$ is interpreted as an array a after replacing element on index i by v

$$\mathrm{read}(a, i) = u \;\wedge\; (b = \mathrm{write}(a, i, v)) \;\wedge\; (a = b)$$

- satisfiability of arbitrary formulas is undecidable
- satisfiability of quantifier-free formulas is NP-complete

Theories – Tour d'horizon

More theories:

- theory of bit-vectors,

Theories – Tour d'horizon

More theories:

- theory of bit-vectors,
- theory of strings,

Theories – Tour d’horizon

More theories:

- theory of bit-vectors,
- theory of strings,
- theory of lists,

Theories – Tour d'horizon

More theories:

- theory of bit-vectors,
- theory of strings,
- theory of lists,
- theory of recursive data structures,

Theories – Tour d'horizon

More theories:

- theory of bit-vectors,
- theory of strings,
- theory of lists,
- theory of recursive data structures,
- theory of groups,

Theories – Tour d’horizon

More theories:

- theory of bit-vectors,
- theory of strings,
- theory of lists,
- theory of recursive data structures,
- theory of groups,
- …

Two views of theories

A theory can be also viewed as a set of closed Σ-formulas (axioms).

A formula is then satisfiable modulo T iff it is true for some structure that satisfies all axioms in T.

Two views of theories

A theory can be also viewed as a set of closed Σ-formulas (axioms).

A formula is then satisfiable modulo T iff it is true for some structure that satisfies all axioms in T.

Example

- The theory of uninterpreted functions with equality is $\mathsf{T}_{=} = \emptyset$

Two views of theories

A theory can be also viewed as a set of closed Σ-formulas (axioms).

A formula is then satisfiable modulo T iff it is true for some structure that satisfies all axioms in T.

Example

- The theory of uninterpreted functions with equality is $T_= = \emptyset$
- The theory of arrays is

$$\begin{aligned} T_A = \{ & \forall a, i, j.\, (i = j \ \rightarrow\ read(a, i) = read(a, j)), \\ & \forall a, v, i, j.\, (i = j \ \rightarrow\ read(write(a, i, v), j) = v), \\ & \forall a, v, i, j.\, (i \neq j \ \rightarrow\ read(write(a, i, v), j) = read(a, j))\} \end{aligned}$$

Two views of theories

These two views are equivalent

Two views of theories

These two views are equivalent

- To the set of Σ-structures assign the set of formulas that are true in all these structures.

Two views of theories

These two views are equivalent

- To the set of Σ-structures assign the set of formulas that are true in all these structures.
- To the set of axioms assign the set of Σ-structures that satisfy all the axioms.

Two views of theories

These two views are equivalent

- To the set of Σ-structures assign the set of formulas that are true in all these structures.
- To the set of axioms assign the set of Σ-structures that satisfy all the axioms.

Two views of theories

These two views are equivalent

- To the set of Σ-structures assign the set of formulas that are true in all these structures.
- To the set of axioms assign the set of Σ-structures that satisfy all the axioms.

Sometimes, one view is better

- A set of structures satisfying axioms of Peano arithmetic is not easily describable.

Two views of theories

These two views are equivalent

- To the set of Σ-structures assign the set of formulas that are true in all these structures.
- To the set of axioms assign the set of Σ-structures that satisfy all the axioms.

Sometimes, one view is better

- A set of structures satisfying axioms of Peano arithmetic is not easily describable.
- A set of axioms for NRA is infinite and complicated.

Two views of theories

These two views are equivalent

- To the set of Σ-structures assign the set of formulas that are true in all these structures.
- To the set of axioms assign the set of Σ-structures that satisfy all the axioms.

Sometimes, one view is better

- A set of structures satisfying axioms of Peano arithmetic is not easily describable.
- A set of axioms for NRA is infinite and complicated.
- A set of axioms for NIA is not recursive.

Two views of theories

These two views are equivalent

- To the set of Σ-structures assign the set of formulas that are true in all these structures.
- To the set of axioms assign the set of Σ-structures that satisfy all the axioms.

Sometimes, one view is better

- A set of structures satisfying axioms of Peano arithmetic is not easily describable.
- A set of axioms for NRA is infinite and complicated.
- A set of axioms for NIA is not recursive.

Two views of theories

These two views are equivalent

- To the set of Σ-structures assign the set of formulas that are true in all these structures.
- To the set of axioms assign the set of Σ-structures that satisfy all the axioms.

Sometimes, one view is better

- A set of structures satisfying axioms of Peano arithmetic is not easily describable.
- A set of axioms for NRA is infinite and complicated.
- A set of axioms for NIA is not recursive. (, 1931)

Two views of theories

These two views are equivalent

- To the set of Σ-structures assign the set of formulas that are true in all these structures.
- To the set of axioms assign the set of Σ-structures that satisfy all the axioms.

Sometimes, one view is better

- A set of structures satisfying axioms of Peano arithmetic is not easily describable.
- A set of axioms for NRA is infinite and complicated.
- A set of axioms for NIA is not recursive. (Gödel, 1931)

Deciding satisfiability modulo theories

Two approaches to SMT

- eager
- lazy

Deciding satisfiability modulo theories

Two approaches to SMT

- eager
- lazy

Eager approach
Encode the formula to SAT

Deciding satisfiability modulo theories

Two approaches to SMT

- eager
- lazy

Eager approach
Encode the formula to SAT

Lazy approach
Use a SAT solver to reason about Boolean structure of the formula and a specialized T-solver to reason about the constraints imposed by the theory.

Deciding satisfiability modulo theories

Two approaches to SMT

- eager
- lazy

Eager approach
Encode the formula to SAT

Lazy approach
Use a SAT solver to reason about Boolean structure of the formula and a specialized T-solver to reason about the constraints imposed by the theory.

Lazy SMT solvers can be further divided to:

- offline
- online

Offline SMT solving

Suppose the formula is in conjunctive normal form – a conjunction of disjunctions of Σ-literals.

Offline SMT solving

Suppose the formula is in conjunctive normal form – a conjunction of disjunctions of Σ-literals.

And suppose we have a solver that can decide conjunctions of Σ-literals – a T-solver.

Offline SMT solving

Suppose the formula is in conjunctive normal form – a conjunction of disjunctions of Σ-literals.

And suppose we have a solver that can decide conjunctions of Σ-literals – a T-solver.

Offline SMT

- Treat each literal as a boolean variable.

Offline SMT solving

Suppose the formula is in conjunctive normal form – a conjunction of disjunctions of Σ-literals.

And suppose we have a solver that can decide conjunctions of Σ-literals – a T-solver.

Offline SMT

- Treat each literal as a boolean variable.
- Use a SAT solver to get a Boolean model of the formula.

Offline SMT solving

Suppose the formula is in conjunctive normal form – a conjunction of disjunctions of Σ-literals.

And suppose we have a solver that can decide conjunctions of Σ-literals – a T-solver.

Offline SMT

- Treat each literal as a boolean variable.
- Use a SAT solver to get a Boolean model of the formula.
- Use a T-solver to check whether the model is satisfiable in the theory (T-consistent).

Offline SMT solving

Suppose the formula is in conjunctive normal form – a conjunction of disjunctions of Σ-literals.

And suppose we have a solver that can decide conjunctions of Σ-literals – a T-solver.

Offline SMT

- Treat each literal as a boolean variable.
- Use a SAT solver to get a Boolean model of the formula.
- Use a T-solver to check whether the model is satisfiable in the theory (T-consistent).
- If not, add a clause that prohibits this Boolean model and repeat.

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6)$$

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$p \;\wedge\; (q \vee r) \;\wedge\; (s \vee t)$$

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \ \wedge \ (y < 3 \vee y > 5) \ \wedge \ (x + y = 4 \vee y = 6)$$

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6)$$

Boolean model
$x = 1,\; y < 3,\; x + y = 4$

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6)$$

Boolean model
$x = 1,\; y < 3,\; x + y = 4$

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \ \wedge \ (y < 3 \vee y > 5) \ \wedge \ (x + y = 4 \vee y = 6)$$

Boolean model
$x = 1, \ y < 3, \ x + y = 4$
Not satisfiable in the theory

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge$$
$$(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4))$$

Boolean model
$x = 1,\; y < 3,\; x + y = 4$
Not satisfiable in the theory

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$\begin{aligned} &p \;\wedge\; (q \vee r) \;\wedge\; (s \vee t) \;\wedge \\ &(\neg p \vee \neg q \vee \neg s) \end{aligned}$$

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \ \wedge \ (y < 3 \vee y > 5) \ \wedge \ (x + y = 4 \vee y = 6) \ \wedge$$
$$(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4))$$

Boolean model

$x = 1, \ y < 3, \ \neg(x + y = 4), \ y = 6$

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge$$
$$(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4))$$

Boolean model
$x = 1,\; y < 3,\; \neg(x + y = 4),\; y = 6$
Not satisfiable in the theory

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$
\begin{aligned}
& x = 1 \ \wedge \ (y < 3 \vee y > 5) \ \wedge \ (x + y = 4 \vee y = 6) \ \wedge \\
& (\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4)) \ \wedge \\
& (\neg(x = 1) \vee \neg(y < 3) \vee (x + y = 4) \vee \neg(y = 6))
\end{aligned}
$$

Boolean model
$x = 1,\ y < 3,\ \neg(x + y = 4),\ y = 6$
Not satisfiable in the theory

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$
\begin{aligned}
&x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge \\
&(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4)) \;\wedge \\
&(\neg(x = 1) \vee \neg(y < 3) \vee (x + y = 4) \vee \neg(y = 6))
\end{aligned}
$$

Boolean model

$x = 1,\; \neg(y < 3),\; y > 5,\; (x + y = 4)$

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$\begin{aligned}
&x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge \\
&(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4)) \;\wedge \\
&(\neg(x = 1) \vee \neg(y < 3) \vee (x + y = 4) \vee \neg(y = 6))
\end{aligned}$$

Boolean model
$x = 1,\ \neg(y < 3),\ y > 5,\ (x + y = 4)$
Not satisfiable in the theory

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$\begin{aligned}
&x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge \\
&(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4)) \;\wedge \\
&(\neg(x = 1) \vee \neg(y < 3) \vee (x + y = 4) \vee \neg(y = 6)) \;\wedge \\
&(\neg(x = 1) \vee (y < 3) \vee \neg(y > 5) \vee \neg(x + y = 4))
\end{aligned}$$

Boolean model
$x = 1,\; \neg(y < 3),\; y > 5,\; (x + y = 4)$
Not satisfiable in the theory

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$
\begin{aligned}
& x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge \\
& (\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4)) \;\wedge \\
& (\neg(x = 1) \vee \neg(y < 3) \vee (x + y = 4) \vee \neg(y = 6)) \;\wedge \\
& (\neg(x = 1) \vee (y < 3) \vee \neg(y > 5) \vee \neg(x + y = 4))
\end{aligned}
$$

Boolean model

$x = 1,\; \neg(y < 3),\; y > 5,\; \neg(x + y = 4),\; y = 6$

Offline SMT solving – example

The formula φ over linear integer arithmetic:

$$\begin{aligned}
&x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge \\
&(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4)) \;\wedge \\
&(\neg(x = 1) \vee \neg(y < 3) \vee (x + y = 4) \vee \neg(y = 6)) \;\wedge \\
&(\neg(x = 1) \vee (y < 3) \vee \neg(y > 5) \vee \neg(x + y = 4))
\end{aligned}$$

Boolean model
$x = 1,\; \neg(y < 3),\; y > 5,\; \neg(x + y = 4),\; y = 6$
Satisfiable in the theory! ($x = 1, y = 6$)

Online SMT solving – towards CDCL(T)

Online SMT
Integrate the CDCL SAT solver and the T-solver more tightly.

After a T-conflict, the T-solver provides the conflict clause and the search backtracks.

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6)$$

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6)$$

Partial assignment

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6)$$

Partial assignment
$x = 1$

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6)$$

Partial assignment
$x = 1,\ (y < 3)^{d}$

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6)$$

Partial assignment
$x = 1,\ (y < 3)^d,\ (x + y = 4)^d$

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge$$
$$(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4))$$

Partial assignment
$x = 1,\ (y < 3)^{d},\ (x + y = 4)^{d}$

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \ \wedge \ (y < 3 \vee y > 5) \ \wedge \ (x + y = 4 \vee y = 6) \ \wedge$$
$$(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4))$$

Partial assignment
$x = 1,\ (y < 3)^d,\ \neg(x + y = 4)$

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge$$
$$(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4))$$

Partial assignment
$x = 1,\ (y < 3)^d,\ \neg(x + y = 4),\ (y = 6)$

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \ \wedge \ (y < 3 \vee y > 5) \ \wedge \ (x + y = 4 \vee y = 6) \ \wedge$$
$$(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4)) \ \wedge$$
$$(\neg(y < 3) \vee \neg(y = 6))$$

Partial assignment
$x = 1$, $(y < 3)^d$, $\neg(x + y = 4)$, $(y = 6)$

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \ \wedge \ (y < 3 \vee y > 5) \ \wedge \ (x + y = 4 \vee y = 6) \ \wedge$$
$$(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4)) \wedge$$
$$(\neg(y < 3) \vee \neg(y = 6))$$

Partial assignment

$x = 1, \neg(y < 3)$

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge$$
$$(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4)) \;\wedge$$
$$(\neg(y < 3) \vee \neg(y = 6))$$

Partial assignment

$x = 1,\ \neg(y < 3),\ (y > 5)$

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \ \wedge \ (y < 3 \vee y > 5) \ \wedge \ (x + y = 4 \vee y = 6) \ \wedge$$
$$(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4)) \ \wedge$$
$$(\neg(y < 3) \vee \neg(y = 6))$$

Partial assignment

$x = 1, \neg(y < 3), (y > 5), (x + y = 4)^d$

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$\begin{aligned}
&x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge \\
&(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4)) \;\wedge \\
&(\neg(y < 3) \vee \neg(y = 6)) \;\wedge \\
&(\neg(x = 1) \vee \neg(y > 5) \vee \neg(x + y = 4))
\end{aligned}$$

Partial assignment

$x = 1,\ \neg(y < 3),\ (y > 5),\ (x + y = 4)^{d}$

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$\begin{aligned}
&x = 1 \ \wedge\ (y < 3 \vee y > 5) \ \wedge\ (x + y = 4 \vee y = 6) \ \wedge \\
&(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4)) \ \wedge \\
&(\neg(y < 3) \vee \neg(y = 6)) \ \wedge \\
&(\neg(x = 1) \vee \neg(y > 5) \vee \neg(x + y = 4))
\end{aligned}$$

Partial assignment

$x = 1,\ \neg(y < 3),\ (y > 5),\ \neg(x + y = 4)$

Online SMT solving – example

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge$$
$$(\neg(x = 1) \vee \neg(y < 3) \vee \neg(x + y = 4)) \;\wedge$$
$$(\neg(y < 3) \vee \neg(y = 6)) \;\wedge$$
$$(\neg(x = 1) \vee \neg(y > 5) \vee \neg(x + y = 4))$$

Partial assignment

$x = 1$, $\neg(y < 3)$, $(y > 5)$, $\neg(x + y = 4)$, $(y = 6)$

Theory propagation

The T-solver can guide the search, if a value of a literal is implied by the current partial assignment.

Theory propagation

The T-solver can guide the search, if a value of a literal is implied by the current partial assignment.

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6)$$

Partial assignment

Theory propagation

The T-solver can guide the search, if a value of a literal is implied by the current partial assignment.

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6)$$

Partial assignment

$x = 1$

Theory propagation

The T-solver can guide the search, if a value of a literal is implied by the current partial assignment.

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6)$$

Partial assignment

$x = 1,\ (y < 3)^{d}$

Theory propagation

The T-solver can guide the search, if a value of a literal is implied by the current partial assignment.

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6)$$

Partial assignment
$x = 1,\ (y < 3)^{d},\ \neg(x + y = 4)$

Theory propagation

The T-solver can guide the search, if a value of a literal is implied by the current partial assignment.

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6)$$

Partial assignment

$x = 1,\ (y < 3)^{d},\ \neg(x + y = 4),\ (y = 6)$

Theory propagation

The T-solver can guide the search, if a value of a literal is implied by the current partial assignment.

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge$$
$$(\neg(y < 3) \vee \neg(y = 6))$$

Partial assignment

$x = 1,\ (y < 3)^{d},\ \neg(x + y = 4),\ (y = 6)$

Theory propagation

The T-solver can guide the search, if a value of a literal is implied by the current partial assignment.

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge\; (\neg(y < 3) \vee \neg(y = 6))$$

Partial assignment

$x = 1$, $\neg(y < 3)$

Theory propagation

The T-solver can guide the search, if a value of a literal is implied by the current partial assignment.

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge$$
$$(\neg(y < 3) \vee \neg(y = 6))$$

Partial assignment

$x = 1$, $\neg(y < 3)$, $(y > 5)$

Theory propagation

The T-solver can guide the search, if a value of a literal is implied by the current partial assignment.

The formula φ over linear integer arithmetic:

$$x = 1 \ \wedge \ (y < 3 \vee y > 5) \ \wedge \ (x + y = 4 \vee y = 6) \ \wedge$$
$$(\neg(y < 3) \vee \neg(y = 6))$$

Partial assignment

$x = 1$, $\neg(y < 3)$, $(y > 5)$, $\neg(x + y = 4)$

Theory propagation

The T-solver can guide the search, if a value of a literal is implied by the current partial assignment.

The formula φ over linear integer arithmetic:

$$x = 1 \;\wedge\; (y < 3 \vee y > 5) \;\wedge\; (x + y = 4 \vee y = 6) \;\wedge\; (\neg(y < 3) \vee \neg(y = 6))$$

Partial assignment

$x = 1$, $\neg(y < 3)$, $(y > 5)$, $\neg(x + y = 4)$, $(y = 6)$

Further enhancements

Online approach can be further improved:

Further enhancements

Online approach can be further improved:

- early pruning,

Further enhancements

Online approach can be further improved:

- early pruning,
- restarts,

Further enhancements

Online approach can be further improved:

- early pruning,
- restarts,
- lemmas on demand.

CDCL(T)

The approach consisting of cooperation of a CDCL SAT solver with a specialized T-solver is called CDCL(T).

T-solver can be instantiated arbitrarily, but it should

CDCL(T)

The approach consisting of cooperation of a CDCL SAT solver with a specialized T-solver is called CDCL(T).

T-solver can be instantiated arbitrarily, but it should

- handle assignment of literal values efficiently,

CDCL(T)

The approach consisting of cooperation of a CDCL SAT solver with a specialized T-solver is called CDCL(T).

T-solver can be instantiated arbitrarily, but it should

- handle assignment of literal values efficiently,
- provide reasons for theory conflicts,

CDCL(T)

The approach consisting of cooperation of a CDCL SAT solver with a specialized T-solver is called CDCL(T).

T-solver can be instantiated arbitrarily, but it should

- handle assignment of literal values efficiently,
- provide reasons for theory conflicts,
- backtrack efficiently.

CDCL(T)

The approach consisting of cooperation of a CDCL SAT solver with a specialized T-solver is called CDCL(T).

T-solver can be instantiated arbitrarily, but it should

- handle assignment of literal values efficiently,
- provide reasons for theory conflicts,
- backtrack efficiently.

CDCL(T)

The approach consisting of cooperation of a CDCL SAT solver with a specialized T-solver is called CDCL(T).

T-solver can be instantiated arbitrarily, but it should

- handle assignment of literal values efficiently,
- provide reasons for theory conflicts,
- backtrack efficiently.

It further can

CDCL(T)

The approach consisting of cooperation of a CDCL SAT solver with a specialized T-solver is called CDCL(T).

T-solver can be instantiated arbitrarily, but it should

- handle assignment of literal values efficiently,
- provide reasons for theory conflicts,
- backtrack efficiently.

It further can

- perform theory propagation (identify implied literals),

CDCL(T)

The approach consisting of cooperation of a CDCL SAT solver with a specialized T-solver is called CDCL(T).

T-solver can be instantiated arbitrarily, but it should

- handle assignment of literal values efficiently,
- provide reasons for theory conflicts,
- backtrack efficiently.

It further can

- perform theory propagation (identify implied literals),
- perform early pruning (identify theory conflicts during the search).

CDCL($T_=$)

Let's consider CDCL($T_=$) with the details of the $T_=$-solver.

$$x = f(y) \ \wedge\ (y = z \vee x = y) \ \wedge\ (g(f(z)) \neq g(x) \vee f(z) \neq f(y))$$

CDCL($T_=$)

Let's consider CDCL($T_=$) with the details of the $T_=$-solver.

$$x = f(y) \;\wedge\; (y = z \vee x = y) \;\wedge\; (g(f(z)) \neq g(x) \vee f(z) \neq f(y))$$

Partial assignment

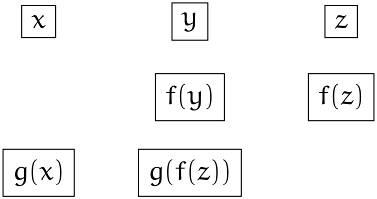
CDCL($T_=$)

Let's consider CDCL($T_=$) with the details of the $T_=$-solver.

$$x = f(y) \;\wedge\; (y = z \vee x = y) \;\wedge\; (g(f(z)) \neq g(x) \vee f(z) \neq f(y))$$

Partial assignment

Equality graph



CDCL($T_=$)

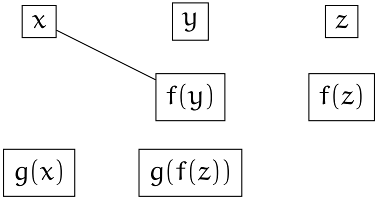
Let's consider CDCL($T_=$) with the details of the $T_=$-solver.

$$x = f(y) \;\wedge\; (y = z \vee x = y) \;\wedge\; (g(f(z)) \neq g(x) \vee f(z) \neq f(y))$$

Partial assignment

$(x = f(y))$

Equality graph



CDCL($T_=$)

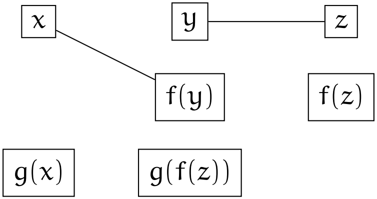
Let's consider CDCL($T_=$) with the details of the $T_=$-solver.

$$x = f(y) \;\wedge\; (y = z \vee x = y) \;\wedge\; (g(f(z)) \neq g(x) \vee f(z) \neq f(y))$$

Partial assignment

$(x = f(y)),\ (y = z)^d$

Equality graph



CDCL($T_=$)

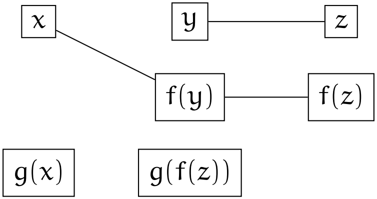
Let's consider CDCL($T_=$) with the details of the $T_=$-solver.

$$x = f(y) \;\wedge\; (y = z \vee x = y) \;\wedge\; (g(f(z)) \neq g(x) \vee f(z) \neq f(y))$$

Partial assignment

$(x = f(y)),\ (y = z)^d,\ (f(z) = f(y))$

Equality graph



CDCL($T_=$)

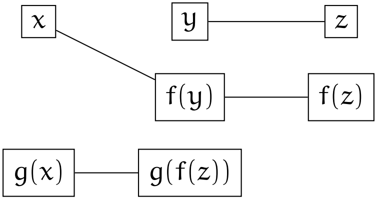
Let's consider CDCL($T_=$) with the details of the $T_=$-solver.

$$x = f(y) \;\wedge\; (y = z \vee x = y) \;\wedge\; (g(f(z)) \neq g(x) \vee f(z) \neq f(y))$$

Partial assignment

$(x = f(y))$, $(y = z)^d$, $(f(z) = f(y))$, $(g(f(z)) = g(x))$

Equality graph



CDCL($T_=$)

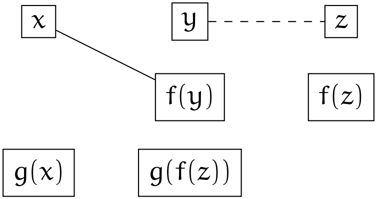
Let's consider CDCL($T_=$) with the details of the $T_=$-solver.

$$x = f(y) \;\wedge\; (y = z \vee x = y) \;\wedge\; (g(f(z)) \neq g(x) \vee f(z) \neq f(y))$$

Partial assignment

$(x = f(y))$, $(y \neq z)$

Equality graph



CDCL($T_=$)

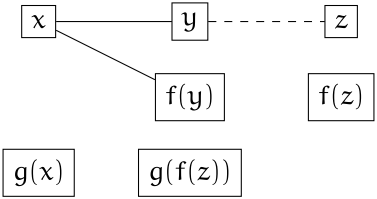
Let's consider CDCL($T_=$) with the details of the $T_=$-solver.

$$x = f(y) \;\wedge\; (y = z \vee x = y) \;\wedge\; (g(f(z)) \neq g(x) \vee f(z) \neq f(y))$$

Partial assignment

$(x = f(y)), (y \neq z), (x = y)$

Equality graph



CDCL($T_=$)

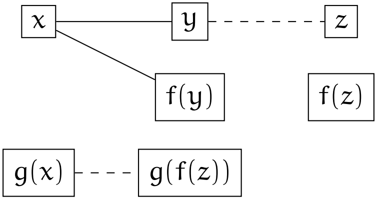
Let's consider CDCL($T_=$) with the details of the $T_=$-solver.

$$x = f(y) \;\wedge\; (y = z \vee x = y) \;\wedge\; (g(f(z)) \neq g(x) \vee f(z) \neq f(y))$$

Partial assignment

$(x = f(y))$, $(y \neq z)$, $(x = y)$, $(g(f(z)) \neq g(x))^d$

Equality graph



Further schedule

- 10. 3. – Combination of Theories (Fanda)

Further schedule

- 10. 3. – Combination of Theories (Fanda)
- 17. 3. – A Tale Of Two Solvers: Eager and Lazy Approaches to Bit-Vectors (Honza)

Further schedule

- 10. 3. – Combination of Theories (Fanda)
- 17. 3. – A Tale Of Two Solvers: Eager and Lazy Approaches to Bit-Vectors (Honza)
- 24. 3. – Abstract Conflict-Driven Clause Learning (Martin)

Further schedule

- 10. 3. – Combination of Theories (Fanda)
- 17. 3. – A Tale Of Two Solvers: Eager and Lazy Approaches to Bit-Vectors (Honza)
- 24. 3. – Abstract Conflict-Driven Clause Learning (Martin)
- 31. 3. – Deciding Bit-Vector Formulas with mcSAT (Marťa)

Further schedule

- 10. 3. – Combination of Theories (Fanda)
- 17. 3. – A Tale Of Two Solvers: Eager and Lazy Approaches to Bit-Vectors (Honza)
- 24. 3. – Abstract Conflict-Driven Clause Learning (Martin)
- 31. 3. – Deciding Bit-Vector Formulas with mcSAT (Marťa)
- 7. 4. – Complexity of Fixed-Size Bit-Vector Logics (Vláďa)

Further schedule

- 10. 3. – Combination of Theories (Fanda)
- 17. 3. – A Tale Of Two Solvers: Eager and Lazy Approaches to Bit-Vectors (Honza)
- 24. 3. – Abstract Conflict-Driven Clause Learning (Martin)
- 31. 3. – Deciding Bit-Vector Formulas with mcSAT (Marťa)
- 7. 4. – Complexity of Fixed-Size Bit-Vector Logics (Vláďa)
- 21. 4. – Counterexample-Guided Model Synthesis (Marek)

Further schedule

- 10. 3. – Combination of Theories (Fanda)
- 17. 3. – A Tale Of Two Solvers: Eager and Lazy Approaches to Bit-Vectors (Honza)
- 24. 3. – Abstract Conflict-Driven Clause Learning (Martin)
- 31. 3. – Deciding Bit-Vector Formulas with mcSAT (Marťa)
- 7. 4. – Complexity of Fixed-Size Bit-Vector Logics (Vláďa)
- 21. 4. – Counterexample-Guided Model Synthesis (Marek)
- 5. 5. – Seminator (Fanda)

Further schedule

- 10. 3. – Combination of Theories (Fanda)
- 17. 3. – A Tale Of Two Solvers: Eager and Lazy Approaches to Bit-Vectors (Honza)
- 24. 3. – Abstract Conflict-Driven Clause Learning (Martin)
- 31. 3. – Deciding Bit-Vector Formulas with mcSAT (Marťa)
- 7. 4. – Complexity of Fixed-Size Bit-Vector Logics (Vláďa)
- 21. 4. – Counterexample-Guided Model Synthesis (Marek)
- 5. 5. – Seminator (Fanda)
- 12. 5. – Effective word-level interpolation for software verification (Viki)

Further schedule

- 10. 3. – Combination of Theories (Fanda)
- 17. 3. – A Tale Of Two Solvers: Eager and Lazy Approaches to Bit-Vectors (Honza)
- 24. 3. – Abstract Conflict-Driven Clause Learning (Martin)
- 31. 3. – Deciding Bit-Vector Formulas with mcSAT (Marťa)
- 7. 4. – Complexity of Fixed-Size Bit-Vector Logics (Vláďa)
- 21. 4. – Counterexample-Guided Model Synthesis (Marek)
- 5. 5. – Seminator (Fanda)
- 12. 5. – Effective word-level interpolation for software verification (Viki)
- 19. 5. – An Approximation Framework for Solvers and Decision Procedures (Katka)